EVALUATING INFLUENCE DIAGRAMS

ROSS D. SHACHTER
Stanford University, Stanford, California

(Received June 1984; revision received January 1986; accepted February 1986)

An influence diagram is a graphical structure for modeling uncertain variables and decisions and explicitly revealing probabilistic dependence and the flow of information. It is an intuitive framework in which to formulate problems as perceived by decision makers and to incorporate the knowledge of experts. At the same time, it is a precise description of information that can be stored and manipulated by a computer. We develop an algorithm that can evaluate any well-formed influence diagram and determine the optimal policy for its decisions. Since the diagram can be analyzed directly, there is no need to construct other representations such as a decision tree. As a result, the analysis can be performed using the decision maker's perspective on the problem. Questions of sensitivity and the value of information are natural and easily posed. Modifications to the model suggested by such analyses can be made directly to the problem formulation, and then evaluated directly.

Many practical problems in operations research are characterized by a large number of interrelated uncertain quantities and alternatives. Decision analysis has been developed to address these problems analytically, based on a normative axiomatic framework. Unfortunately, this approach often transforms the problem as perceived by the decision maker into a different representation, such as a decision tree, in order to evaluate it. Many decision makers have resisted this approach, despite its fundamental basis, in favor of more ad hoc procedures that let them maintain their way of thinking about the problem.

The influence diagram has been designed as a knowledge representation to bridge the gap between analysis and formulation. It is intuitive enough to communicate with decision makers and experts and, at the same time, precise enough for normative analysis. An influence diagram is a graphical representation of uncertain quantities and decisions that explicitly reveals probabilistic dependence and the flow of information. In recent years, it has become an established tool for developing models and communicating among people.

An influence diagram is a network with directed arcs and no cycles. The nodes represent random variables and decisions. Arcs into random variables indicate probabilistic dependence, while arcs into decisions specify the information available at the time of the decision. The diagram is compact and intuitive, emphasizing the relationships among variables, and yet it represents a complete probabilistic description of the problem. For example, it is easy to convert any decision tree into an influence diagram. Conversely, it is possible to transform any well-formed influence diagram into a decision tree, though doing so may require repeated applications of Bayes' theorem.

From their inception, influence diagrams were conceived as a "front end" for a decision analysis computer system (Miller, Merkhofer, Howard, Matheson and Rice 1976). There have been several attempts to automate the transition from influence diagram formulation to analysis (Merkhofer 1981, Korsan and Matheson 1978, Howard and Matheson 1981). Olmsted (1983) proposed evaluating a decision problem within the influence diagram representation. This paper completes that process with an algorithm that can evaluate any well-formed influence diagram directly.

There are several benefits to evaluating a problem through influence diagram operations. Since the algorithm performs all of the inference and analysis automatically, the analyst is able to use a representation that is natural to the decision maker. For example, the algorithm makes it easy to formulate sensitivity questions, such as the value of information or control, and to evaluate them. Since the model is stored in its original formulation, the information from the sensitivity analysis can be used to modify and refine the formulation directly. This capability is convenient not only when the analyst is dealing with a decision maker, but simplifies the construction of an automated decision system.

The influence diagram solution procedure can also

Subject classification: 91 knowledge representation and evaluation algorithm, 113 automated probabilistic inference, 481 network structure to represent and solve decision problems.

0030-364X/86/3406-0871 $01.25

result in significant gains in efficiency. Conditional independence is explicitly revealed in the diagram, so the algorithm can take advantage of it. This can reduce the size of intermediate calculations and result in considerable reductions in processing time and memory requirements. In fact, decision tree algorithms that are "smart" enough to keep track of conditional independence are really building influence diagrams. On the other hand, there are some problems for which decision trees may be more efficient. An influence diagram corresponds to a symmetric decision tree, so if most of the computational savings can be achieved through asymmetric processing, the influence diagram algorithm, as currently conceived, cannot take advantage of those savings.

Section 1 introduces influence diagrams and shows some formulation examples. Section 2 develops a formal definition of an influence diagram, along with sufficient conditions for it to be well-formed. Section 3 shows the basic operations used in transforming the influence diagram. In Section 4, these operations are combined into an algorithm that can evaluate any well-formed influence diagram. Section 5 states the conclusions and some directions for future research.

1. Formulation Examples

This section contains a brief description of influence diagrams and how they can be used to formulate decision problems. For more information on this subject, see Howard and Matheson.

An influence diagram is a network with three types of nodes: chance nodes, decision nodes and a value node, drawn respectively as circles, squares and a rounded rectangle. There are two types of directed arcs: conditional arcs (into chance and value nodes) and informational arcs (into decision nodes).

First, consider a diagram containing only chance nodes. Associated with each node is a random variable, and there is an underlying joint probability distribution for all of the random variables. This joint distribution can be decomposed into a set of conditional distributions, to be assessed by the analyst, with conditioning represented by arcs in the diagram. If there is no undirected path between two nodes, then they must be independent. If arc (i, j) is part of the diagram, then the assessed distribution for the jth random variable is conditioned on the value of the ith. When a chance node has no arcs into it, then the assessed distribution is a marginal (unconditional) distribution.

Consider the case of two chance nodes, shown in Figure 1. Either they are independent and have no arcs between them, or there is an arc from one to the other. When they are dependent, either one can have a marginal distribution, and the other a conditional distribution. The direction of the arc can be reversed through inference, by invoking Bayes' theorem. Note that a conditional arc represents probabilistic dependence and not (necessarily) causality, so the meaning of the diagram (the underlying joint distribution) is the same, no matter which direction the arc points.

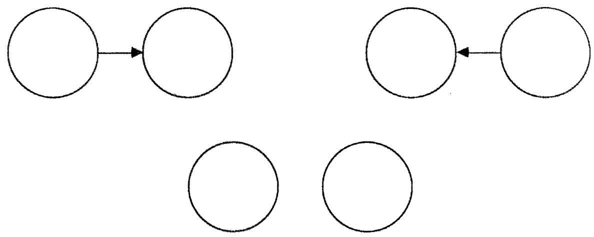

Figure 1. All possibilities with two chance nodes.

Figure 2 illustrates the different cases for three nodes. Cases (a) and (b) show total and partial independence, and case (c) shows complete dependence among the associated random variables. Now, however, there is also the possibility of conditional independence, shown in case (d). The outer two random variables are dependent, but only through their direct dependence on the middle variable. If the value of the middle variable is known, then the other two random variables are independent. Note that a cycle is not

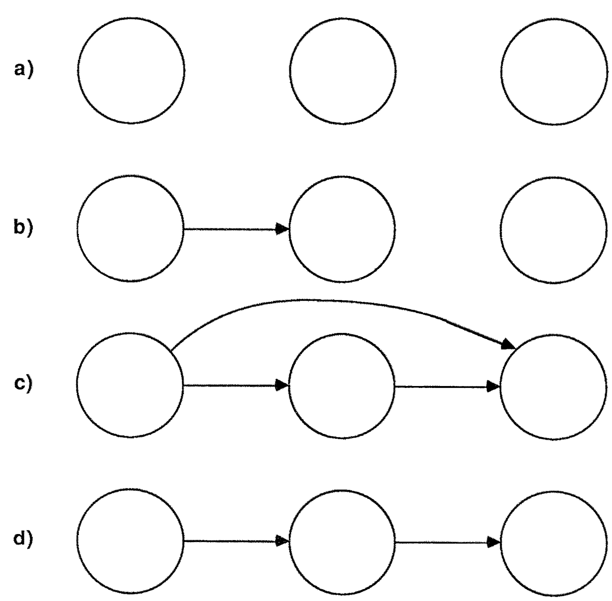

Figure 2. Some of the possibilities with three chance nodes.

permitted. In that case, it would be impossible to compute the underlying joint distribution.

As another example, consider a discrete-time Markov chain and let each chance node correspond to the state variable at a given time. (Note that this is not the usual network representation of a Markov chain.) The Markov property, that the future is independent of the past, given the present, is represented by a series network (Figure 3). Since each arc may be reversed, using Bayes' theorem, from left to right in the diagram, it is easy to see that the reverse chain must always satisfy the Markov property as well.

In order to evaluate the influence diagram, there must be some question to be answered, some random variable(s) whose distribution must be determined. The associated chance node is singled out as the value node, and an influence diagram containing a value node is said to be oriented. In this paper we will assume that there is only a single random variable associated with the value node, that it needs only to be calculated in expectation, and that it represents the expected utility of the outcome. (In general, we could maintain more information than just an expected value—additional moments, a full lottery, or even a vector of lotteries.) If there are decisions to be made, then the expected utility will be used to compare alternatives. The variables associated with nodes having arcs into the value node are the attributes of the decision maker's utility function.

There may also be decision nodes in the diagram. Each node represents a choice among a set of alternatives. Arcs into decision nodes indicate time precedence, that the information at the source of the arc is available at the time the decision is made. Given the state of information at the time of the decision, the alternative(s) to be selected should maximize the expected utility of the resulting outcome, normally associated with the value node.

Figure 4 shows several cases involving one node of each type, a chance, a value, and a decision node. In case (a), the value depends on the random variable,

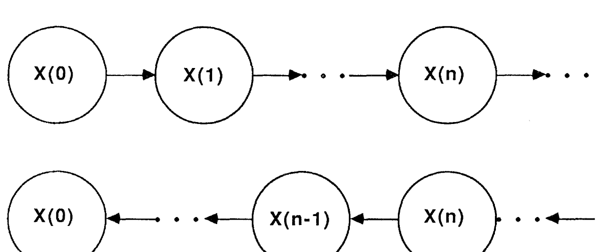

Figure 3. Discrete-time Markov chain with forward and backward transitions.

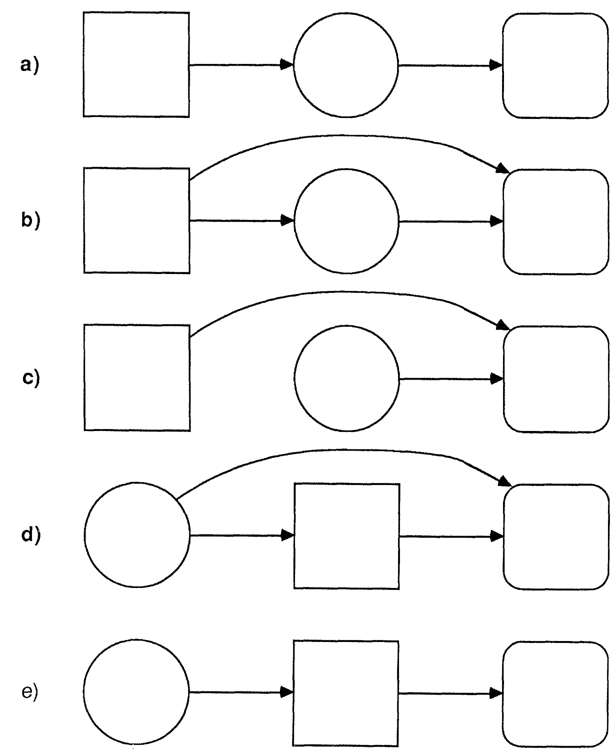

Figure 4. Some possibilities with a chance, decision and value node.

which itself depends upon a decision. Case (b) is similar, but the value depends on both the decision and the random variable. In case (c), the value depends on both, but the random variable is independent of the decision. In case (d), the random variable is observed before the decision is made. This case may be thought of as the "closed loop" version of case (c), and must yield at least as much expected utility, since the decision maker is better informed at the time of the decision. Finally, in case (e), the random variable is observed but has no effect on the value and may be considered irrelevant with respect to the decision.

Cycles are still not permitted. A cycle involving a decision and a random variable would violate the decision maker's free will—it would imply that he can infer something about a decision he has not yet made. On the other hand, if the cycle contains only decision nodes, then it contradicts the assumption of time precedence.

It is not possible, however, to reverse an arc into or out of a decision node. Because an arc into a decision node represents time precedence, reversal would change the meaning of the diagram.

There is also an asymmetry in the definition of the arcs. A conditional arc (into a chance or value node) indicates that there *may* be dependence. An informational arc (into a decision node) indicates that the information *must* be available at the time of the decision. The strong statements are the absence of conditional arcs (independence) and the presence of informational arcs (the acquisition of information).

As a final example, consider the oil wildcatter problem in Raiffa (1968), shown in Figure 5a. This influence diagram is solved in Section 4 as an illustration of the algorithm. The value in this problem is the decision maker's profit, and it is a function of the decision to drill, the cost of drilling, the amount of oil to be found, and the type of test performed. At the time the wildcatter must choose whether to drill, he knows the results of the test he ordered. That test is a (possibly noisy) determination of the seismic structure or no test, so the results depend both on the testing decision and the seismic structure. The seismic structure itself is dependent on the possible presence of oil. Finally, there is no observation available before the testing decision must be made. (In this problem, the drilling decision is whether to drill, and the cost of drilling does not depend upon that decision. If there were different drilling alternatives available, then the cost of drilling could depend upon the drilling decision.)

A variation on the influence diagram includes a utility function over the wildcatter's profits, as is shown in Figure 5b. If the wildcatter's preferences are better described by a multiattribute utility function, the diagram could represent this problem feature as well, by drawing arcs to the value node from the nodes corresponding to those attributes.

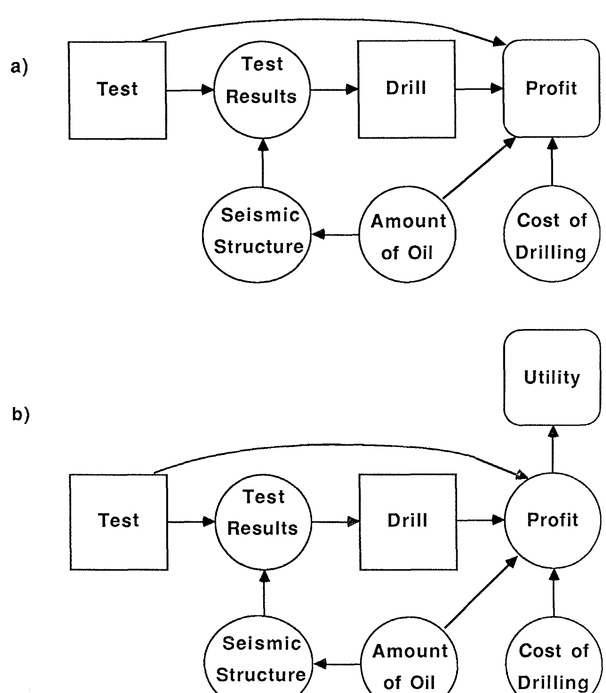

Figure 5. Oil wildcatter's problem with expected monetary value and with utility.

2. Formal Definition of an Influence Diagram

This section contains a formal definition of an influence diagram, and develops the notation needed to prove and explain the influence diagram solution algorithms. Before an influence diagram can be defined, however, a number of concepts and objects must be introduced.

An influence diagram is a network consisting of a directed graph $G = (N, A)$ and associated node sets and functions. It contains three types of nodes in the set N, partitioned into sets V, C and D. There is at most one *value* node $v \in V$, drawn as a rounded rectangle, which represents the objective to be maximized in expectation. There are zero or more *chance* nodes in the set C, shown as circles, representing random variables (or uncertain quantities). Finally, there may be zero or more *decision* nodes in the set D, drawn as squares, corresponding to choices available to the decision maker.

The arcs A in the graph have different meanings, based on the target. Arcs into utility and chance nodes are *conditional* and represent probabilistic dependence. They do not imply causality or time precedence. Arcs into decision nodes are *informational* and imply time precedence. Any uncertainties or decisions at the tails of such arcs have been resolved before the decision at the head of the arc must be made.

In describing the influence diagram evaluation algorithm, it is more convenient to think in terms of the predecessors and successors of a node in the graph rather than the arc set A. The set of *direct successors*, $S(i)$, of node i is defined as

$$S(i) = \{j \in N: (i, j) \in A\},$$

while the *indirect successors* (or simply *successors*) is the set of nodes along directed paths emanating from node i. In a similar fashion, the set of *direct predecessors* is $\{j \in N: (j, i) \in A\}$ and the *indirect predecessors* (or simply *predecessors*) is the set of nodes along directed paths into node i. It is useful to distinguish between the two kinds of direct predecessors. Direct predecessors of chance or value node i are called *conditional predecessors*, and denoted by $C(i)$, and the direct predecessors of decision node i are the *informational predecessors*, $I(i)$.

Associated with each node i in the graph is a variable

X_i in the decision maker's problem, and a set Ω_i of possible values it may assume. If i is the value node, then X_i represents the expected utility and its domain Ω_i is a subset of the real line. If i is a chance node, then Ω_i is the sample space for the random variable X_i. Finally, decision node i has alternative X_i chosen from the set Ω_i. For ease of presentation, whenever convenient we assume all sets Ω_i are finite. The notation Ω_J, with $J = \{j1, \ldots, jn\} \subset$ N, refers to the cross-product space $\Omega_{j1} \times \ldots \times \Omega_{jn}$. Likewise, X_J denotes the random vector $(X_{j1}, \ldots, X_{jn})$.

Each node i in the influence diagram has an associated mapping. For chance and value nodes, this mapping is an input that must be assessed before evaluation can begin. As it transforms the diagram, the algorithm redefines these mappings. For decision nodes, the algorithm calculates the mapping and represents it as an output from the process.

The value node $v \in$ V has an associated utility function U: $\Omega_{C(v)} \to \Omega_v$, which represents the expected utility as a function of the values of the conditioning predecessors of the value node. At the conclusion of the algorithm, the value node has no predecessors, $C(v) = \emptyset$, and U( ) evaluates to the maximal expected utility.

There is a conditional probability distribution π_i for every chance node i, given the values of its conditional predecessors, $\pi_i(x_i \mid x_{C(i)}) = \Pr\{X_i = x_i \mid X_{C(i)} = x_{C(i)}\}$. If the node i has no predecessors, then π_i represents the marginal distribution for X_i,

$$\pi_i(x_i) = \Pr\{X_i = x_i\}.$$

For each decision node i, there is an optimal policy d_i^* computed during the algorithm. It maps from $\Omega_{I(i)}$ into Ω_i, indicating the optimal alternatives given the decision maker's state of information at the time of the decision.

It is now possible to formally define an influence diagram.

An *influence diagram* consists of a directed graph G = (N, A) with nodes N and arcs A. The nodes are partitioned into sets V, C and D. For each node i, there is a set Ω_i and a mapping, either U, π_i or d_i^*, depending upon the node type.

An influence diagram is said to be *proper* if it is an unambiguous representation of a single decision maker's view of the world. It is said to be *oriented* if it contains a value node.

Slightly stronger conditions must be assumed in the development of an algorithm to evaluate influence diagrams. The algorithm will then be a constructive proof that these conditions are sufficient for an influence diagram to be proper. An influence diagram is said to be *regular* if it satisfies the following conditions:

(1) the directed graph has no cycles,
(2) the value node, if present, has no successors, and
(3) there is a directed path that contains all of the decision nodes.

The third condition is equivalent to requiring a total ordering of all of the decisions, a reasonable condition when there is a single decision maker. As a result, any relevant information available at the time of one decision should be available for all subsequent decisions. This "no forgetting" property could be enforced as a condition for regularity, but as a convenience to the modeler, the algorithm can assume it with no ambiguity. This way, at most one arc to a decision node needs to be specified from any node.

Proposition 1. No Forgetting. *If decision node i precedes decision node j in a regular influence diagram, then node i and all of its informational predecessors should be informational predecessors of node j.*

Figure 6 shows the addition of "no forgetting" arcs.

It is easy to determine whether an influence diagram is regular by invoking the following well-known result from a graph theory (Lawler 1976).

Proposition 2. *A directed graph has no cycles if and only if some list of the nodes has all of the successors of a node follow it in the list.*

An algorithm to build such a list and check for a

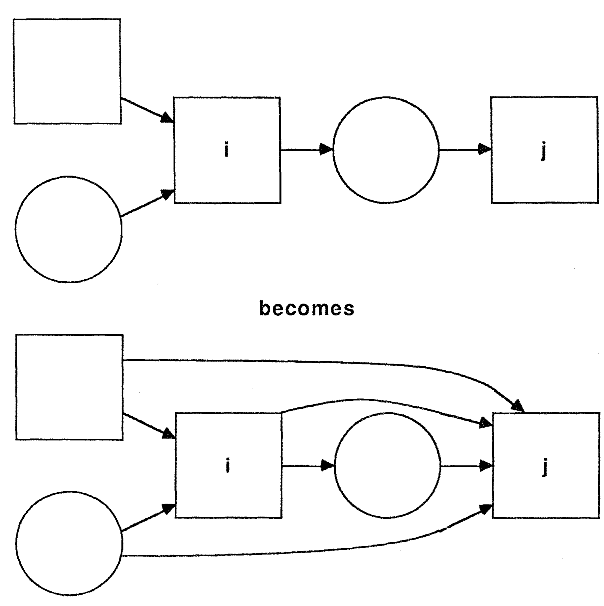

Figure 6. Adding "no forgetting" arcs.

cycle is given by *order* (N):

```
DEFINE PROCEDURE order (M) TO BE
  BEGIN
    W ← {i ∈ M | S(i) ∩ M = ∅}
    IF W ≠ ∅
      THEN RETURN (APPEND (order(M\W),
        W))
    ELSE IF M = ∅
        THEN RETURN(W)
        ELSE ERROR ("Cycle in graph")
  END
```

Ordering the decision nodes in the graph simplifies the propagation of "no forgetting" arcs.

3. Transformations to the Influence Diagram

The procedure to evaluate an influence diagram consists of a sequence of transformations to the diagram that maintain feasibility and do not modify the optimal policy or maximal expected value. Such a transformation will be called a *value-preserving reduction.* A node will be said to be *removed* from the diagram if it is eliminated through some value-preserving reduction. When a node is removed, it can be dropped from the current node set N, and all arcs incident to it can be dropped from the current arc set A.

This section develops two kinds of reductions—those that remove nodes and one that reverses arcs. The chance and decision node removal transformations are based on results by Olmsted, but are really just the basic steps in evaluating a stochastic dynamic program (Bellman 1957). The arc reversal transformation is Bayes' Theorem (Howard and Matheson, and Olmsted).

A chance or decision node will be called a *barren node* if it is a sink, that is, it has no successors. No matter what value is assigned to the barren node variable, no other node is affected, so it may be removed from the diagram. The algorithm does not need to know a probability distribution for barren chance nodes, and so it may be possible to evaluate an incomplete influence diagram if all of the chance nodes missing probability distributions become barren nodes. Figure 7 illustrates the removal of barren nodes.

Proposition 3. Barren Node Removal. *A barren node may be simply removed from an oriented, regular influence diagram. If it is a decision node, then any alternative would be optimal.*

Once a barren node has been removed, other nodes may become barren, as shown in Figure 7. In general,

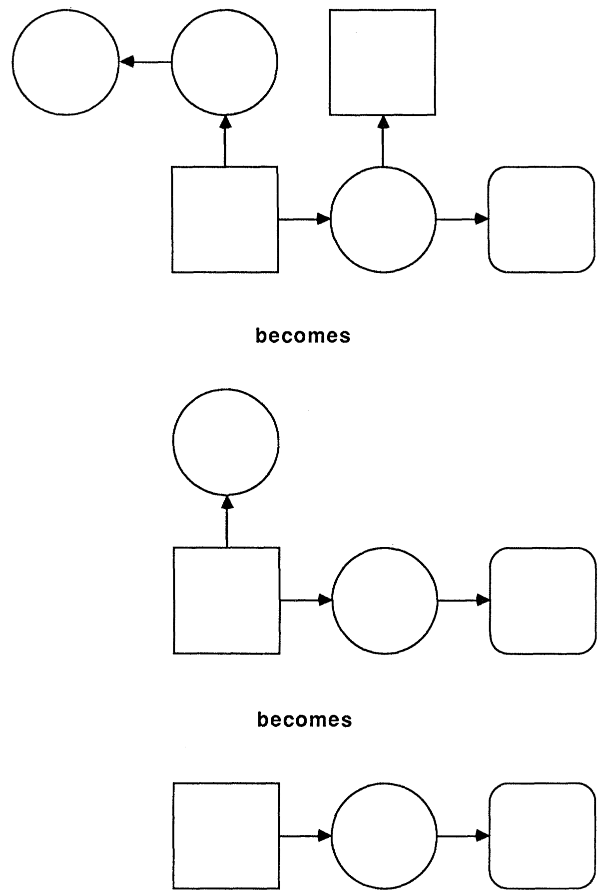

Figure 7. Reducing barren nodes.

any chance or decision node that does not indirectly precede the value node can be considered a barren node. Such a node may be simply removed from an oriented, regular influence diagram.

Theorem 1. Chance Node Removal. *Given that chance node i directly precedes the value node and nothing else in an oriented, regular influence diagram, node i may be removed by conditional expectation. Afterward, the value node inherits all of the conditional predecessors from node i, and thus the process creates no new barren nodes.*

Proof. Figure 8 gives a picture of the process. The conditional predecessors of the value node after the removal of node *i* become

$$C^{new}(v) \leftarrow C^{old}(v) \cup C(i)\backslash\{i\}.$$

[The operator "\" denotes set subtraction. $A\backslash B = \{i \in A \mid i \notin B\}$.] Since the conditional expectation with respect to X_i affects only the value node,

$$
\begin{aligned}
U^{new}(x_{C^{new}(v)}) &\leftarrow E\{X_v \mid X_{C^{new}(v)} = x_{C^{new}(v)}\} \\
&= E\{E\{X_v \mid X_i\} \mid X_{C^{new}(v)} = x_{C^{new}(v)}\} \\
&= \sum_{x_i \in \Omega_i} E\{X_v \mid X_i = x_i, X_{C^{new}(v)} = x_{C^{new}(v)}\} \\
&\qquad \cdot \Pr\{X_i = x_i \mid X_{C^{new}(v)} = x_{C^{new}(v)}\} \\
&= \sum_{x_i \in \Omega_i} E\{X_v \mid X_{C^{old}(v)} = x_{C^{old}(v)}\} \\
&\qquad \cdot \Pr\{X_i = x_i \mid X_{C(i)} = x_{C(i)}\} \\
&= \sum_{x_i \in \Omega_i} U(x_{C^{old}(v)})\pi_i(x_i \mid x_{C(i)})
\end{aligned}
$$

for all $x_{C^{new}(v)} \in \Omega_{C^{new}(v)}$.

Decision nodes may be removed from the diagram by maximizing the expected utility. First, however, the procedure must use conditional expectation to remove any conditional predecessors of the value node not observable at the time of the decision. Therefore, decision node i may be removed only when all direct predecessors of the value node (except for i itself) directly precede node i. Figure 9 shows the decision node removal process.

Theorem 2. Decision Node Removal. *Given that all barren nodes have been removed, that decision node i is a conditional predecessor of the value node, and that all other conditional predecessors of the value node are informational predecessors of node i in an oriented,*

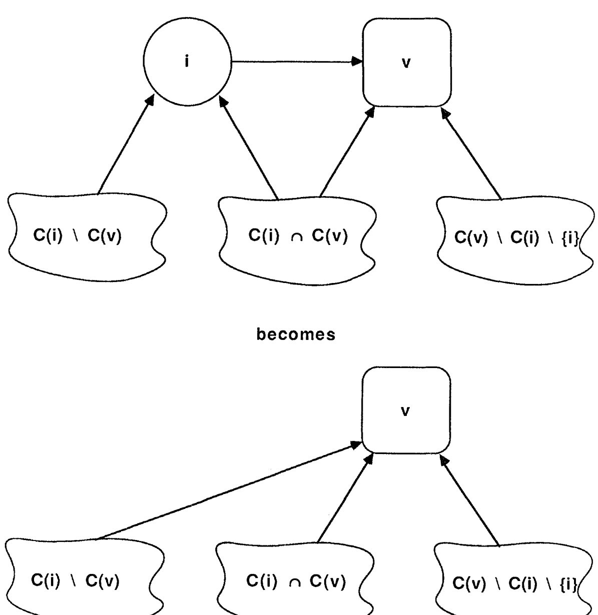

Figure 8. Removing a chance node.

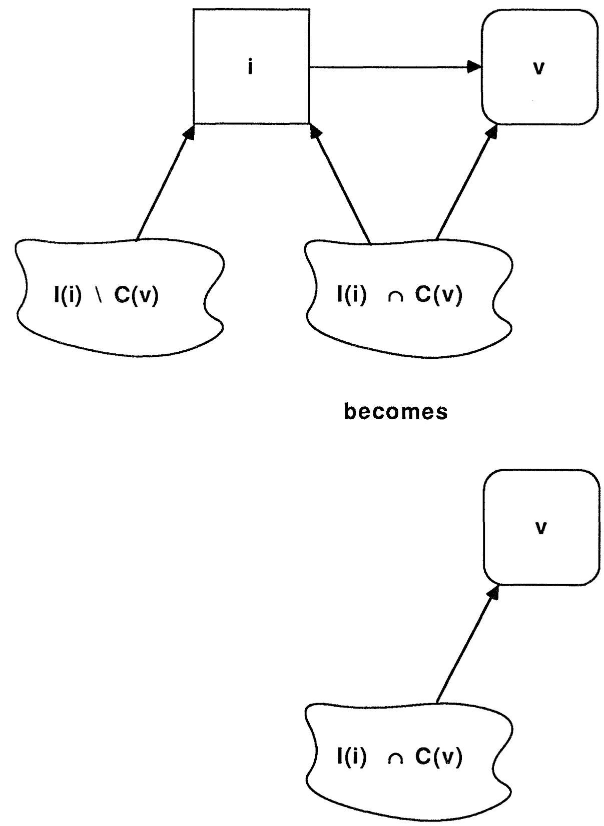

Figure 9. Reducing a decision node.

regular influence diagram, node i may be removed by maximizing expected utility, conditioned on the values of its informational predecessors. The maximizing alternative(s) should be recorded as the optimal policy. The value node inherits no new conditional predecessors from this operation. As a result, it is possible that some of the informational predecessors of node i may become barren nodes.

Proof. First, we must show that the value node is the only successor of node i. Suppose that this is not the case, that is, some node $j \neq v$ is also a successor of i. Since the diagram contains no barren nodes, there must be a directed path from j to v. Let k (possibly j) be the penultimate node on that path. By construction, k is a successor of i and a conditional predecessor of v. The assumptions in the theorem require that k must also be an informational predecessor of i, which means there is a cycle and hence a contradiction.

The conditional predecessors of the value node after the removal of node i become

$$C^{new}(v) \leftarrow C^{old}(v) \setminus \{i\}.$$

Note that the variables $X_{I(i) \setminus C(v)}$, although known at the time of the decision, are irrelevant and do not play a role.

The assumptions ensure that the expected utility depends only on variables know when the decision corresponding to node i is made. Therefore the optimal expected utility is given by

$$U^{\text{new}}(x_{C^{\text{new}}(v)}) \leftarrow \max_{x_i \in \Omega_i} E(X_v \mid X_i = x_i, X_{I(i)} = x_{I(i)}\}$$

$$= \max_{x_i \in \Omega_i} E\{X_v \mid X_{C^{\text{old}}(v)} = x_{C^{\text{old}}(v)}\}$$

$$= \max_{x_i \in \Omega_i} U(x_i, x_{C^{\text{new}}(v)})$$

and the optimal policy is determined by

$$d_i^*(x_{C^{\text{new}}(v)}) \leftarrow \arg\max_{x_i \in \Omega_i} U(x_i, x_{C^{\text{new}}(v)})$$

for all $x_{C^{\text{new}}(v)} \in \Omega_{C^{\text{new}}(v)}$.

It may be worthwhile to keep track of ties and to let d_i^* be the set of all optimal policies. It is also useful to store with d_i^* the expected utility value that it achieves.

The final transformation is the reversal of arcs between chance nodes, and is the implementation of Bayes' Theorem. Remember that it is not possible to reverse an arc incident on a decision node. Figure 10 illustrates the arc reversal process.

Theorem 3. Arc Reversal. *Given that there is an arc (i, j) between chance nodes i and j, but no other directed (i, j)-path in a regular influence diagram, arc (i, j) can be replaced by arc (j, i). Afterward, both nodes inherit each other's conditional predecessors.*

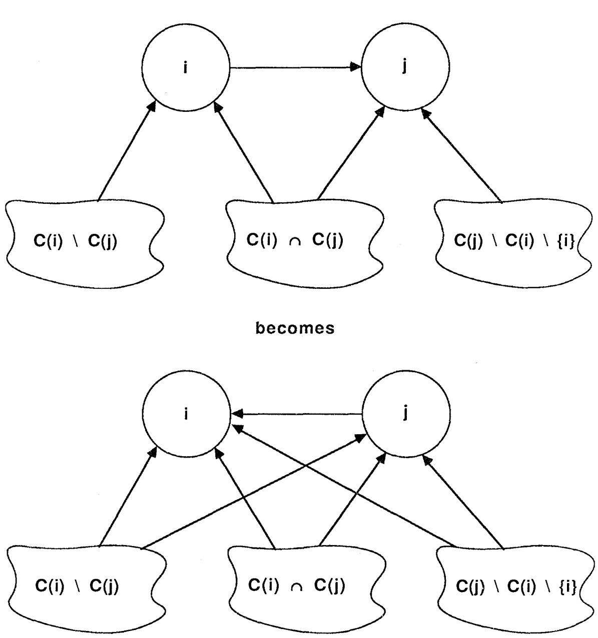

Figure 10. Reversing an arc between chance nodes.

Proof. The conditional predecessors of each chance node after arc reversal become

$$C^{\text{new}}(j) \leftarrow C^{\text{old}}(i) \cup C^{\text{old}}(j) \backslash \{i\}$$

and

$$C^{\text{new}}(i) \leftarrow C^{\text{new}}(j) \cup \{j\}.$$

In addition to the new arc (j, i) which replaces arc (i, j), the reversal adds arcs from the conditional predecessors of each node to the other node, if not already present, bringing both to the same state of information before the arc reversal. The requirement that there is no other directed (i, j)-path is necessary and sufficient to ensure that no cycle is created by the addition of these arcs.

By conditional expectation,

$$\pi_j^{\text{new}}(x_j \mid x_{C^{\text{new}}(j)}) \leftarrow \Pr\{X_j = x_j \mid X_{C^{\text{new}}(j)} = x_{C^{\text{new}}(j)}\}$$

$$= E\{\Pr\{X_j = x_j \mid X_i\} \mid X_{C^{\text{new}}(j)} = x_{C^{\text{new}}(j)}\}$$

$$= \sum_{x_i \in \Omega_i} \Pr\{X_j = x_j \mid X_i = x_i, X_{C^{\text{new}}(j)} = x_{C^{\text{new}}(j)}\} \cdot \Pr\{X_i = x_i \mid X_{C^{\text{new}}(j)} = x_{C^{\text{new}}(j)}\}$$

$$= \sum_{x_i \in \Omega_i} \Pr\{X_j = x_j \mid X_{C^{\text{old}}(j)} = x_{C^{\text{old}}(j)}\} \cdot \Pr\{X_i = x_i \mid X_{C^{\text{old}}(i)} = x_{C^{\text{old}}(i)}\}$$

$$= \sum_{x_i \in \Omega_i} \pi_j^{\text{old}}(x_j \mid x_{C^{\text{old}}(j)}) \pi_i^{\text{old}}(x_i \mid x_{C^{\text{old}}(i)}),$$

and, by Bayes' theorem,

$$\pi_i^{\text{new}}(x_i \mid x_{C^{\text{new}}(i)}) \leftarrow \Pr\{X_i = x_i \mid X_{C^{\text{new}}(i)} = x_{C^{\text{new}}(i)}\}$$

$$= \frac{\Pr\{X_j = x_j, X_i = x_i \mid X_{C^{\text{new}}(j)} = x_{C^{\text{new}}(j)}\}}{P\{X_j = x_j \mid X_{C^{\text{new}}(j)} = x_{C^{\text{new}}(j)}\}}$$

$$= \Pr\{X_j = x_j \mid X_i = x_i, X_{C^{\text{new}}(j)} = x_{C^{\text{new}}(j)}\} \cdot \frac{\Pr\{X_i = x_i \mid X_{C^{\text{new}}(j)} = x_{C^{\text{new}}(j)}\}}{\Pr\{X_j = x_j \mid X_{C^{\text{new}}(j)} = x_{C^{\text{new}}(j)}\}}$$

$$= \Pr\{X_j = x_j \mid X_{C^{\text{old}}(j)} = x_{C^{\text{old}}(j)}\} \cdot \frac{\Pr\{X_i = x_i \mid X_{C^{\text{old}}(i)} = x_{C^{\text{old}}(i)}\}}{\Pr\{X_j = x_j \mid X_{C^{\text{new}}(j)} = x_{C^{\text{new}}(j)}\}}$$

$$= \frac{\pi_j^{\text{old}}(x_j \mid x_{C^{\text{old}}(j)}) \pi_i^{\text{old}}(x_i \mid x_{C^{\text{old}}(i)})}{\pi_j^{\text{new}}(x_j \mid x_{C^{\text{new}}(j)})}$$

for all $x_i \in \Omega_i$, $x_j \in \Omega_j$, $x_{C^{\text{new}}(j)} \in \Omega_{C^{\text{new}}(j)}$.

4. The Algorithm

This section combines the basic transformations developed so far into a procedure that can evaluate any oriented, regular influence diagram. The procedure will remove nodes from the diagram until only the value node remains. At that point, it has determined all of the optimal policies and computed the maximal expected utility. The following theorem justifies the crucial step in the procedure.

Theorem 4. Existence of a Node to Remove. *Given that an influence diagram is oriented and regular, has no barren nodes, and has "no forgetting" arcs added, if the value node has predecessor(s) but there is no decision node that may be removed, then there is a chance node that is a conditional predecessor of the value node but not an informational predecessor of any decision node. That chance node may be removed, perhaps after some arc reversals.*

Proof. First, suppose the diagram contains no decision nodes. In that case all of the conditional predecessors of the value node are chance nodes with no decision successors, and the result follows.

Hereafter, suppose the diagram contains at least one decision node. The diagram is regular, so the decisions are completely ordered, and there is some decision node j corresponding to the latest decision. Because all "no forgetting" arcs have been added, any information predecessor of any decision node must also directly precede node j. Therefore, we are looking for some node $i \in C \cap C(v) \backslash I(j)$.

Suppose $j \in C(v)$. Node j cannot be removed so $C(v) \backslash (I(j) \cup \{j\})$ is not empty. Due to the "no forgetting" arcs, $D \subset I(j) \cup \{j\}$, so there is some node $i \in C \cap C(v) \backslash I(j)$.

Otherwise $j \notin C(v)$. Since the diagram contains no barren nodes, it must contain a directed path from j to v. Let node i be the penultimate node on the path. By the construction, $i \in C \cap C(v)$ and, because there cannot be a cycle, $i \notin I(j)$. Thus $i \in C \cap C(v) \backslash I(j)$.

During its execution, the algorithm may select a chance node $i \in C(v)$ for removal which has other (chance node) successors besides the value node. In this case, its arcs to chance nodes must be reversed before it can be removed from the diagram. Care must be taken when performing these reversals so that a cycle will not be formed. For example, if j and k are chance node successors of i, and j is a (possibly indirect) predecessor of k, then arc (i, j) must be reversed before arc (i, k). (Otherwise there is another directed (i, k)-path, through j, which violates the arc reversal conditions.) Fortunately, due to the acyclicity of the graph, some ordering of reversals does not create a cycle.

We can now state an algorithm that can evaluate any regular influence diagram:

```
DEFINE PROCEDURE IDEVAL TO BE
  BEGIN
    check for oriented, regular influence diagram
    and "no forgetting" arcs
    eliminate all barren nodes
    WHILE C(v) ≠ ∅ DO
      BEGIN
        IF there exists i ∈ C ∩ C(v) st S(i) = {v}
        THEN remove chance node i
        ELSE IF there exists i ∈ D ∩ C(v)
             st C(v) ⊂ I(i) ∪ {i}
          THEN BEGIN
               remove decision node i
               eliminate barren nodes
            END
          ELSE BEGIN
               find i ∈ C ∩ C(v) st D ∩ S(i) = ∅
               WHILE C ∩ S(i) ≠ ∅ DO
                 BEGIN
                   find j ∈ C ∩ S(i) st there is no
                        other directed (i, j)-path
                   reverse arc (i, j)
                 END
               remove chance node i
            END
      END
  END
```

These value-preserving reductions can be used to evaluate any regular influence diagram. Every step of the algorithm removes at least one node, so the algorithm will always terminate. Since the decision maker's problem can be evaluated uniquely, the influence diagram must be proper.

Corollary 1. *If an influence diagram is regular, then it is also proper.*

As an example of the algorithm, consider again the oil wildcatter problem from Raiffa as formulated in Section 1, and shown in Figure 11a. This influence diagram is feasible and contains no barren nodes, but has one added "no forgetting" arc, since the type of test chosen is known at the time of the drilling decision.

In its first iteration, the algorithm will remove the "Cost of Drilling" chance node since the value node is its only successor.

In the next iteration, no node can be removed

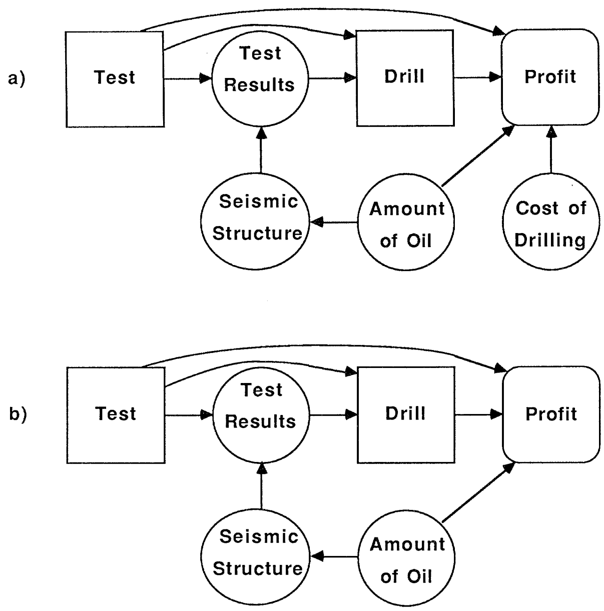

Figure 11. Evaluating oil wildcatter's problem (first part).

directly. The "Amount of Oil" chance node is a direct predecessor of the value node, but it has another successor as well. The "Drill" and "Test" decision nodes are also direct predecessors of the value node but the "Amount of Oil" is not known at the time these decisions must be made. Therefore, the algorithm must reverse the arc from "Amount of Oil" to "Seismic Structure" as shown in Figure 11b. This operation is the common "flipping" of the decision tree, familiar to everyone who has solved these problems by hand. Now it is possible to remove the "Amount of Oil" chance node (Figure 12a). Notice that "Seismic Structure" becomes a direct predecessor of the value node in the process.

The third iteration is similar to the second in that it is impossible to remove any of the nodes directly. Reversing the arc from "Seismic Structure" to "Test Results," however, removes the "Seismic Structure" and as shown in the new diagram in Figure 12b. Note that when the arc is reversed, "Test" becomes a direct predecessor of "Seismic Structure." When "Seismic Structure" is removed, "Profit" inherits conditional predecessor "Test Results."

In the fourth iteration, we are finally able to remove the "Drill" decision node, since the other conditional predecessors of the value node are informational predecessors of "Drill." Figure 12c illustrates the new diagram. In the process of removing "Drill," a table is constructed showing, for each type of test chosen and the results of that test, which drilling decision should be made.

The fifth iteration removes the "Test Results" chance node and the new diagram has just two nodes (Figure 12d).

The sixth and last iteration removes the "Test" decision node. By comparing the value of each alternative, we can now select the optimal testing decision:

(1) No test should be done and always drill. This alternative has an expected value of $40,000.
(2) If the seismic test were conducted, then drill unless there is no structure. The test is not conducted, because it is worth $4,300 but it costs $10,000.
(3) No matter what the experimental test reveals, drill anyway. Therefore, this test is worthless to us.

5. Conclusions

The influence diagram has become a useful tool for analysts in communicating with decision makers and experts. The development of an algorithm to evaluate such diagrams directly creates new possibilities in decision analysis to construct intelligent systems using influence diagrams as a knowledge representation. The convenience it allows in posing sensitivity questions and the efficiency in solution from explicit

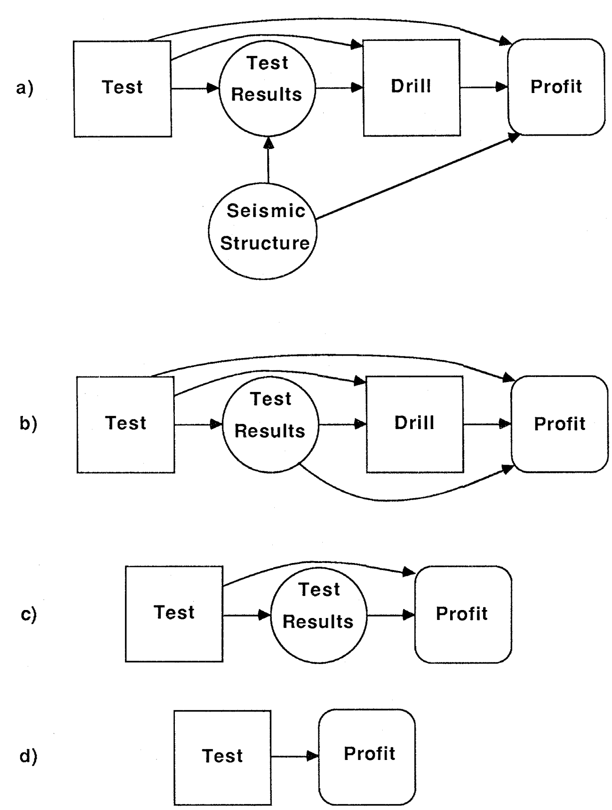

Figure 12. Evaluating oil wildcatter's problem (second part).

recognition of conditional independence offer clear advantages over decision tree processing for many applications. Moreover, the analyst is able to frame the problem from the perspective of the decision maker, and to maintain and revise the model from that perspective.

The algorithm allows the evaluation phase to become transparent, so that analysts can place emphasis on formulating the model and asking sensitivity questions. Because the model can be represented in the same form in which it was assessed, it is natural for the decision makers and experts to understand the process and to be involved. Even when there are no decisions in the model, such as in a complex reliability analysis, the focus is on the relationships among the variables and the most natural way to obtain probabilities. Probabilistic inference on the model can then be performed by the algorithm.

For example, influence diagrams make it easy to compute the expected value of information, since the required modification is the addition or deletion of an informational arc from chance node(s) to decision node(s). (Note that this assessment is actually the selling price for information rather than the desired buying price, but they are the same when our utility function is linear or exponential.) When we evaluate value of information on an influence diagram, we must explicitly specify *which* information and *when* it will become available (Merkhofer 1977).

Consider the oil wildcatter problem and the value of knowing the amount of oil before making the drilling decision. In Figure 13a an arc is added from "Amount of Oil" to "Drill." The expected value of the new diagram is $65,000, and it would never be worth more than $25,000 (i.e., $65,000 − $40,000) to obtain any information about the presence of oil.

Suppose we could find out the cost of drilling before making a testing decision. In this case, shown in Figure 13b, if the cost is highest ($70,000 in the problem), then the experimental test should be ordered and drilling conducted unless it reveals no structure. (Otherwise, there is drilling without any test.) The value of learning the cost in advance is $385.

In general, the decision maker may specify a collection of informational arcs to add or delete simultaneously, and may determine their collective value.

It is prohibitive to try to determine the value of all present and potential informational arcs. It is reasonable, however, to consider changing the time that a particular random variable is known. We can do so by adding or deleting arcs from the corresponding chance node to different decision nodes. Since this operation will add "no forgetting" arcs, there never needs to be more than one arc into a decision node—

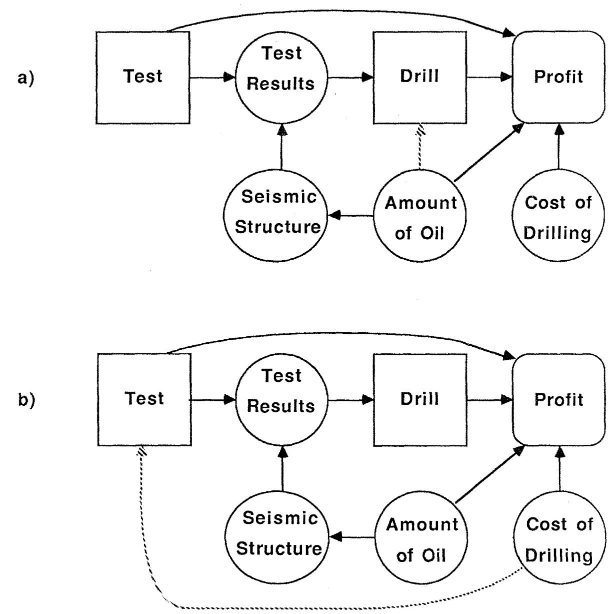

Figure 13. Value of information in oil wildcatter's problem.

the earliest decision by which the information will be available. There is no limit to how late the information may be revealed since it is possible that it may never be observed. On the other hand, there can be a limit on how early it may be known, if the chance node is a successor of some decision node. Within these limits, it is possible to obtain the value of finding out the information earlier or later than in the original problem formulation.

It is also possible to change the time that a decision is to be made, or, in general, to change the order of decisions (assuming that the sets of alternatives do not change). There is no limit to how early a particular decision can be made, since if it is made before anything is observed then it is just an "open loop" decision. There is, however, a limit on how late a decision may be made, since some chance node may be the successor of this decision and the predecessor of another. Within these limits, one can find the value of postponing or hurrying a decision as compared to the original formulation. These changes are implemented by the addition and deletion of informational arcs into and out of the particular decision node.

In a similar fashion, we could consider other sensitivity questions. The value of control is the improvement we obtain when a chance node becomes a decision node. Conversely, policies may be compared by replacing a decision node with a (deterministic) chance node. Stochastic sensitivity is performed by reducing the diagram to our key variables as conditional predecessors of the value node. Of course all of these options may be combined along with the most basic sensitivities, the perturbation of numbers and the inclusion or exclusion of outcomes or alternatives.

In addition, if the algorithm is to be run many times with a variety of changes to the diagram, it may be possible to reuse partial solutions. As an example, a decision tree is one such partial solution, and the algorithm can be easily modified to just perform the inferences needed to build a decision tree from an influence diagram.

There are many directions for future research. These include refinement of the algorithm and its implementation, application to specific problem structures, possible uses as a subproblem within a larger decision system, and extensions to the influence diagram "language."

There are challenges to implement the algorithm in a framework that extends its power and flexibility. For example, a graphical user interface is essential, as is efficient processing of deterministic variables. An important improvement in the algorithm would be determining the optimal choice when breaking "ties." There may often be a choice as to which chance node to remove and which one to reverse. Since the removal process can add arcs to the value node, this choice affects the time and memory requirements for future iterations.

In general, even an expensive procedure to determine the optimal sequence of node reductions may be worthwhile.

Several problem structures seem particularly appropriate for special influence diagram algorithms. These include fault trees, dynamic programs, and multivariate normal problems. Maintaining large probabilistic databases as influence diagrams would permit the calculation of arbitrary conditional probability distributions on demand. In particular, influence diagrams would be an ideal framework for representing probabilistic rules and knowledge in an expert system semantic network.

The influence diagram, as presented in this paper, represents a single decision maker's problem. Most games and decentralized team decisions do not fit that model. While the algorithm and analysis for a single decision maker do not directly extend to these problems, it may be worthwhile to develop a generalized influence diagram that is able to represent them.

Acknowledgment

The algorithm was implemented by Jack Breese as it was being developed. His experiences and ideas played a key role. Sam Holtzman, Ron Howard, Jim Matheson, and Scott Olmsted have all done work leading up to these results, and their suggestions and encouragement were helpful, as were the comments of the anonymous referees and the Associate Editor. I must also mention Stuart Dreyfus, who taught me many things, including dynamic programming.

References

BELLMAN, R. 1957. *Dynamic Programming.* Princeton University Press, Princeton, N.J.

HOWARD, R. A., AND J. E. MATHESON. 1981. Influence Diagrams. In *The Principles and Applications of Decision Analysis,* Vol. II, (1984), R. A. Howard and J. E. Matheson (eds.). Strategic Decisions Group, Menlo Park, Calif.

KORSAN, R. J., AND J. E. MATHESON. 1978. Pilot Automated Influence Diagram Decision Aid. SRI International, Menlo Park, Calif.

LAWLER, E. L. 1976. *Combinatorial Optimization: Networks and Matroids.* Holt, Rinehart & Winston, New York.

MERKHOFER, M. W. 1977. The Value of Information Given Decision Flexibility. *Mgmt. Sci.* **23,** 716–727.

MERKHOFER, M. W. 1981. A Computer Aided Decision Structuring Process. SRI International.

MILLER, A. C., M. W. MERKHOFER, R. A. HOWARD, J. E. MATHESON AND T. R. RICE. 1976. *Development of Automated Aids for Decision Analysis.* Stanford Research Institute, Menlo Park, Calif.

OLMSTED, S. M. 1983. On Representing and Solving Decision Problems. Ph.D. Thesis, EES Dept., Stanford University.

RAIFFA, H. 1968. *Decision Analysis.* Addison-Wesley, Reading, Mass.