SCHOOL OF BUSINESS WORKING PAPER NO. 282

A COMPARISON OF GRAPHICAL TECHNIQUES FOR ASYMMETRIC DECISION PROBLEMS: SUPPLEMENT TO MANAGEMENT SCIENCE PAPER

Concha Bielza and Prakash P. Shenoy

December 1998[†]

Decision Analysis Group
Universidad Politécnica de Madrid
Campus de Montegancedo
Boadilla del Monte
28660 Madrid, SPAIN
mcbielza@fi.upm.es

School of Business
University of Kansas
Summerfield Hall
Lawrence, KS 66045-2003, USA
pshenoy@ukans.edu

[†] This working paper is a supplement to Bielza, C. and P. P. Shenoy, "A Comparison of Graphical Techniques for Asymmetric Decision Problems," *Management Science*, to appear in 1999. This working paper should be read in conjunction with the corresponding Management Science paper, and it contains supplementary material that could not be included in the published version due to page limitations.

TABLE OF CONTENTS

A COMPARISON OF GRAPHICAL TECHNIQUES FOR ASYMMETRIC DECISION PROBLEMS: SUPPLEMENT TO MANAGEMENT SCIENCE PAPER

Concha Bielza and Prakash P. Shenoy

ABSTRACT

In this paper, we provide a statement of the Reactor problem, and we describe a representation and solution of this problem using decision trees, Smith, Holtzman and Matheson's asymmetric influence diagrams, Shenoy's asymmetric valuation networks, and Covaliu and Oliver's sequential decision diagrams. We provide details that could not be included in the published paper in Management Science due to page limitations.

Key Words: Asymmetric decision problems, decision trees, influence diagrams, valuation networks, sequential decision diagrams

1 INTRODUCTION

The main goal is of this paper is to provide details of the Reactor problem and its representation and solution using traditional decision trees (DTs), Smith, Holtzman and Matheson's (SHM) [1993] influence diagrams (IDs), Shenoy's [1993b, 1996] valuation networks (VNs), and Covaliu and Oliver's [1995] sequential decision diagrams (SDDs). This paper should be read in conjunction with Bielza and Shenoy's paper in Management Science where the four graphical techniques are compared. This paper contains details that could not be included in the published version due to page limitations.

2 THE REACTOR PROBLEM

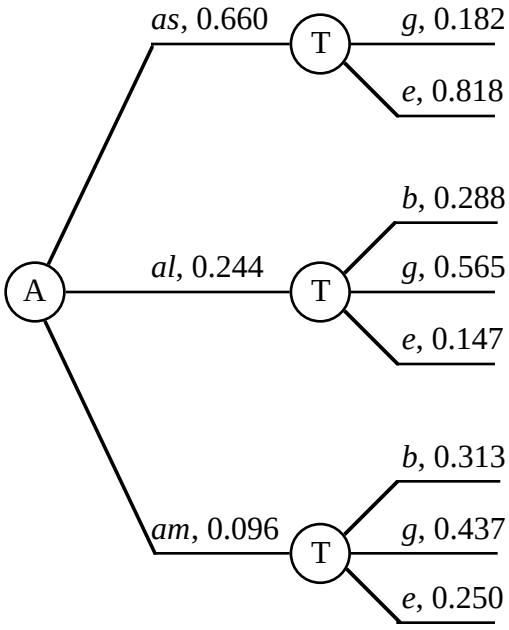
In this section, we describe a small asymmetric decision problem called the Reactor problem. This problem is a modified version of the problem described by Covaliu and Oliver [1995]. In our version, Bayesian revision of probabilities is required during the solution process, and the joint utility function decomposes into three factors only.

An electric utility firm must decide whether to build (D_2) a reactor of advanced design (*a*), a reactor of conventional design (*c*), or neither (*n*). If successful, an advanced reactor is more profitable, but is riskier. Based on past experience, a conventional reactor (C) has probability 0.980 of no failure (*cs*), and a probability 0.020 of a failure (*cf*). On the other hand, an advanced reactor (A) has probability 0.660 of no failure (*as*), probability 0.244 of a limited accident (*al*), and probability

0.096 of a major accident (*am*). The profits for the case the firm builds a conventional reactor are \$8B if there is no failure, and –\$4B if there is a failure. The profits for the case the firm builds an advanced reactor are \$12B if there is no failure, –\$6B if there is a limited accident, and –\$10B if there is a major accident. The firm's utility function is a linear function of the profits.

Before making this decision, the firm can conduct an expensive test of the components of the advanced reactor. The test results (T) can be classified as bad (*b*), good (*g*) or excellent (*e*). The cost of this test is \$1B. The test results are highly correlated with the success or failure of the advanced reactor. Figure 2.1 describes a causal probability model for A and T. If the test results are bad, the Nuclear Regulatory Commission will not permit an advanced reactor. The firm needs to decide (D_1) whether to conduct the test (*t*), or not (*nt*).

Figure 2.1. A causal probability model for A and T in the Reactor problem.



3 DECISION TREES (DTs)

In this section, we will represent and solve the Reactor problem using the decision tree (DT) technique.

Figures 3.2 and 3.3 show a decision tree representation and solution of this problem. Notice that even before the decision tree can be completely specified, the conditional probabilities required by the decision tree representation have to be computed from those specified in the problem as is done in Figure 3.2: The probability tree on the left is used to compute the joint probability distribution for A and T; and the probability tree on the right is used to compute the pre-posterior for test results and the posterior distribution for success and failures of an advanced reactor.

Figure 3.2. The preprocessing of probabilities in the Reactor problem.

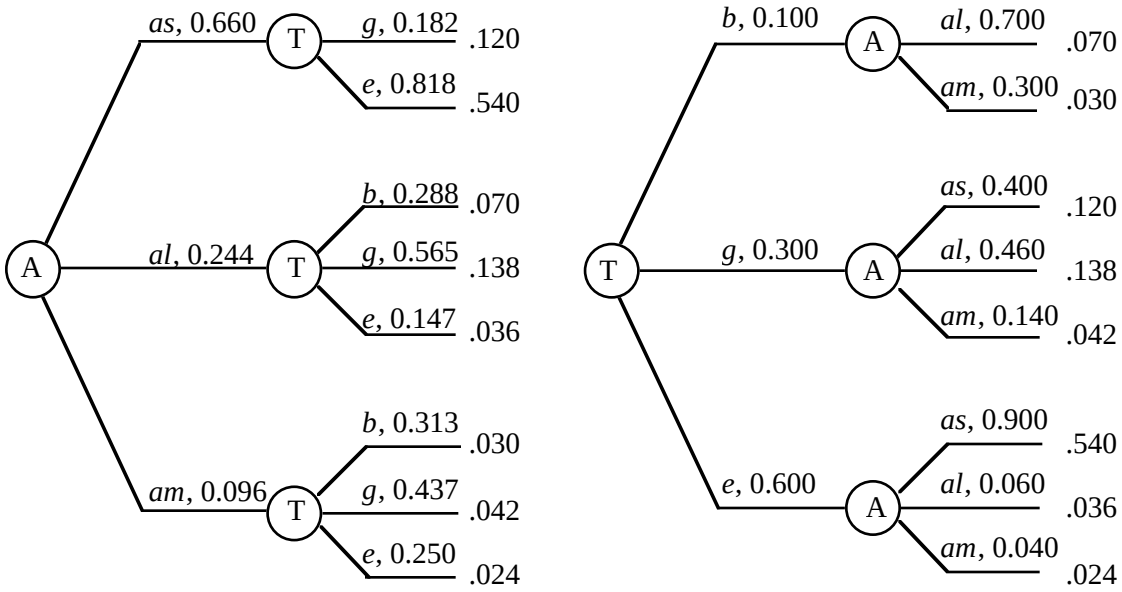


Figure 3.3. A decision tree representation and solution of the Reactor problem.

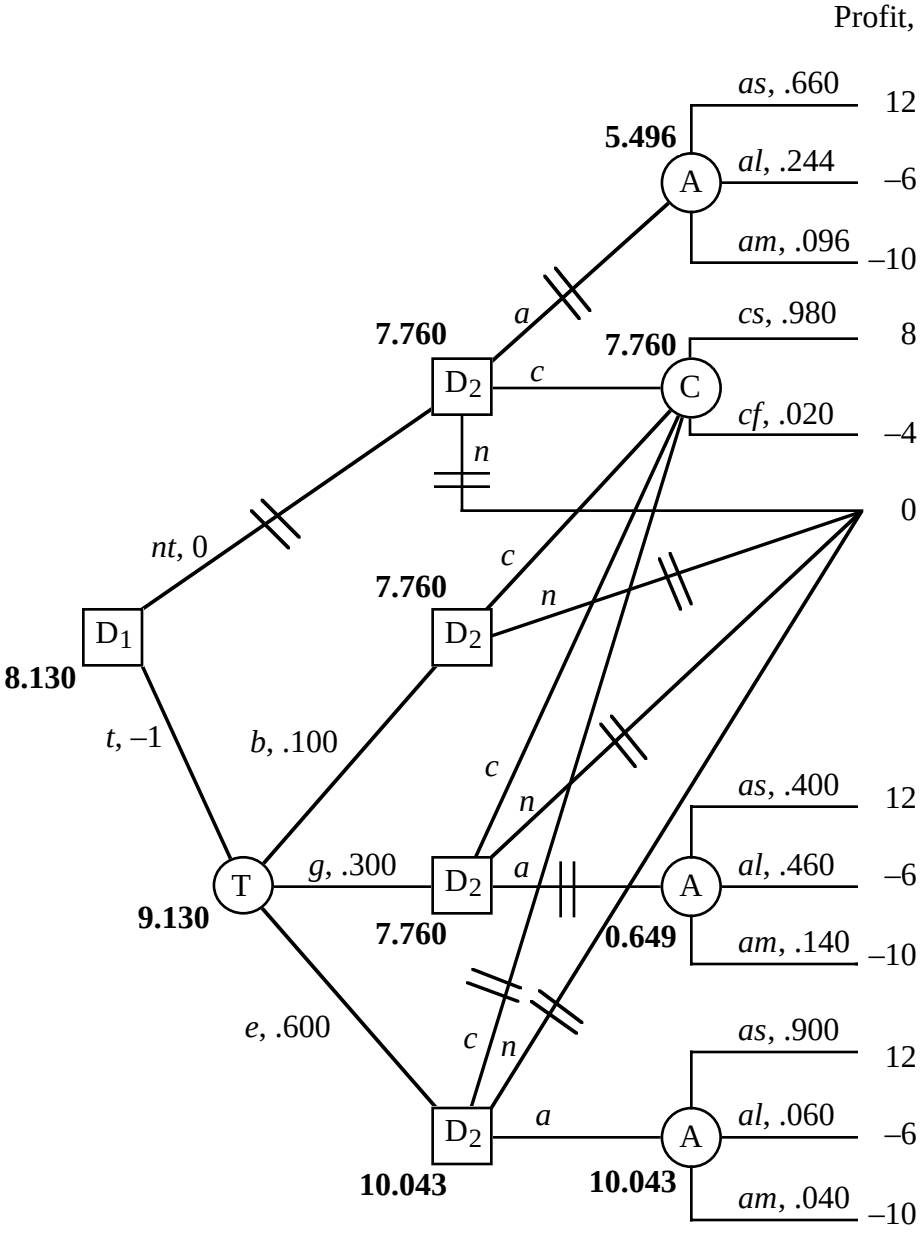


Figure 3.3 shows the solution of the Reactor problem. The optimal strategy is to do the test; build a conventional reactor if the test results are bad or good, and build an advanced reactor if the test results are excellent. The expected profit associated with this strategy is \$8.130B.

Although we have shown the decision tree representation using coalescence [Olmsted 1983], it should be noted that automating coalescence in decision trees is not easy since it involves constructing the complete (uncoalesced) tree and then recognizing repeated subtrees.

The preprocessing requires 21 arithmetic operations (8 multiplications (×), 5 additions (+), and 8 divisions (÷)). Solving the decision tree without coalescence requires 42 operations (12 for re-

ducing four copies of C, 15 for reducing A, 7 for reducing D_2, 5 for reducing T, and 3 for reducing D_1). Thus, the decision tree technique requires a total of 63 arithmetic operations.

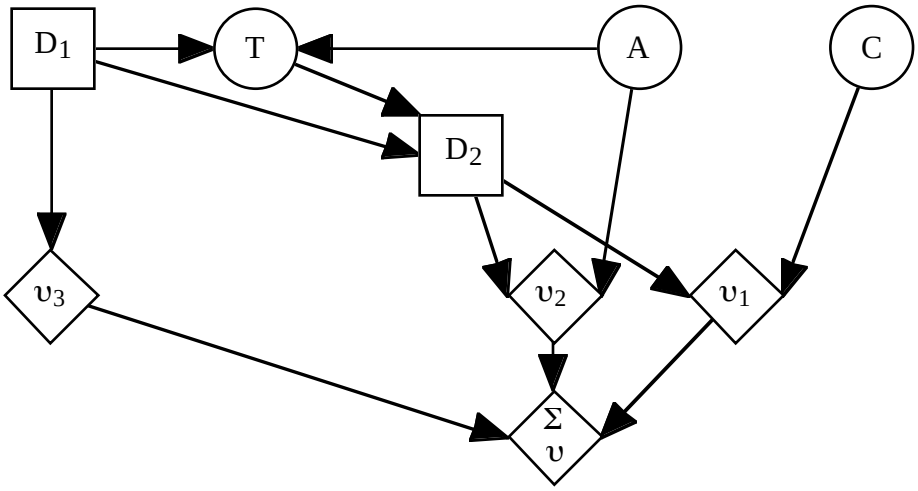
4 ASYMMETRIC INFLUENCE DIAGRAMS (IDs)

In this section, we will represent and solve the Reactor problem using Smith, Holtzman and Matheson's [1993] (henceforth, SHM) asymmetric influence diagram (ID) technique.

4.1 ID Representation

An influence diagram representation of a problem is specified at three levels—graphical, functional, and numerical. At the graphical level, we have a directed acyclic graph, called an influence diagram, that displays decision variables, chance variables, factorization of the joint probability distribution into conditionals, factorization of the joint utility function, and information constraints. Figure 4.1 shows an influence diagram for the Reactor problem at the graphical level.

Figure 4.1. An ID for the Reactor problem.



We could have drawn only one value node υ as a function of D_1, D_2, A, and C. Instead, we have decomposed the single value node υ into three value nodes—υ_1, υ_2 and υ_3. The value node structure is that of a tree. Node υ is the terminal super value node and the others (the leaves, υ_1, υ_2, and υ_3) are non-super value nodes representing the three addends of υ. Although we are increasing the number of value nodes, we will get much computational savings as a result.

At the functional level, we specify the structure of the conditional distribution (or simply, conditional) for each node (except super value nodes) in the ID, and at the numerical level, we specify the numerical details of the probability distributions and the utilities. The key idea of the SHM technique is a new tree representation for describing the conditionals. These are called *distribution*

trees with paths showing the *conditioning scenarios* that lead to *atomic distributions* that describe either probability distributions, set of alternatives, or (expected) utilities, assigned in each conditioning scenario. A conditional for a chance node represents a factor of the joint probability distribution. A conditional for a decision node can be thought of as describing the alternatives available to the decision-maker in each conditioning scenario. A conditional for a value node represents a factor of the joint utility function. For the Reactor problem, the conditionals are shown in Figure 4.2.

Since node D_1 has no conditioning predecessors in the ID, its distribution tree consists of a single atomic distribution. The distribution trees for A and C have also single atomic distributions.

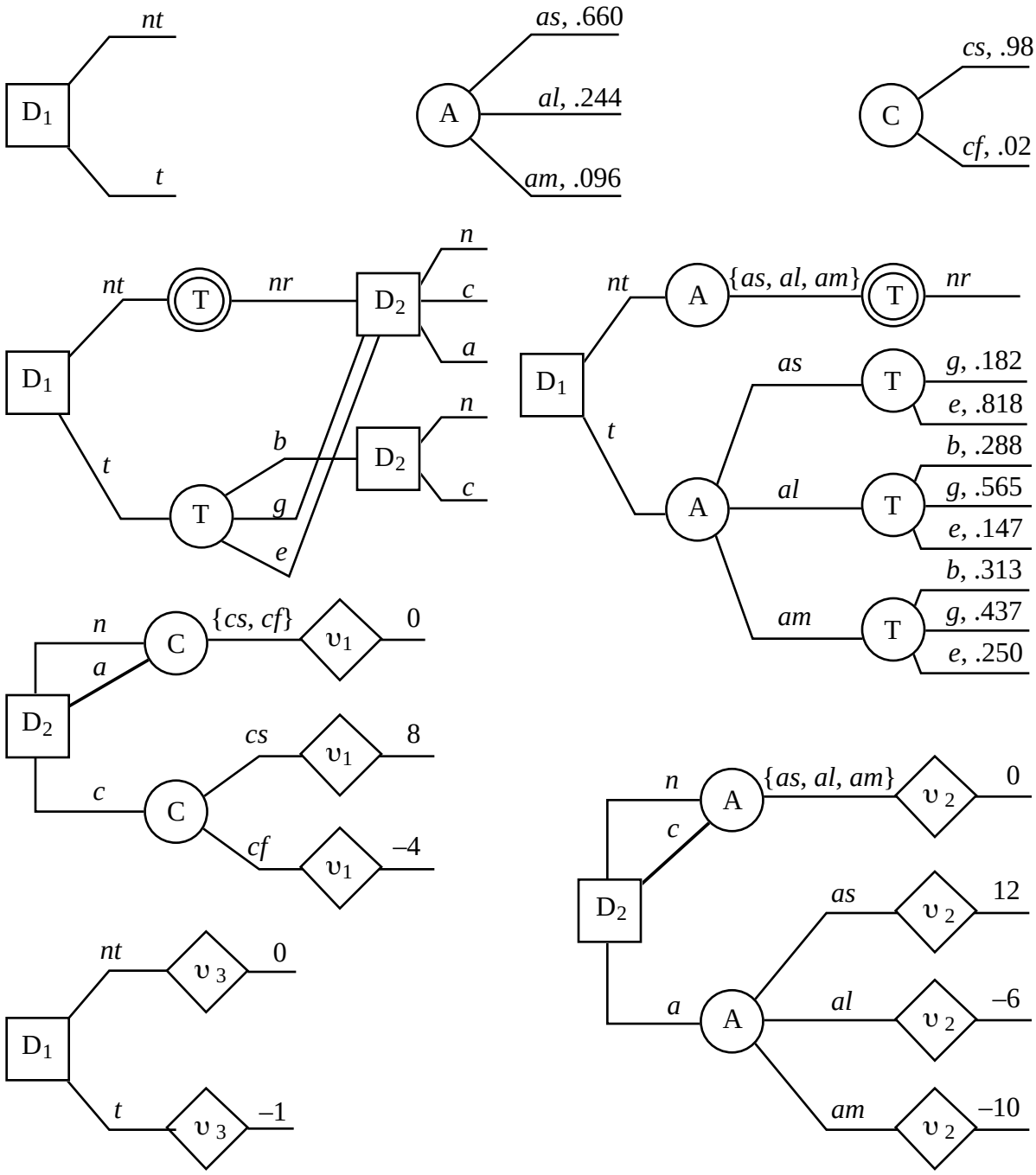
The distribution tree for D_2 has two atomic distributions. The firm will choose among three alternatives (conventional or advanced reactor or neither) only if it decides to not do the test (D_1 = *nt*) or if it conducts the test and its result is good or excellent. The conditional for D_2 is *coalesced*, i.e., the atomic distribution with three alternatives is shared by three distinct scenarios, and is *clipped*, i.e., many branches in conditioning scenarios are omitted because the corresponding conditioning scenarios are impossible. For example, if the firm chooses to not do the test, then it is impossible to observe any test results.

The distribution tree for T shows that if the firm decides to not perform the test (D_1 = *nt*), then T = *nr* with probability 1 regardless of the advanced reactor state. Thus, the conditional for T can be *collapsed* across A given D_1 = *nt*. Collapsed scenarios are shown by indicating the set of possible states on a single edge emanating from the node. They allow the representation of conditional independence between variables that holds only given particular outcomes of some other variables. Deterministic atomic distributions for chance and decision variables are shown by double-bordered nodes.

The conditionals for the three utility nodes provide other examples of coalesced, clipped, and collapsed distributions. They are deterministic nodes because we assign a single utility for each conditioning scenario. Since utility functions are always deterministic, and we use diamond-shaped nodes to indicate utility functions, we do not draw these nodes with a double border.

Another feature of distribution trees not illustrated in the Reactor problem is *unspecified distributions* where certain atomic distributions of a chance node are left unspecified since they are not required during the solution phase. If only the probabilities are unspecified, then we have a *partially* unspecified distribution. All of these features—coalesced, clipped, collapsed, and unspecified distributions—provide a more compact and expressive representation than the usual table in the symmetric ID literature.

Figure 4.2. Distribution trees for the conditionals in the ID.



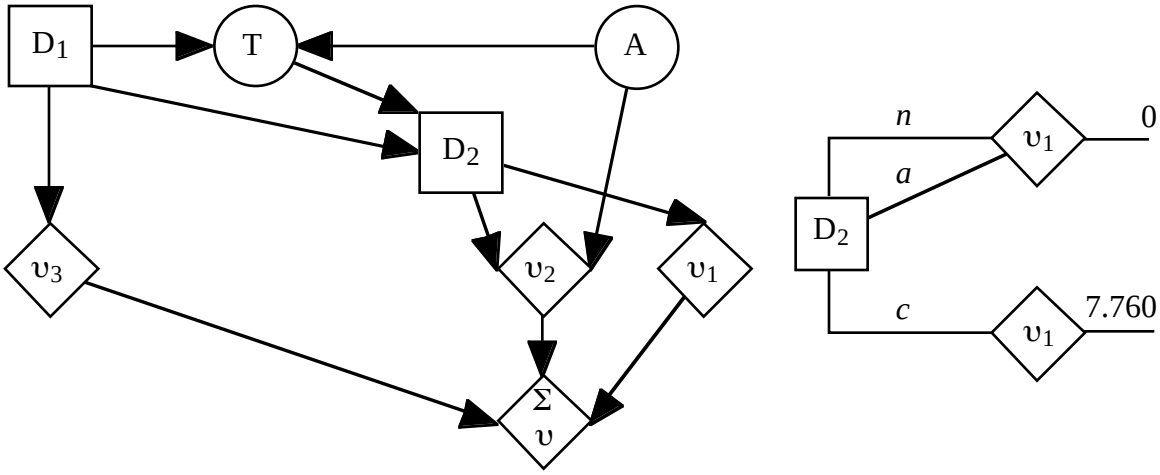
4.2 ID Solution

The algorithm for solving an asymmetric ID is conceptually the same as that for conventional ID. However, SHM describe methods for exploiting different features of a distribution tree (such as clipped scenarios, coalescence, collapsed scenarios, etc.) to simplify the computations.

We solve an ID by reducing variables in a sequence that respects the information constraints. If the true state of a chance variable C is not known at the time the decision maker must choose an alternative from the atomic distribution of decision variable D, then C must be reduced before D, and vice versa. In the Reactor problem, there are two possible reduction sequences, CAD_2TD_1 and ACD_2TD_1. Both of these reduction sequences require the same computational effort. In the following, we use the first reduction sequence CAD_2TD_1 as we do when we solve this problem with the VN and the SDD techniques.

We start by reducing node C. Essentially, we absorb the conditional for C into utility function υ_1 using the expectation operation (following Theorem 5 in Tatman and Shachter [1990]). The expectation operation is carried out by considering each conditioning scenario separately. The case to apply will depend on the structure of the utility and probability functions. Since for $D_2 = n$ or a the utility function υ_1 shares in these scenarios one deterministic distribution, the conditional expected utility function υ_1 must be that deterministic distribution. Only one scenario is left ($D_2 = c$) and we compute the expected utility in the usual manner. Figure 4.3 shows the ID and the distribution tree for υ_1 after reducing C.

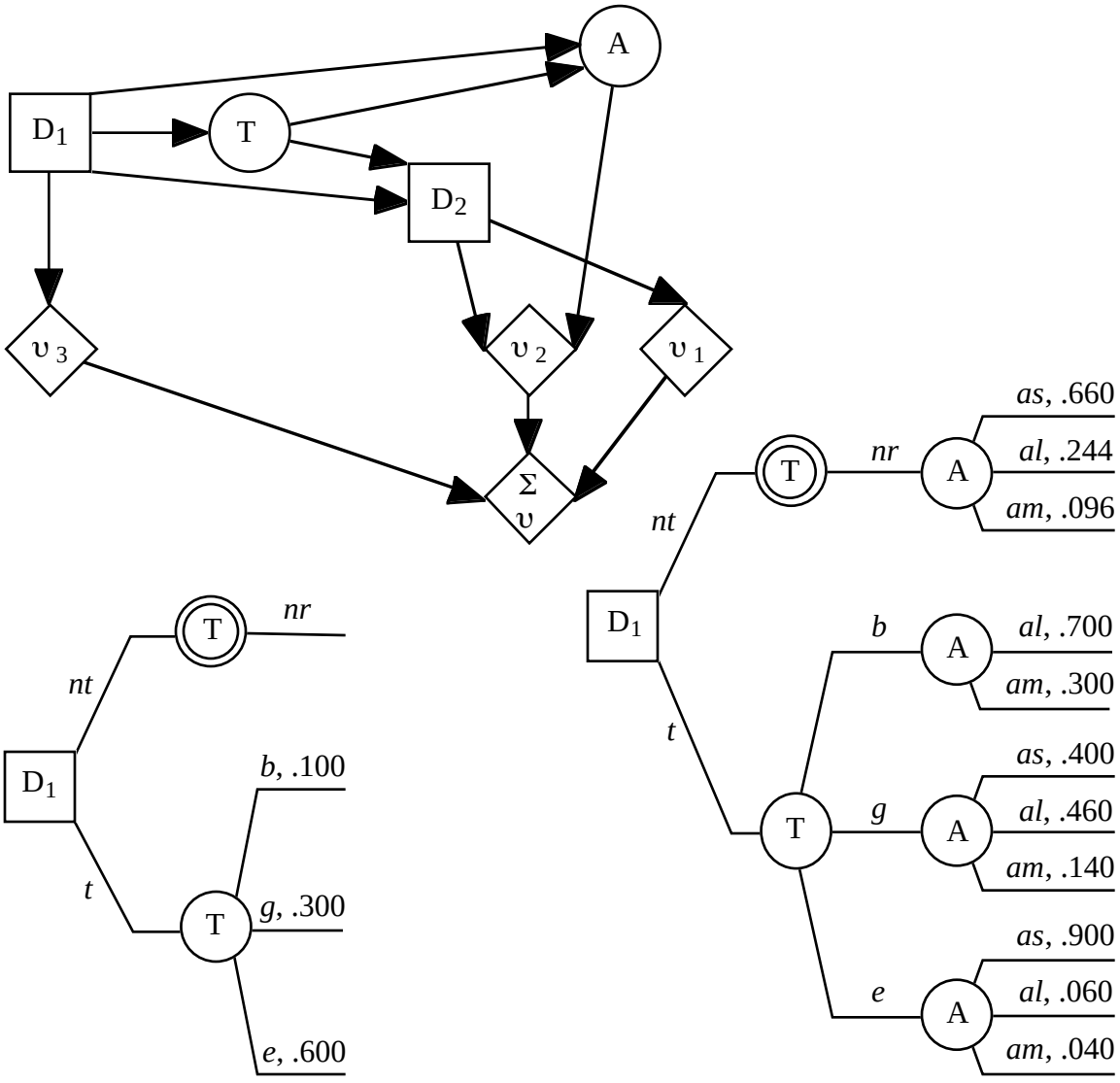
Figure 4.3. The ID and the distribution tree for υ_1 after reducing node C.



Next, we reduce A. To do so, we first reverse arc (A, T), and then absorb the posterior for A into utility function υ_2. To reverse arc (A, T), we first add arc (D_1, A) to the ID so that A and T have the same set of direct predecessors. Any special structure in the original distribution tree is preserved in the resulting one, and nothing new is computed. The arc reversal involves numerical computations (by means of Bayes' rule) only for the scenarios containing $D_1 = t$. The other ones, corresponding to collapsed distributions, remain as before. For example, P(T | *nt*, A) shares one atomic distribution for all possible states of A. Therefore, the pre-posterior distribution P(T | *nt*) must be that shared atomic distribution and the posterior distribution for A must be equal to the

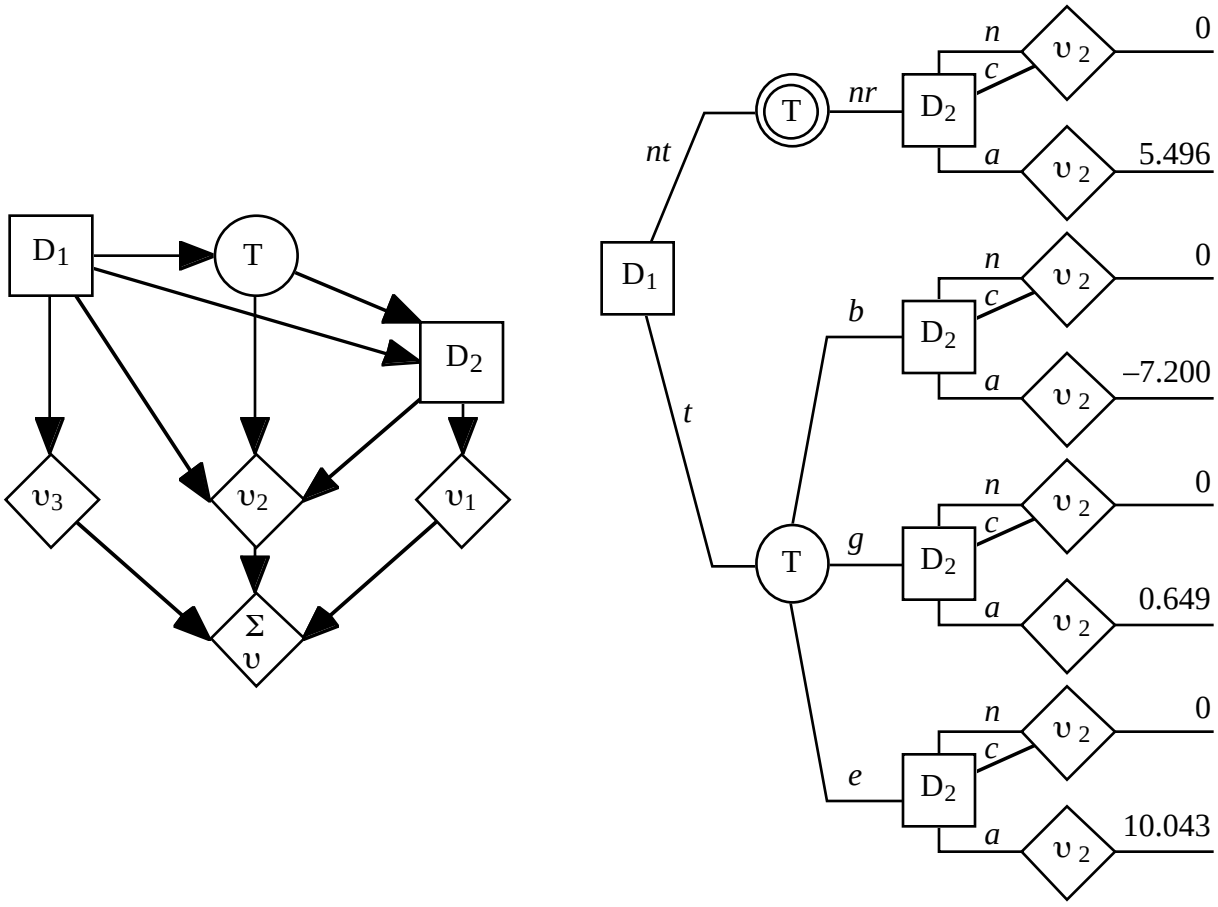
prior. Figure 4.4 shows the ID, the preposterior distribution for T, and the posterior distribution for A, after arc (A, T) reversal.

Figure 4.4. The ID, the preposterior for T, and the posterior for A, after reversal of arc (A, T).



Next we add arcs (D_2, A), (T, υ_2), and (D_1, υ_2) to the ID, modify accordingly the distribution trees for A and υ_2, and absorb the conditional for A into the utility function υ_2 using the expectation operation. Figure 4.5 shows the ID and the distribution tree for υ_2 after reduction of A.

Figure 4.5. The ID and the distribution tree for υ_2 after reducing node A.



Next, we need to reduce D_2. Since D_2 has two value node successors, before we reduce D_2, we introduce a new super-value node ω (as shown in Figure 4.6), and then we merge υ_1 and υ_2 into ω (as per Theorem 5 in Tatman and Shachter [1990]). Figure 4.7 shows the ID and the distribution tree for ω after combining υ_1 and υ_2. Special structures in the conditional distributions are treated as special cases again.

Figure 4.6. The ID after introducing a new super-value node ω.

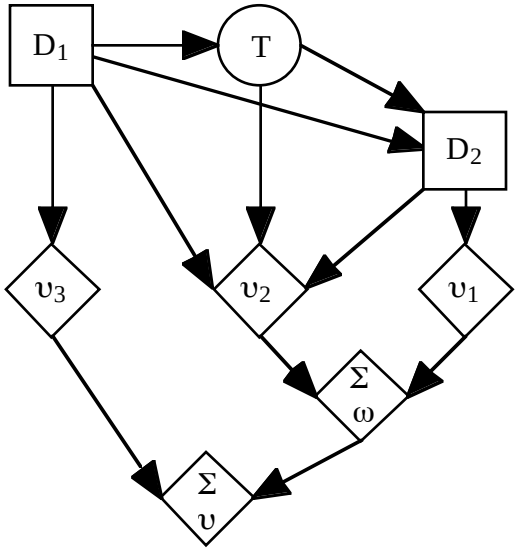
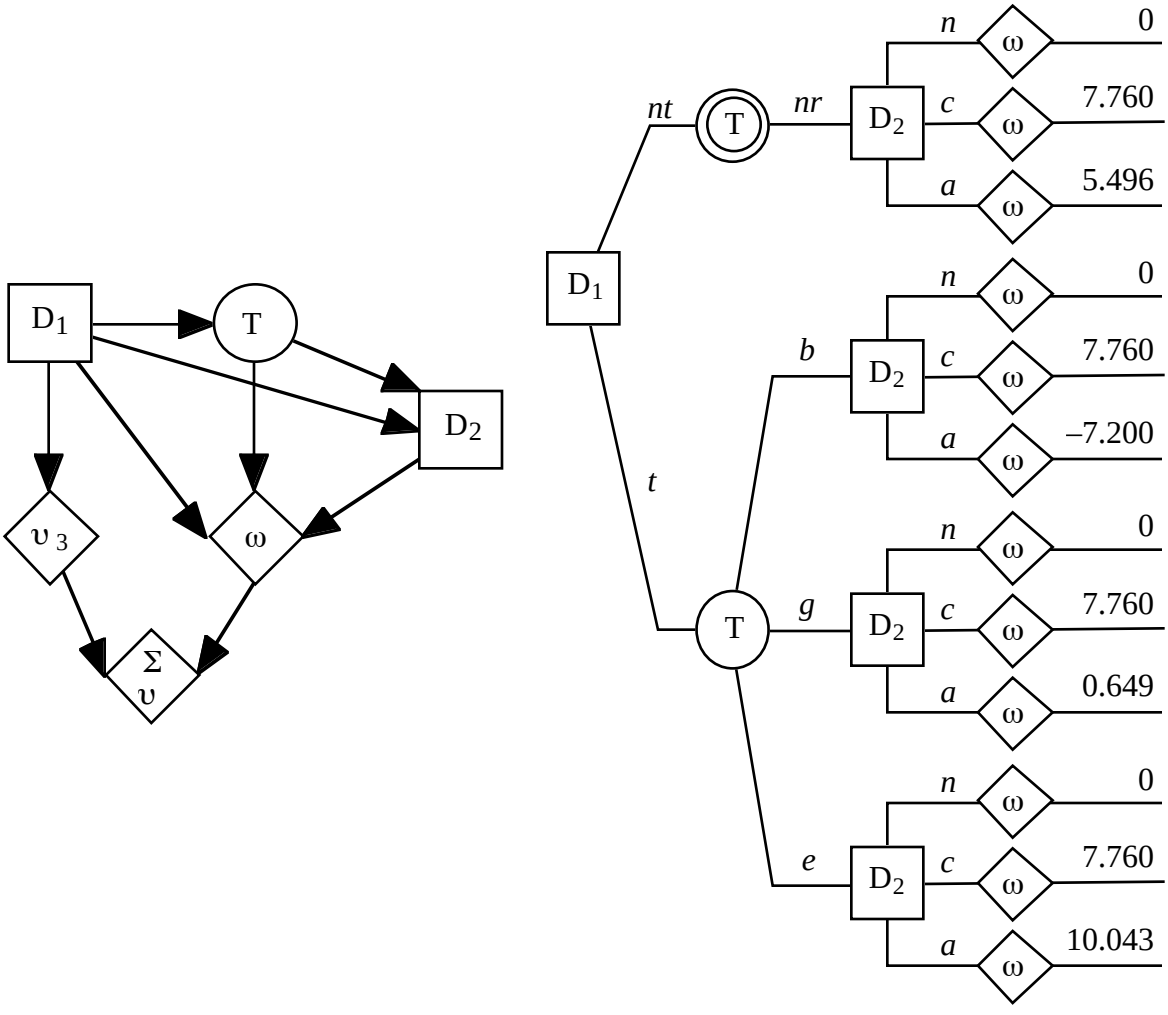
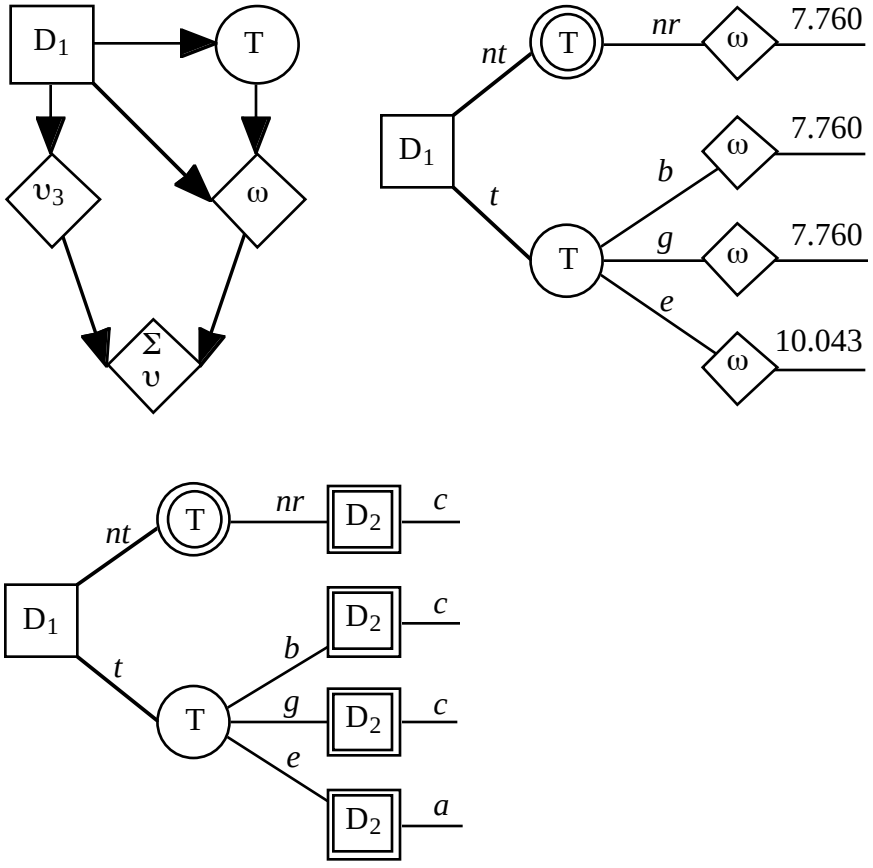


Figure 4.7. The ID and the distribution tree for ω after combining υ_1 and υ_2.



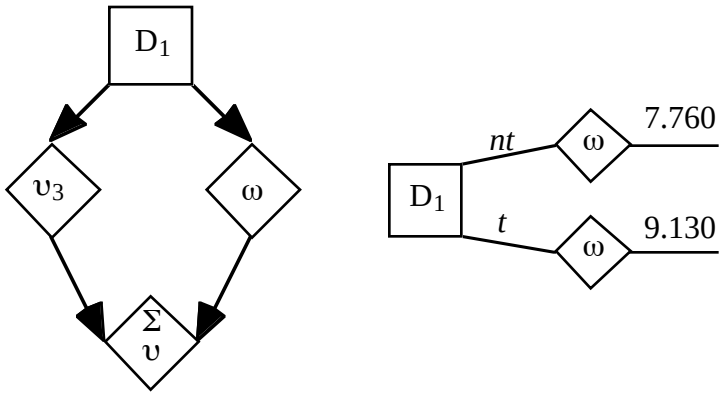
D_2 is now ready to be reduced. We reduce D_2 by maximizing ω over the states of D_2 permitted by the distribution tree for D_2. Notice that this distribution tree (shown in Figure 4.2) has asymmetry in the atomic alternative sets, but this is not exploited either during the reduction of A or during the processing prior to reduction of D_2. Figure 4.8 shows the ID, the distribution tree for ω, and the optimal decision function for D_2, after reduction of D_2.

Figure 4.8. The ID, the distribution tree for ω, and the optimal decision function for D_2, after reduction of D_2.



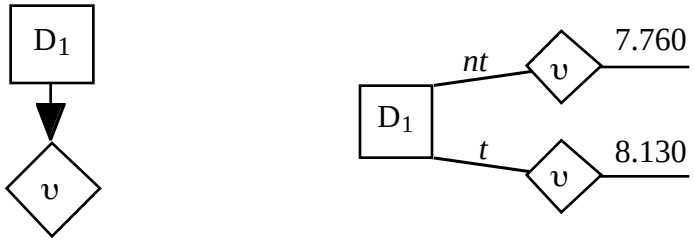
Next, we reduce T. Notice that ω is the only value node that has T in its domain (see Figure 4.8). We absorb the conditional for T (shown in Figure 4.4) into the utility function ω (shown in Figure 4.8) using the expectation operation. Figure 4.9 shows the resulting ID and the distribution tree for ω.

Figure 4.9. The ID and the distribution tree for ω after reducing T.



Finally, we need to reduce D_1. Since D_1 is in the domain of υ_3 and ω, first we combine υ_3 (shown in Figure 4.2) and ω (shown in Figure 4.9) obtaining υ. Figure 4.10 shows the ID and the distribution tree for υ after combining υ_3 and ω. Next we reduce D_1 by maximizing υ over the possible states of D_1. The optimal decision function for D_1 is $D_1 = t$, and the maximum expected utility associated with the optimal strategy is 8.130.

Figure 4.10. The ID and the distribution tree for υ after combining υ_3 and ω.



This completes the solution of the Reactor ID representation. An optimal strategy can be pieced together from the optimal decision function for D_1 and the optimal decision function for D_2 (shown in Figure 4.8). The optimal strategy is to do the test ($D_1 = t$); if the test result is either bad (*b*) or good (*g*), build a conventional reactor ($D_2 = c$); if the test result is excellent (*e*), build an advanced reactor ($D_2 = a$). The expected utility of the optimal strategy is \$8.130B.

Reducing C requires 3 arithmetic operations (2×, 1+), reducing A requires 39 operations (19×, 12+, 8÷), reducing D_2 requires 19 operations (12+, 7>), reducing T requires 5 operations (3×, 2+), and reducing D_1 requires 3 operations (2+, 1>), for a total of 69 operations.

5 ASYMMETRIC VALUATION NETWORKS (VNs)

In this section, we will represent and solve the Reactor problem using Shenoy's [1993b, 1996] asymmetric valuation network (VN) technique. The symmetric VN technique is described in [She-

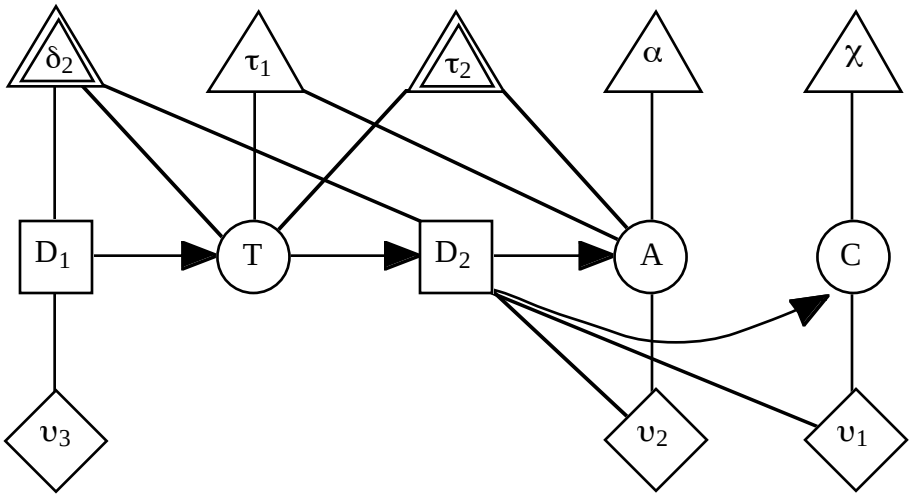
noy 1993a] for the case of a single undecomposed utility function, and in [Shenoy 1992] for the case of an additive decomposition of the joint utility function.

5.1 VN Representation

A valuation network representation is specified at three levels—graphical, dependence, and numerical. The graphical and dependence levels refer to qualitative (or symbolic) knowledge, whereas the numerical level refers to quantitative knowledge.

At the graphical level, we have a graph called a *valuation network*. Figure 5.1 shows a valuation network for the Reactor problem. A valuation network consists of two types of nodes—variable and valuation. Variables are further classified as either decision or chance, and valuations are further classified as either indicator, probability, or utility. Thus, in all there are five different types of nodes—decision, chance, indicator, probability, and utility.

Figure 5.1. A valuation network for the Reactor problem.



Decision nodes correspond to decision variables and are depicted by rectangles. Chance nodes correspond to chance variables and are depicted by circles. This part of VNs is similar to IDs.

Indicator valuations represent qualitative constraints on the joint state spaces of decision and chance variables and are depicted by double-triangular nodes. The set of variables directly connected to an indicator valuation by undirected edges constitutes the domain of the indicator valuation. In the Reactor problem, there are two indicator valuations labeled δ_2 and τ_2. δ_2's domain is $\{D_1, T, D_2\}$ and it represents the constraints that the test results are available only in the case we decide to do the test, and that the alternatives at D_2 depend on the choices at D_1 and the test results T. τ_2's domain is $\{T, A\}$ and it represents the constraint that if A = *as*, then T = *b* is not possible.

Utility valuations represent additive factors of the joint utility function and are depicted by diamond-shaped nodes. The set of variables directly connected to a utility valuation constitutes the domain of the utility valuation. In the Reactor problem, there are three additive utility valuations labeled υ_1, υ_2, and υ_3, with domains $\{D_2, C\}$, $\{D_2, A\}$, and $\{D_1\}$, respectively.

Probability valuations represent multiplicative factors of the family of joint probability distributions for the chance variables in the problem, and are depicted by triangular nodes. The set of all variables directly connected to a probability valuation constitutes the domain of the probability valuation. In the Reactor problem, there are three probability valuations labeled τ_1, α, and χ, with domains $\{A, T\}$, $\{A\}$, and $\{C\}$, respectively.

The specification of the valuation network at the graphical level includes directed arcs between pairs of distinct variables. These directed arcs represent information constraints. Suppose R is a chance variable and D is a decision variable. An arc (R, D) means that the true state of R is known to the decision maker (DM) at the time the DM has to choose an alternative from D's state space, and, conversely, an arc (D, R) means that the true state of R is not known to the DM at the time the DM has to choose an alternative from D's state space.

Next, we specify a valuation network representation at the dependence level. At this level, we specify the state spaces of all variables and we specify the details of the indicator valuations.

Associated with each variable X is a *state space* 0_X. As in the cases of IDs and SDDs, we assume that all variables have finite state spaces. Suppose s is a subset of variables. An *indicator* valuation for s is a function $\iota: 0_s \rightarrow \{0, 1\}$. An efficient way of representing an indicator valuation is simply to describe the elements of the state space that have value 1, i.e., we represent ι by Ω_ι where $\Omega_\iota = \{\mathbf{x} \in 0_s \mid \iota(\mathbf{x}) = 1\}$. Obviously, $\Omega_\iota \subseteq 0_s$. To minimize jargon, we also call Ω_ι an indicator valuation for s. In the Reactor problem, the details of the two indicator valuations are as follows:

$$\Omega_{\delta_2} = \{(nt, nr, n), (nt, nr, c), (nt, nr, a), (t, b, n), (t, b, c), (t, g, n), (t, g, c), (t, g, a), (t, e, n), (t, e, c), (t, e, a)\}$$

$$\Omega_{\tau_2} = \{(as, nr), (as, g), (as, e), (al, nr), (al, b), (al, g), (al, e), (am, nr), (am, b), (am, g), (am, e)\}$$

Notice that the indicator valuation Ω_{δ_2} is identical to the scenarios in the distribution tree for D_2 depicted in Figure 4.2. The indicator valuation Ω_{τ_2} rules out the scenario A = *as*, T = *b*.

Before we can specify the valuation network at the numerical level, it is necessary to introduce the notion of effective state spaces for subsets of variables. Suppose that each variable is in the domain of some indicator valuation. (If not, we can create "vacuous" indicator valuations that are identically one for every state of such variables.) We define *combination* of indicator valuations as pointwise Boolean multiplication, and *marginalization* of an indicator valuation as Boolean addition over the state space of reduced variables. Then, the *effective state space* for a subset s of variables,

denoted by Ω_s, is defined as follows: First we combine all indicator valuations that include some variable from s in their domains, and next we marginalize the combination so that only the variables in s remain in the marginal. Shenoy [1994] has shown that these definitions of combination and marginalization satisfy the three axioms that permit local computation [Shenoy and Shafer 1990]. Thus, the computation of the effective state spaces can be done efficiently using local computation. For example, consider the effective state space for subset $\{T, D_2, A\}$. By definition, $\Omega_{\{T, D_2, A\}} = (\delta_2 \otimes \tau_2)^{\downarrow \{T, D_2, A\}}$. However, we can compute this more efficiently as follows. $\Omega_{\{T, D_2, A\}} = \delta_2^{\downarrow \{T, D_2\}} \otimes \tau_2$. Notice that the combination operation in $(\delta_2 \otimes \tau_2)^{\downarrow \{T, D_2, A\}}$ is on the state space of $\{D_1, T, D_2, A\}$ whereas the combination operation in $\delta_2^{\downarrow \{T, D_2\}} \otimes \tau_2$ is on the state space of $\{T, D_2, A\}$.

Finally, we specify a valuation network at the numerical level. At this level, we specify the details of the utility and probability valuations. A *utility* valuation υ for s is a function $\upsilon: \Omega_s \rightarrow R$, where R is the set of real numbers. The values of υ are utilities. In the Reactor problem, there are three utility valuations whose details are shown in Table 5.1.

Table 5.1. Utility valuations in the Reactor problem.

$\Omega_{\{D_2, C\}}$		υ_1
n	*cs*	0
n	*cf*	0
c	*cs*	8
c	*cf*	–4
a	*cs*	0
a	*cf*	0

$\Omega_{\{D_2, A\}}$		υ_2
n	*as*	0
n	*al*	0
n	*am*	0
c	*as*	0
c	*al*	0
c	*am*	0
a	*as*	12
a	*al*	–6
a	*am*	–10

Ω_{D_1}	υ_3
nt	0
t	–1

Table 5.2. Probability valuations in the Reactor problem.

Ω_C	χ
cs	.98
cf	.02

Ω_A	α
as	.660
al	.244
am	.096

$\Omega_{\{A, T\}}$		τ_1
as	*nr*	1
as	*g*	.182
as	*e*	.818
al	*nr*	1
al	*b*	.288
al	*g*	.565
al	*e*	.147
am	*nr*	1
am	*b*	.313
am	*g*	.437
am	*e*	.250

A *probability* valuation π for s is a function $\pi: \Omega_s \rightarrow [0, 1]$. The values of π are probabilities. In the Reactor problem, there are three probability valuations whose details are shown in Table 5.2. What do these probability valuations mean? χ is the marginal for C, α is the marginal for A, and $\delta_2^{\downarrow\{D_1, T\}} \otimes \tau_2 \otimes \tau_1$ is the conditional for T given A and D_1. Thus the conditional for T factors into three valuations such that τ_1 has the numeric information and δ_2 and τ_2 include the structural information.

Notice that the utility and probability valuations are described only for effective state spaces which are computed (using local computation) from the specifications of the indicator valuations. There is no redundancy in the representation. However, in υ_2, unlike the ID representation, the irrelevance of A in scenarios where $D_2 = n$ or c is not represented in the VN representation because we are unable to. Also, in υ_1, the irrelevance of C in scenarios $D_2 = n$ or a is not represented. This completes the valuation network representation of the Reactor problem.

5.2 VN Solution

In this section, first we sketch the fusion algorithm for solving valuation network representations of decision problems, and then we solve the Reactor problem in complete detail.

The fusion algorithm is essentially the same as in the symmetric case [Shenoy 1992]. The main difference is in how indicator valuations are handled. Since indicator valuations are identically one on effective state spaces, there are no numeric computations involved in combining indicator valuations. Indicator valuations do contribute domain information and cannot be totally ignored. In the fusion algorithm, we reduce a variable by doing a fusion operation on the set of all valuations (utility, probability, and indicator) with respect to the variable. All numeric computations are done

on effective state spaces only. This means that the effective state spaces may need to be computed prior to doing the fusion operation if the effective state space has not been already computed during the representation phase.

Fusion with respect to a decision variable D is defined as follows. The utility, probability, and indicator valuations whose domains do not include D remain unchanged. All utility valuations that include D in their domain are combined together, and the resulting utility valuation υ is marginalized such that D is eliminated from its domain. A new indicator valuation ζ_D corresponding to the decision function for D is created. All probability and indicator valuations that include D in their domain are combined together and the resulting probability valuation ρ is combined with ζ_D and the result is marginalized so that D is eliminated from its domain.

Fusion with respect to a chance variable C is defined as follows. The utility, probability, and indicator valuations whose domains do not include C remain unchanged. A new probability valuation, say ρ, is created by combining all probability and indicator valuations whose domain include C and marginalizing C out of the combination. Finally, we combine all probability and indicator valuations whose domains include C, divide the resulting probability valuation by the new probability valuation ρ that was created, combine the resulting probability valuation with the utility valuations whose domains include C, and finally marginalize the resulting utility valuation such that C is eliminated from its domain. In some special cases—such as if ρ is identically one, or if C is the only chance variable left—we can avoid creating a new probability valuation and the corresponding division.

The solution of the Reactor problem using the fusion algorithm is as follows.

Fusion with respect to C. First we fuse valuations in $\{\delta_2, \tau_2, \upsilon_1, \upsilon_2, \upsilon_3, \chi, \alpha, \tau_1\}$ with respect to C. Since $\chi^{\downarrow \varnothing}$ is identically one,

$$\mathrm{Fus}_C\{\delta_2, \tau_2, \upsilon_1, \upsilon_2, \upsilon_3, \chi, \alpha, \tau_1\} = \{\delta_2, \tau_2, \upsilon_2, \upsilon_3, (\upsilon_1 \otimes \chi)^{\downarrow D_2}, \alpha, \tau_1\}.$$

Let υ_4 denote $(\upsilon_1 \otimes \chi)^{\downarrow D_2}$. The result of fusion with respect to C is shown graphically in Figure 5.2. The details of the numerical computation involved in the fusion operation are shown in Table 5.3.

Figure 5.2. Fusion with respect to C: Before and after.

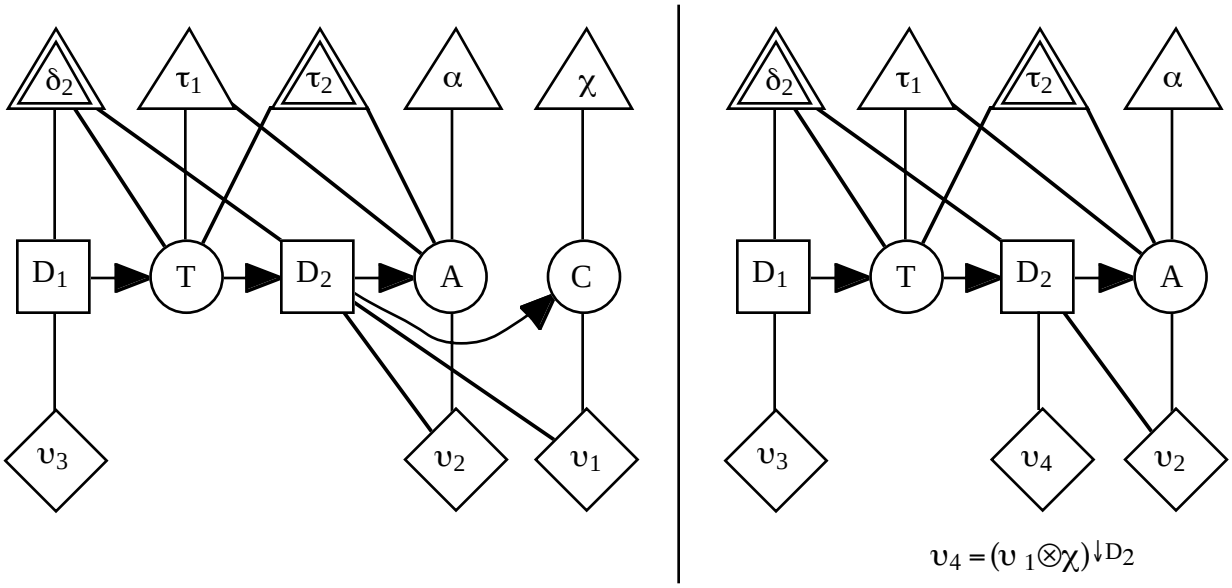


Table 5.3. Details of fusion with respect to C.

$\Omega_{\{D_2, C\}}$	υ_1	χ	$\upsilon_1 \otimes \chi$	$(\upsilon_1 \otimes \chi)^{\downarrow D_2} = \upsilon_4$
n *cs*	0	0.98	0	0
n *cf*	0	0.02	0	
c *cs*	8	0.98	7.840	7.760
c *cf*	–4	0.02	–0.080	
a *cs*	0	0.98	0	0
a *cf*	0	0.02	0	

Fusion with respect to A. Next, we fuse the valuations in $\{\delta_2, \tau_2, \upsilon_2, \upsilon_3, \upsilon_4, \alpha, \tau_1\}$ with respect to A.

$$\text{Fus}_A\{\delta_2, \tau_2, \upsilon_2, \upsilon_3, \upsilon_4, \alpha, \tau_1\} = \{\delta_2, \upsilon_3, \upsilon_4, (\upsilon_2 \otimes (\alpha \otimes \tau_1 \otimes \tau_2)/(\alpha \otimes \tau_1 \otimes \tau_2)^{\downarrow T})^{\downarrow \{D_2, T\}},$$
$$(\alpha \otimes \tau_1 \otimes \tau_2)^{\downarrow T}\}.$$

Let υ_5 denote $(\upsilon_2 \otimes (\alpha \otimes \tau_1 \otimes \tau_2)/(\alpha \otimes \tau_1 \otimes \tau_2)^{\downarrow T})^{\downarrow \{D_2, T\}}$, and let τ' denote $(\alpha \otimes \tau_1 \otimes \tau_2)^{\downarrow T}$. The result of fusion with respect to A is shown graphically in Figure 5.3. Details of fusion are shown in Tables 5.4 and 5.5.

Figure 5.3. Fusion with respect to A: Before and after.

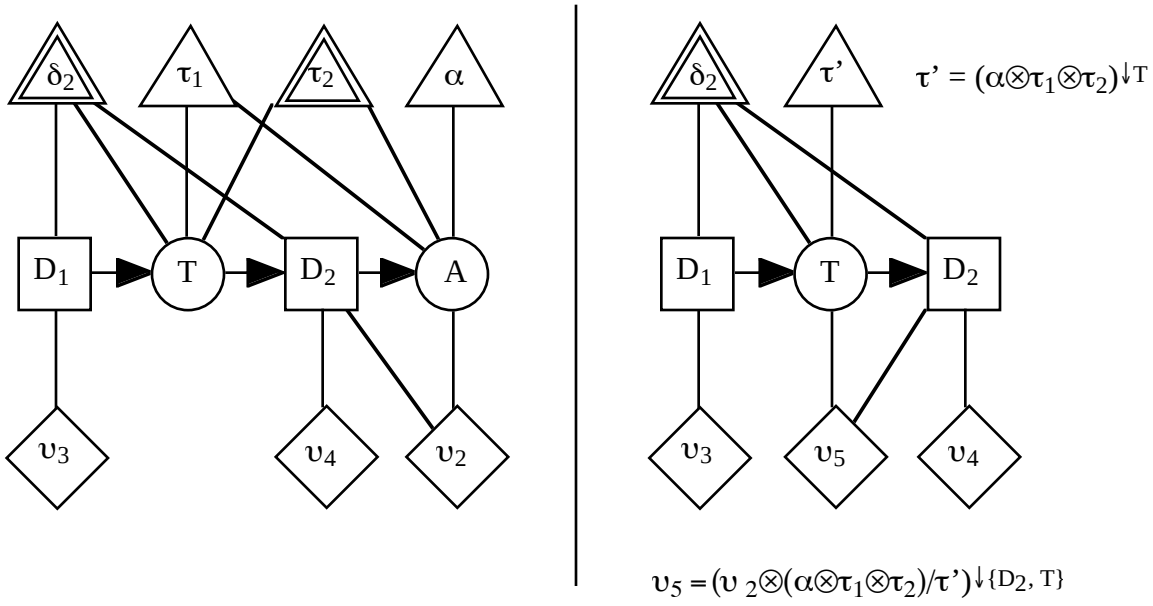


Table 5.4. Details of fusion with respect to A (continued in Table 5.5).

$\Omega_{\{T, A\}}$		α	$\tau_1 \otimes \tau_2 = \tau$	$\alpha \otimes \tau$	$(\alpha \otimes \tau)^{\downarrow T} = \tau'$	$(\alpha \otimes \tau)/(\alpha \otimes \tau)^{\downarrow T}$
nr	*as*	0.660	1	0.660	1	0.660
nr	*al*	0.244	1	0.244		0.244
nr	*am*	0.096	1	0.096		0.096
b	*al*	0.244	0.288	0.070	0.100	0.700
b	*am*	0.096	0.313	0.030		0.300
g	*as*	0.660	0.182	0.120	0.300	0.400
g	*al*	0.244	0.565	0.138		0.460
g	*am*	0.096	0.437	0.042		0.140
e	*as*	0.660	0.818	0.540	0.600	0.900
e	*al*	0.244	0.147	0.036		0.060
e	*am*	0.096	0.250	0.024		0.040

Table 5.5. Details of fusion with respect to A (continued from Table 5.4).

$\Omega_{\{T, D_2, A\}}$	υ_2	$(\alpha\otimes\tau)/(\alpha\otimes\tau)^{\downarrow T}$	$\upsilon_2\otimes(\alpha\otimes\tau)/(\alpha\otimes\tau)^{\downarrow T} = \upsilon'$	$(\upsilon')^{\downarrow\{T, D_2\}} = \upsilon_5$
nr n as	0	0.660	0	0
nr n al	0	0.244	0	
nr n am	0	0.096	0	
nr c as	0	0.660	0	0
nr c al	0	0.244	0	
nr c am	0	0.096	0	
nr a as	12	0.660	7.920	5.496
nr a al	–6	0.244	–1.464	
nr a am	–10	0.096	–0.960	
b n al	0	0.700	0	0
b n am	0	0.300	0	
b c al	0	0.700	0	0
b c am	0	0.300	0	
g n as	0	0.400	0	0
g n al	0	0.460	0	
g n am	0	0.140	0	
g c as	0	0.400	0	0
g c al	0	0.460	0	
g c am	0	0.140	0	
g a as	12	0.400	4.800	0.649
g a al	–6	0.460	–2.760	
g a am	–10	0.140	–1.400	
e n as	0	0.900	0	0
e n al	0	0.060	0	
e n am	0	0.040	0	
e c as	0	0.900	0	0
e c al	0	0.060	0	
e c am	0	0.040	0	
e a as	12	0.900	10.800	10.043
e a al	–6	0.060	–0.360	
e a am	–10	0.040	–0.400	

Notice that all computations are done on effective state spaces, and so we need to compute the effective state space of $\{T, D_2, A\}$ prior to doing the fusion (since it has not been already computed during the representation stage). Typically, we can do so using the local computational algo-

rithm of Shenoy and Shafer [1990]. Here, this means the effective state space of {T, D_2, A} is computed as $\delta_2^{\downarrow\{T, D_2\}} \otimes \tau_2$ (as explained in Section 5.1). One consequence of working on effective state spaces is that we avoid the computation of utilities for impossible scenarios such as T = *b*, D_2 = *a*, A = *as*, etc., that are done in IDs (see Figure 4.5—the presence of scenario D_1 = *t*, T = *b*, D_2 = *a* in the utility function υ_2 is the result of averaging over the scenarios D_1 = *t*, T = *b*, D_2 = *a*, A = *as*, D_1 = *t*, T = *b*, D_2 = *a*, A = *al*, D_1 = *t*, T = *b*, D_2 = *a*, A = *am*).

Fusion with respect to D_2. Next we fuse $\{\delta_2, \upsilon_3, \upsilon_4, \upsilon_5, \tau'\}$ with respect to D_2. Since D_2 is a decision variable,

$$\mathrm{Fus}_{D_2}\{\delta_2, \upsilon_3, \upsilon_4, \upsilon_5, \tau'\} = \{(\delta_2 \otimes \zeta_{D_2})^{\downarrow\{D_1, T\}}\, \upsilon_3, (\upsilon_4 \otimes \upsilon_5)^{\downarrow T}, \tau'\},$$

where ζ_{D_2} is the indicator valuation representation of the decision function for D_2 (determined during the computation of $(\upsilon_4 \otimes \upsilon_5)^{\downarrow T}$). Let υ_6 denote $(\upsilon_4 \otimes \upsilon_5)^{\downarrow T}$, and δ_2' denote $(\delta_2 \otimes \zeta_{D_2})^{\downarrow\{D_1, T\}}$. The result of fusion with respect to D_2 is shown graphically in Figure 5.4. The details of the numerical computation involved in the fusion operation are shown in Table 5.6.

Figure 5.4. Fusion with respect to D_2: Before and after.

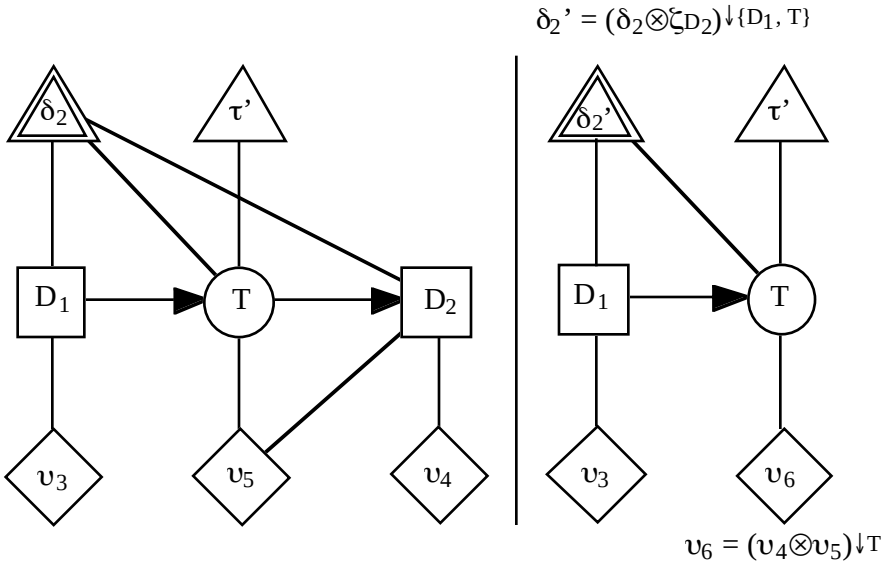


Table 5.6. Details of fusion with respect to D_2.

$\Omega_{\{T, D_2\}}$		υ_4	υ_5	$\upsilon_4 \otimes \upsilon_5$	$(\upsilon_4 \otimes \upsilon_5)^{\downarrow T} = \upsilon_6$	Ψ_{D_2}
nr	*n*	0	0	0		
nr	*c*	7.760	0	7.760	7.760	*c*
nr	*a*	0	5.496	5.496		
b	*n*	0	0	0		
b	*c*	7.760	0	7.760	7.760	*c*
g	*n*	0	0	0		
g	*c*	7.760	0	7.760	7.760	*c*
g	*a*	0	0.649	0.649		
e	*n*	0	0	0		
e	*c*	7.760	0	7.760		
e	*a*	0	10.043	10.043	10.043	*a*

Details of the symbolic computation of δ_2' are as follows. It follows from Table 5.6 that the effective state space representation of the decision function for D_2 is $\Omega_{\zeta_{D_2}} = \{(nr, c), (b, c), (g, c), (e, a)\}$. δ_2 was described earlier in Section 5.1. Therefore, the effective state space representation of $\delta_2 \otimes \zeta_{D_2}$ is $\Omega_{\zeta_{D_2} \otimes \delta_2} = \{(nt, nr, c), (t, b, c), (t, g, c), (t, e, a)\}$. Finally, the effective state space representation of $\delta_2' = (\delta_2 \otimes \zeta_{D_2})^{\downarrow \{D_1, T\}}$ is $\Omega_{\delta_2'} = \{(nt, nr), (t, b), (t, g), (t, e)\}$.

Fusion with respect to T. Next we fuse $\{\delta_2', \upsilon_3, \upsilon_6, \tau'\}$ with respect to T. Since T is the only chance variable,

$$\mathrm{Fus}_R\{\delta_2', \upsilon_3, \upsilon_6, \tau'\} = \left\{\upsilon_3, (\tau' \otimes \delta_2' \otimes \upsilon_6)^{\downarrow D_1}\right\}.$$

Let υ_7 denote $(\tau' \otimes \delta_2' \otimes \upsilon_6)^{\downarrow D_1}$. The result of fusion with respect to T is shown graphically in Figure 5.5. The details of the numerical computation involved in the fusion operation are shown in Table 5.7.

Figure 5.5. Fusion with respect to T: Before and after

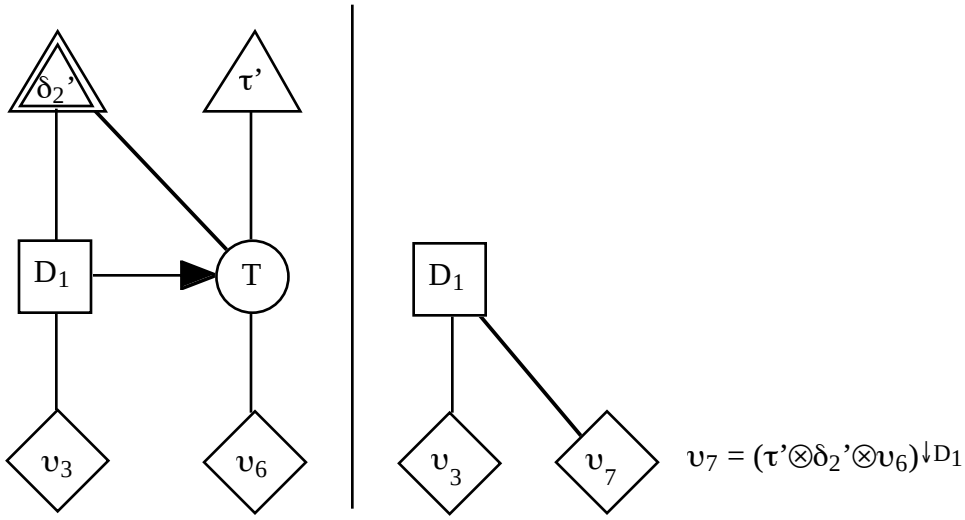


Table 5.7. Details of fusion with respect to T.

$\Omega_{\{D_1, T\}}$		$\delta_2' \otimes \tau'$	υ_6	$\delta_2' \otimes \tau' \otimes \upsilon_6$	$(\delta_2' \otimes \tau' \otimes \upsilon_6)^{\downarrow D_1} = \upsilon_7$
nt	*nr*	1	7.760	7.760	7.760
t	*b*	0.100	7.760	0.776	9.130
t	*g*	0.300	7.760	2.328	
t	*e*	0.600	10.043	6.026	

Fusion with respect to D_1. Next, we fuse $\{\upsilon_3, \upsilon_7\}$ with respect to D_1. Since D_1 is a decision variable,

$$\mathrm{Fus}_{D_1}\{\upsilon_3, \upsilon_7\} = \left\{(\upsilon_3 \otimes \upsilon_7)^{\downarrow \varnothing}\right\}.$$

Let υ_8 denote $(\upsilon_3 \otimes \upsilon_7)^{\downarrow \varnothing}$. The result of fusion with respect to D_1 is shown in Figure 5.6. The details of the numerical computation involved in the fusion operation are shown in Table 5.8.

Figure 5.6. Fusion with respect to D_1: Before and after

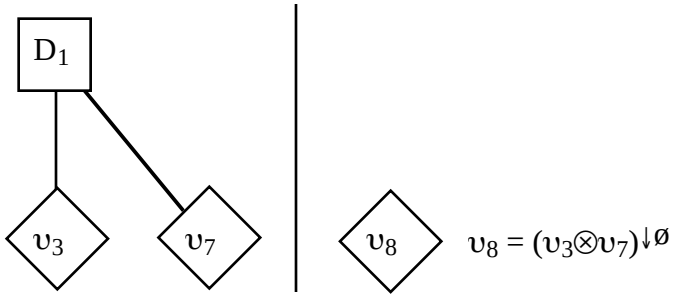


Table 5.8. Details of fusion with respect to D_1.

Ω_{D_1}	υ_7	υ_3	$\upsilon_7 \otimes \upsilon_3$	$(\upsilon_7 \otimes \upsilon_3)^{\downarrow \varnothing}(\blacklozenge)$	$\Psi_{D_1}(\blacklozenge)$
nt	7.760	0	7.760		
t	9.130	–1	8.130	8.130	*t*

This completes the fusion algorithm. An optimal strategy can be pieced together from the decision functions Ψ_{D_1} (in Table 5.8) and Ψ_{D_2} (in Table 5.6). An optimal strategy is to do the test (*t*); if the test results are *b* or *g*, then build a conventional reactor (*c*), and if the test result is excellent, then build an advanced reactor (*a*). The expected utility of the optimal strategy is \$8.130B ($= (\upsilon_7 \otimes \upsilon_3)^{\downarrow \varnothing}(\blacklozenge)$).

Fusion with respect to C requires 9 operations (6×, 3+), fusion with respect to A requires 80 operations (42×, 27+, 11÷), fusion with respect to D_2 requires 18 operations (11+, 7>), fusion with respect to T requires 6 operations (4×, 2+), and fusion with respect to D_1 requires 3 operations (2+, 1>) for a total of 116 operations.

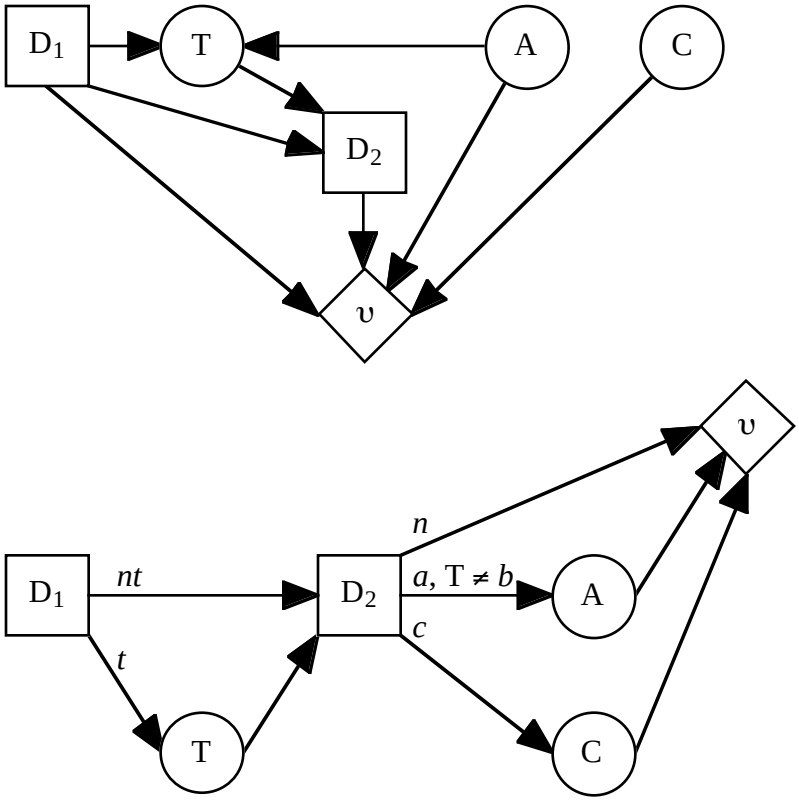
6 SEQUENTIAL DECISION DIAGRAMS (SDDs)

In this section, we will represent and solve the Reactor problem using Covaliu and Oliver's [1995] sequential decision diagram (SDD) technique. The SDD technique is described either for a problem in which the utility function is undecomposed, or for a problem in which the utility function decomposes into additive factors (or multiplicative) such that each factor has only one variable in its domain. Since our version of the Reactor problem is not in either of these two categories, first we combine the three utility factors and then we use the undecomposed version of the SDD technique to represent and solve the Reactor problem.

6.1 SDD Representation

In this technique, a decision problem is modeled at two levels, graphical and numerical. At the graphical level, we model a decision problem using two directed graphs—an ID to describe the probability model, and a new diagram, called a *sequential decision diagram* (SDD), which captures the asymmetric and the information constraints of the problem. Figure 6.1 shows an ID and a SDD for the Reactor problem. At the numerical level, we specify the conditionals for each node in the ID, and data built from both diagrams are organized in a *formulation table*, similar to the one used by Kirkwood [1993], in such a way that the recursive algorithm used in the solution process can easily access the data contained in it. Table 6.1 shows the formulation table for the Reactor problem.

Figure 6.1. The initial ID and the SDD for the Reactor problem.



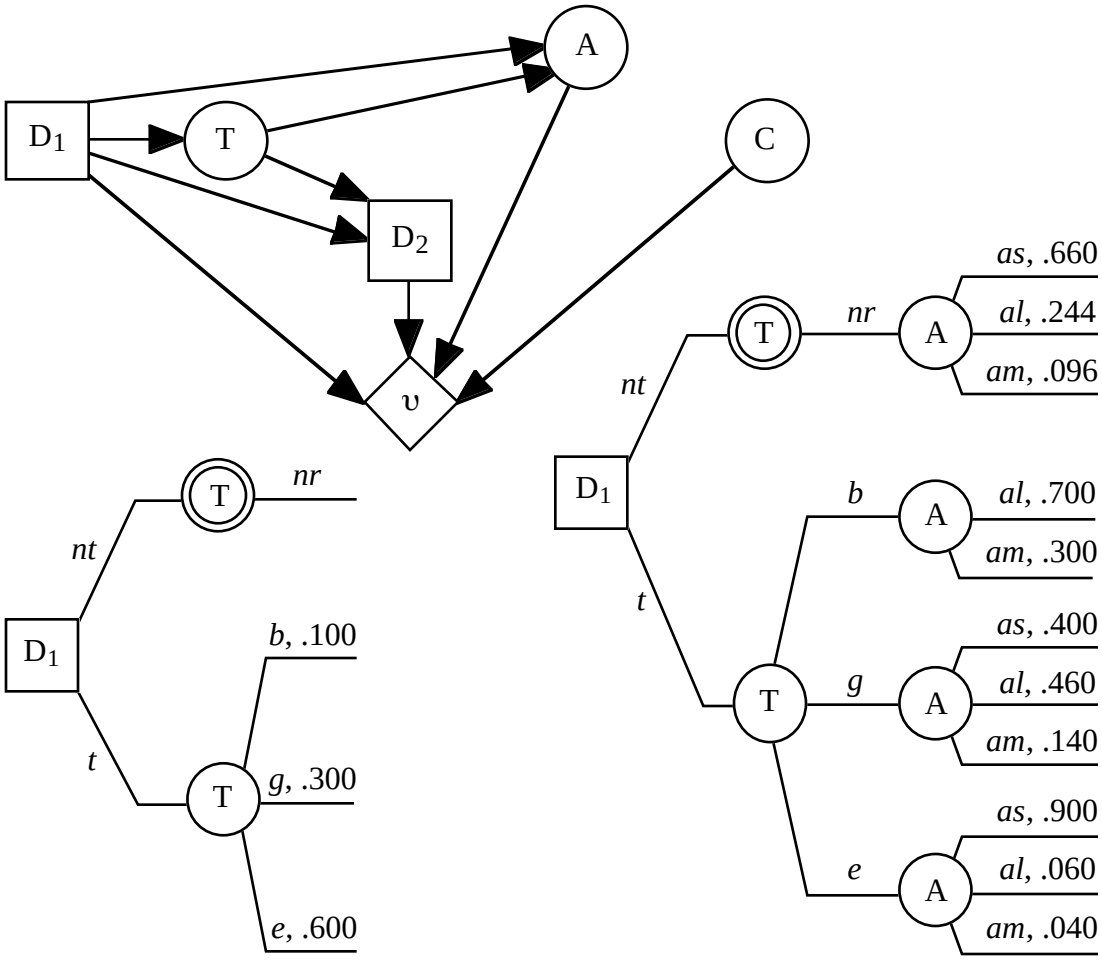
A SDD is a directed acyclic graph, with the same set of nodes as in the ID. However, its paths show all possible scenarios in a compact way, as if it were a clustered decision tree. A SDD is said to be *proper* if (i) there is only one source node (a node with no arrows pointing to it), (ii) there is only one sink node (a node with no arrows emanating from it) and it is the value node, and (iii) there is a directed path that contains all decision nodes.

In the SDD for the Reactor problem, the arc (D_1, T) with the label *t* tells us that if we perform the test (D_1 = *t*) then we will observe its result (T = *b*, *g* or *e*). Arc (D_1, D_2) with label *nt* tells us that we will not observe T when D_1 = *nt*. Arcs (D_2, υ), (D_2, C) and (D_2, A) show that A is relevant only if D_2 = *a*, and C is relevant only if D_2 = *c*. The label over the arc (D_2, A) also indicates dependence on realized states at predecessor nodes, i.e., the alternative D_2 = *a* is available only if T ≠ *b*. The six directed paths from D_1 to υ in the SDD are a compact representation of the twenty-one possible scenarios in the decision tree representation (Figure 3.3).

Notice that the partial order implied by the arrows in an ID may be different from the partial order implied by the arrows in a corresponding SDD. Let $<_D$ and $<_I$ denote the partial orders in SDD and ID respectively. If C is a chance node, D is a decision node, and C $<_I$ D implies C $<_D$ D, then we say the ID and SDD are *compatible* [Covaliu and Oliver 1995]. In Figure 6.1, e.g., we have A $<_I$ D_2 (since there is a directed path from A to D_2 in the ID), and D_2 $<_D$ A (since there is an arrow from D_2 to A in the SDD). Therefore the two diagrams are incompatible. The next step in

completing the SDD representation is to transform the ID so that it is compatible with the SDD. In the Reactor problem, we must reverse the arc (A, T) in the ID to make the ID compatible with the SDD. The transformed ID is shown in Figure 6.2.

Figure 6.2. The transformed ID and the conditionals for T and A.



Next, we organize data in the formulation table, which contains the complete information the solution algorithm will require. Table 6.1 is the formulation table for the Reactor problem.

Table 6.1. A formulation table for the Reactor problem.

Node Name	Node Type	Standard Histories (Minimal in bold)	State Space	Probability Distribution	Next-Node Function
D_1	decision	ø	*nt* *t*		D_2 T
T	chance	D_1			
		t	*b* *g* *e*	.1 .3 .6	D_2 D_2 D_2
D_2	decision	D_1 **T**			
		nt **-**	*n* *c* *a*		υ C A
		t ***b***	*n* *c*		υ C
		t ***g***	*n* *c* *a*		υ C A
		t ***e***	*n* *c* *a*		υ C A
A	chance	D_1 **T** D_2			
		nt **-** *a*	*as* *al* *am*	.660 .244 .096	υ υ υ
		t ***g*** *a*		.400 .460 .140	
		t ***e*** *a*		.900 .060 .040	
C	chance	D_1 T D_2			
		nt - *c*	*cs* *cf*	.98 .02	υ υ
		t *b* *c*			
		t *g* *c*			
		t *e* *c*			
υ	value	**D_1** T **D_2** **A** **C**			
		nt - ***n*** - -	0		
		nt - ***c*** - ***cs***	8		
		nt - ***c*** - ***cf***	–4		
		nt - ***a*** ***as*** -	12		
		nt - ***a*** ***al*** -	–6		
		nt - ***a*** ***am*** -	–10		
		t *b* ***n*** - -	–1		
		t *b* ***c*** - ***cs***	7		
		t *b* ***c*** - ***cf***	–5		
		t *g* ***n*** - -	–1		
		t *g* ***c*** - ***cs***	7		
		t *g* ***c*** - ***cf***	–5		
		t *g* ***a*** ***as*** -	11		
		t *g* ***a*** ***al*** -	–7		
		t *g* ***a*** ***am*** -	–11		
		t *e* ***n*** - -	–1		
		t *e* ***c*** - ***cs***	7		
		t *e* ***c*** - ***cf***	–5		
		t *e* ***a*** ***as*** -	11		
		t *e* ***a*** ***al*** -	–7		
		t *e* ***a*** ***am*** -	–11		

The formulation table has a row for each node in the SDD. If $X <_D Y$, then the row for X precedes the row for Y. Each row includes node name, node type, standard histories and minimal histories, state space, conditional distribution (for chance nodes only), and next-node function. It should be noted that the formulation table is part of the representation of the decision problem.

The term *history* refers to how one gets to a node through the directed paths in the SDD. It can be represented as a 2-row matrix, the first row listing a node sequence of all nodes that precede it in the partial order, and the second row listing the corresponding realized states. The next-node function (in the last column) denotes the node that is realized after a node for each of its states and for each minimal history. There are different kinds of histories. Minimal histories are always sufficient for defining node state spaces, probability distributions (for chance nodes), and next node functions. For a decision node, the minimal histories will include those variables that affect its state space, and its next-node function. For example, for D_2, variable T is the only one under these conditions. So, at node D_2 we have the minimal histories:

$$\begin{pmatrix} T \\ - \end{pmatrix}, \begin{pmatrix} T \\ b \end{pmatrix}, \begin{pmatrix} T \\ g \end{pmatrix}, \text{ and } \begin{pmatrix} T \\ e \end{pmatrix},$$

where - denotes the absence of T in a path to D_2, i.e., when $D_1 = nt$. For a chance node, the minimal histories will include the nodes that suffice for defining its next-node function, and its conditional probability distribution. For example, for C, the set of minimal histories is the empty set. For a value node, the minimal histories will include the nodes that suffice to define the values of the corresponding utility function and they are the direct predecessors of υ in the ID.

As we will see, minimal histories are not always sufficient to solve a decision problem. We need a new kind of history called relevant history. The node sets of relevant histories contain the node sets of minimal histories and are contained in the node sets of full histories. Also relevant histories can be computed from minimal and full histories. Therefore, we do not show relevant histories in the formulation, just full and minimal histories.

6.2 SDD Solution

Let $w_N(H_N) = E(u \mid H_N)$ denote the maximum expected utility at node N of the SDD given history H_N if optimal decisions are made at N (if N is a decision node) and from there onwards. Let $1(H_N)$ denote the set of nodes in H_N. The solution technique is based on the same backwards recursive relations used in decision trees, but here we use a new kind of history called relevant history. We cannot use only minimal histories because, when calculating $w_N(H_N)$, we may reference the next nodes n_N and their histories H_{n_N}, and $w_N(H_N)$ is not well defined if there exists at least one n_N such that $1(H_{n_N}) - \{N\} \not\subset 1(H_N)$. We obtain the node sets in the relevant histories by enlarging the node sets in minimal histories by those SDD predecessors that appear in the node sets of relevant histories of any of the direct successors nodes. Covaliu and Oliver [1995] give a recursive defini-

tion of this term. The solution algorithm then follows a backward recursive method. The solution of the Reactor problem is as follows.

w_υ is the utility function associated with node υ in the formulation table, e.g.,

$w_\upsilon\left(\begin{matrix} D_1 & D_2 & C & A \\ nt & n & - & - \end{matrix}\right) =$ 0, etc.

First, we reduce node C. The relevant history for node C includes nodes D_1 and D_2 (since C's minimal history node set is ø, C's successor is node υ, υ's minimal history node set is $\{D_1, D_2, C, A\}$, and the set of predecessors of C is $\{D_1, T, D_2\}$).

$$\begin{aligned} w_C\left(\begin{matrix} D_1 & D_2 \\ nt & c \end{matrix}\right) &= w_\upsilon\left(\begin{matrix} D_1 & D_2 & C \\ nt & c & cs \end{matrix}\right)(.98) + w_\upsilon\left(\begin{matrix} D_1 & D_2 & C \\ nt & c & cf \end{matrix}\right)(.02) \\ &= w_\upsilon\left(\begin{matrix} D_1 & D_2 & C & A \\ nt & c & cs & - \end{matrix}\right)(.98) + w_\upsilon\left(\begin{matrix} D_1 & D_2 & C & A \\ nt & c & cf & - \end{matrix}\right)(.02) \\ &= (8)(.98) + (-4)(.02) \\ &= 7.760. \end{aligned}$$

In the computation of $w_C\left(\begin{matrix} D_1 & D_2 \\ nt & c \end{matrix}\right)$ above, notice that, e.g., strictly speaking $w_\upsilon\left(\begin{matrix} D_1 & D_2 & C \\ nt & c & cs \end{matrix}\right)$ is not defined. However, the history $\left(\begin{matrix} D_1 & D_2 & C \\ nt & c & cs \end{matrix}\right)$ differs from the history $\left(\begin{matrix} D_1 & D_2 & C & A \\ nt & c & cs & - \end{matrix}\right)$ only in the variable A that has state -, i.e., $\left(\begin{matrix} D_1 & D_2 & C & A \\ nt & c & cs & - \end{matrix}\right)$ is simply a standardized version of $\left(\begin{matrix} D_1 & D_2 & C \\ nt & c & cs \end{matrix}\right)$. Therefore, $w_\upsilon\left(\begin{matrix} D_1 & D_2 & C \\ nt & c & cs \end{matrix}\right) = w_\upsilon\left(\begin{matrix} D_1 & D_2 & C & A \\ nt & c & cs & - \end{matrix}\right)$, and the RHS of this equality is well defined.

Similarly,

$$w_C\left(\begin{matrix} D_1 & D_2 \\ t & c \end{matrix}\right) = 6.76.$$

Next, we reduce node A. The relevant histories for node A include D_1, T, and D_2 (since A's minimal history node set is {T}, A's successor is node υ, υ's relevant history node set is $\{D_1, D_2, C, A\}$, and the set of predecessors of A is $\{D_1, T, D_2\}$).

$$\begin{aligned} w_A\left(\begin{matrix} D_1 & T & D_2 \\ nt & - & a \end{matrix}\right) &= w_\upsilon\left(\begin{matrix} D_1 & T & D_2 & A \\ nt & - & a & as \end{matrix}\right)(.660) + w_\upsilon\left(\begin{matrix} D_1 & T & D_2 & A \\ nt & - & a & al \end{matrix}\right)(.244) + \\ & \qquad w_\upsilon\left(\begin{matrix} D_1 & T & D_2 & A \\ nt & - & a & am \end{matrix}\right)(.096) \end{aligned}$$

$$= w_{\upsilon}\left(\begin{matrix} D_1 & D_2 & C & A \\ nt & a & - & as \end{matrix}\right)(.660) + w_{\upsilon}\left(\begin{matrix} D_1 & D_2 & C & A \\ nt & a & - & al \end{matrix}\right)(.244) +$$

$$w_{\upsilon}\left(\begin{matrix} D_1 & D_2 & C & A \\ nt & a & - & am \end{matrix}\right)(.096)$$

$$= (12)(.660) + (-6)(.244) + (-10)(.096)$$

$$= 5.496.$$

Similarly,

$$w_A\left(\begin{matrix} D_1 & T & D_2 \\ t & g & a \end{matrix}\right) = -0.351; \text{ and}$$

$$w_A\left(\begin{matrix} D_1 & T & D_2 \\ t & e & a \end{matrix}\right) = 9.043.$$

Next we reduce node D_2. Its relevant histories include variables D_1, and T.

$$w_{D_2}\left(\begin{matrix} D_1 & T \\ nt & - \end{matrix}\right) = \text{MAX}\{w_{\upsilon}\left(\begin{matrix} D_1 & T & D_2 \\ nt & - & n \end{matrix}\right),$$

$$w_C\left(\begin{matrix} D_1 & T & D_2 \\ nt & - & c \end{matrix}\right), w_A\left(\begin{matrix} D_1 & T & D_2 \\ nt & - & a \end{matrix}\right)\}$$

$$= \text{MAX}\{0, 7.760, 5.496\}$$

$$= 7.760\ [c].$$

Optimal alternatives are indicated in square brackets adjacent to the maximum utility value. Similarly,

$$w_{D_2}\left(\begin{matrix} D_1 & T \\ t & b \end{matrix}\right) = 6.760\ [c];$$

$$w_{D_2}\left(\begin{matrix} D_1 & T \\ t & g \end{matrix}\right) = 6.760\ [c]; \text{ and}$$

$$w_{D_2}\left(\begin{matrix} D_1 & T \\ t & e \end{matrix}\right) = 9.043\ [a].$$

Next, we reduce T. The relevant histories for T include D_1.

$$w_T\left(\begin{matrix} D_1 \\ t \end{matrix}\right) = w_{D_2}\left(\begin{matrix} D_1 & T \\ t & b \end{matrix}\right)(.1) + w_{D_2}\left(\begin{matrix} D_1 & T \\ t & g \end{matrix}\right)(.3) + w_{D_2}\left(\begin{matrix} D_1 & T \\ t & e \end{matrix}\right)(.6)$$

$$= (6.760)(.1) + (6.760)(.3) + (9.043)(.6)$$

$$= 8.130.$$

Finally, we reduce D_1, whose set of relevant histories is empty.

$$w_{D_1} = \text{MAX}\{w_{D_2}\left(\begin{matrix} D_1 \\ nt \end{matrix}\right), w_T\left(\begin{matrix} D_1 \\ t \end{matrix}\right)\}$$

$$= \text{MAX}\ \{w_{D_2}(\begin{pmatrix} D_1 & T \\ nt & - \end{pmatrix}), w_T(\begin{pmatrix} D_1 \\ t \end{pmatrix})\}$$

$$= \text{MAX}\{7.760,\ 8.130\}$$

$$= 8.130\ [t].$$

This completes the solution of the Reactor problem using the SDD technique. Regarding the number of operations, aggregating the utility factors into the joint utility function requires 24 operations, reversing arc (A, T) requires 21 operations, reducing C requires 6 operations, reducing A requires 15 operations, reducing D_2 requires 7 operations, reducing T requires 5 operations, and reducing D_1 requires 1 operation, for a total of 79 operations.

7 SUMMARY OF STRENGTHS, WEAKNESSES AND OPEN ISSUES

In this section, we summarize the strengths, weaknesses, and open issues of each technique. For details, see the Management Science paper.

7.1 Decision Trees

Strengths

- Easy to understand and solve
- Encoding of asymmetry without introducing dummy states for variables

Weaknesses

- Global representation of asymmetry
- Exponential growth of representation
- Automation of coalescence is difficult
- Decision tree graph encodes a complete ordering of variables even though information constraints impose a partial ordering

Open Issues

- Avoiding preprocessing using information sets
- Making preprocessing more efficient by using Bayesian networks

7.2 Asymmetric Influence Diagrams

Strengths

- Compact, intuitive, encode conditional independence relations
- Encoding of asymmetry by distribution trees which are easy to understand and specify
- Collapsed scenarios, clipped scenarios, sharing of scenarios, and unspecified distributions features of distribution trees can represent many different kinds of asymmetries

- Distribution trees can mix different types of atomic distributions
- Can detect and eliminate barren nodes

Weaknesses

- Only suitable for problems in which we have a Bayesian network model for the uncertainties
- Encoding of asymmetry by introducing dummy states for variables

Open Issues

- Distinguishing between pure informational and conditioning arcs for decision variables
- Repetition of asymmetry information in distribution trees
- Non-availability of all asymmetry information during the node-reduction process

7.3 Valuation Networks

Strengths

- Compact, encode conditional independence relations
- Can represent all probability models without any preprocessing
- Encoding of information constraints more flexible than IDs
- Encoding of asymmetry using indicator valuations which are easy to understand and specify
- Domains of probability functions are typically smaller than conditionals in IDs
- Avoidance of unnecessary divisions in the solution process
- Existence of conditions for when a representation is well defined

Weaknesses

- Modeling of conditionals is not as intuitive as in IDs
- Specification of VN representation must be done sequentially in stages—first graphical, next dependence, and finally numerical
- Graphical representation has more nodes than IDs
- Encoding of asymmetry by introducing dummy states for variables

Open Issues

- Not all asymmetry can be represented
- Unable to use sharing of scenarios and collapsed scenarios features of IDs

7.4 Sequential Decision Diagrams

Strengths

- Compact representation, as intuitive as DTs
- Encoding of asymmetry without introducing dummy states for variables
- Can exploit coalescence by using minimal and relevant histories
- Can detect and eliminate barren nodes

Weaknesses

- Unable to represent probability models consistently
- May require preprocessing in constructing formulation tables
- Formulation table encodes a complete ordering of variables even though information constraints impose a partial ordering

Open Issues

- Next node function is too restrictive
- Need for an efficient method to compute minimal and relevant histories without enumerating all histories
- Redundancy in the representation of IDs, SDDs, and formulation tables
- Unable to represent arbitrary factorization of the utility function

8 CONCLUSIONS

The main goal of this work is to compare four distinct techniques proposed for representing and solving asymmetric decision problems—traditional decision trees, SHM influence diagrams, Shenoy's valuation networks, and Covaliu and Oliver's sequential decision diagrams. For each technique, we have identified the main strengths, intrinsic weaknesses, and some open issues that perhaps can be resolved with further research.

One conclusion is that no single technique stands out as always superior in all respects to the others. Each technique has some unmatched strengths. Another conclusion is that considerable work remains to be done to resolve the open issues of each technique. One possibility here is to borrow the strengths of a technique to resolve the issues of another. Also, there is need for automating each technique by building computer implementations, and there is very little literature on this topic.

At the end of Sections 3-6, we provide a summary of the total number of arithmetic operations required to solve the Reactor problem. Since we solve only one problem, it is incorrect to conclude that the technique with the lowest total is the most efficient. It would be more interesting to study how this number grows in general as a function of the dimension of the problem. But this depends on the structure of the problem. We conjecture that we can concoct examples where each technique would have the least number of total arithmetic operations. Also, it is obvious that the efficiency of each technique cannot be measured by only counting the number of arithmetic operations. There are other features to take into account. For example, in the formulation phase:

- defining the variables and their state spaces to reflect asymmetry;
- (possible) preprocessing for representing the probabilities in DTs and SDDs;
- specification of distribution trees in IDs;

- computation of the effective state spaces prior to the specification of probability valuations in VNs;
- computation of minimal histories node sets, minimal histories, relevant history node sets, and relevant histories in SDDs;

and in the solution phase:

- number of table look ups necessary to remove each node and the number of configurations used for reduction of a node;
- computation of effective state spaces in the solution stage for VNs; and
- the cardinality of the largest state space used during the solution phase.

The actual computational efficiency will depend very much on the details of the computer implementation of the different techniques.

In trying to represent asymmetries, one introduces some additional costs, and in some cases, the costs may exceed the actual benefits. For example, one can easily make up a large "asymmetric" problem in which very few scenarios are eliminated. And in this case, the cost of representing and processing the asymmetries may well exceed the benefits. This is the case for IDs, VNs, and SDDs.

ACKNOWLEDGMENTS

Most of the details of the solutions of the four representations provided in this working paper were taken from Bielza and Shenoy [1996]. We are grateful to Jim Smith for extensive discussions about the SHM ID technique that led to several changes from earlier drafts, to Zvi Covaliu for discussions about the SDD technique, and to David Ríos Insua for general comments and discussions. Of course, any errors in the paper are solely our responsibility.

REFERENCES

Bielza, C. and P. P. Shenoy (1996), "A comparison of graphical techniques for asymmetric decision problems," Working Paper No. 271, School of Business, University of Kansas, Lawrence, KS.

Covaliu, Z. and R. M. Oliver (1995), "Representation and solution of decision problems using sequential decision diagrams," *Management Science*, **41**(12), 1860–1881.

Kirkwood, C. W. (1993), "An algebraic approach to formulating and solving large models for sequential decisions under uncertainty," *Management Science*, **39**(7), 900–913.

Olmsted, S. M. (1983), "On representing and solving decision problems," Ph.D. thesis, Department of Engineering-Economic Systems, Stanford University, Stanford, CA.

Shenoy, P. P. (1992), "Valuation-based systems for Bayesian decision analysis," *Operations Research*, **40**(3), 463–484.

Shenoy, P. P. (1993a), "A new method for representing and solving Bayesian decision problems," in D. J. Hand (eds.), *Artificial Intelligence Frontiers in Statistics: AI and Statistics III*, 119–138, Chapman & Hall, London.

Shenoy, P. P. (1993b), "Valuation network representation and solution of asymmetric decision problems," Working Paper No. 246, School of Business, University of Kansas, Lawrence, KS. To appear in *European Journal of Operational Research* in 1999. Available by anonymous ftp from <ftp.bschool.ukans.edu/home/pshenoy/wp246.ps>.

Shenoy, P. P. (1994), "Consistency in valuation-based systems," *ORSA Journal on Computing*, **6**(3), 281–291.

Shenoy, P. P. (1996), "Representing and solving asymmetric decision problems using valuation networks," in D. Fisher and H.-J. Lenz (eds.), *Learning from Data: Artificial Intelligence and Statistics V*, Lecture Notes in Statistics, **112**, 99–108, Springer-Verlag, New York, NY.

Shenoy, P. P. and G. Shafer (1990), "Axioms for probability and belief-function propagation," in R. D. Shachter, T. S. Levitt, J. F. Lemmer and L. N. Kanal (eds.), *Uncertainty in Artificial Intelligence*, **4**, 169–198, North-Holland, Amsterdam.

Smith, J. E., S. Holtzman and J. E. Matheson (1993), "Structuring conditional relationships in influence diagrams," *Operations Research*, **41**(2), 280–297.

Tatman, J. A. and R. D. Shachter (1990), "Dynamic programming and influence diagrams," *IEEE Transactions on Systems, Man, and Cybernetics*, **20**(2), 365–379.