ORSA Journal on Computing

Publication details, including instructions for authors and subscription information:
http://pubsonline.informs.org

Consistency in Valuation-Based Systems

Prakash P. Shenoy,

To cite this article:
Prakash P. Shenoy, (1994) Consistency in Valuation-Based Systems. ORSA Journal on Computing 6(3):281-291. http://dx.doi.org/10.1287/ijoc.6.3.281

ORSA Journal on Computing
Vol 6, No. 3, Summer 1994

0899-1499/94/0603-0281 $01.25

Consistency in Valuation-Based Systems

PRAKASH P. SHENOY / *School of Business, University of Kansas, Summerfield Hall, Lawrence, KS 66045-2003; BITNET: pshenoy@ukanvm*

(Received: March 1990; revised: August 1991; accepted: June 1992)

This paper has three main results. First, we present a new computational technique for checking for inconsistencies in valuation-based systems. This technique is different from the implicit enumeration method of Davis-Putnam and its variants. Our technique uses the divide-and-conquer method of dynamic programming. The computational complexity of this technique depends on the sizes of the valuations and on the graphical structure of the valuation-based system. Second, if a valuation-based system is consistent, we describe a method for generating a model for the system, i.e., an assignment of values for each variable that is consistent with each valuation in the system. Third, if a valuation-based system is inconsistent, we describe a method for isolating a minimal inconsistent set of valuations.

A valuation-based system is a knowledge-based system in which knowledge is represented by functions called valuations.[16 22] For example, a rule-based system can be represented as a valuation-based system.

This paper presents a new computational technique for checking for inconsistencies in valuation-based systems. We use the word inconsistency in the logical sense—a set of propositions is inconsistent if contradiction is entailed. This technique is capable of detecting all contradictions that can be detected using the expressive power of propositional logic. The computational complexity of this technique depends on the sizes of the valuations and on the graphical structure of the valuation-based system.

The problem of consistency is NP-complete in the worst case.[3] However, this does not mean that real-world valuation-based systems always conform to the worst case. In such cases, it is desirable to have an efficient method for detecting inconsistencies. In this paper, we describe a fusion algorithm for checking for inconsistency that uses only local computation. The computational complexity of the fusion algorithm is $O(n^2k + (n + k)s)$, where n is the number of variables, k is the number of valuations, and s is the cardinality of the frame of the largest subset of variables on which combination is done in the fusion algorithm.

The issue of consistency in valuation-based systems is an important one. Most commercial rule-based languages do not check whether the rule-base is consistent or not. For example, in Texas Instruments' PERSONAL CONSULTANT system, if two rules such as *If* $X = x$ *then* $Y = y$, and *If* $X = x$ *then* $Y = \sim y$ are entered with variable Y as the goal, the system will first ask the user for the value of X. If the user responds by stating that $X = x$, the system then responds by concluding either $Y = y$ or $Y = \sim y$, depending on the order in which the rules were entered in the rule-base.

The method we propose is distinct from previous approaches to this problem. Suwa et al.[26] and Nguyen et al.[14] restrict their analyses to pairs of rules with contradictory conclusions. As Ginsberg[9] has pointed out, such pairwise comparisons cannot guarantee detection of all potential inconsistencies. Ginsberg's[9] method, called "knowledge-base reduction," is based on techniques developed by de Kleer[5] for assumption-based truth maintenance systems. His method flags all potential inconsistencies. The task we address is a bit simpler. We do not attempt to detect potential inconsistencies. The method we present detects actual inconsistencies. Thus, two rules such as *If* $X = x$ *then* $Y = y$, and *If* $X = x$ *then* $Y = \sim y$ are not inconsistent until $X = x$ is posited.

The problem of consistency we address is the same as the satisfiability problem in propositional logic. Several fast algorithms have been developed for this problem (see, for example, [4, 12, 8, 10]). In fact, if all knowledge can be represented as propositional Horn formulae, then consistency checking can be done in linear time.[6] All of these methods are based on the branch-and-bound technique of integer programming. In comparison, the method we propose is based on the divide-and-conquer technique of dynamic programming. We exploit the fact that knowledge can be broken down into small formulae connected together, for example, by logical conjunction. This is the only fact that is exploited in our method. As such, our method is not as fast as the branch-and-bound algorithms mentioned earlier. However, our method is more general. It applies not only to Boolean propositions, but also to multiple-valued variables. Also, our method has several by-products not obtained from these other methods.

If a valuation-based system is consistent, then it is useful to have a configuration of all variables that is consistent with each valuation in the system. We call such a configuration a "model" for the system. One of the by-products of the fusion algorithm is that we can generate a model for a

Subject classifications: Computers/computer science. Artificial intelligence; Information systems. Expert systems.
Other key words: Valuation-based systems, consistency in rule-based systems, satisfiability problem, propositional logic, theorem proving, knowledge representation.

consistent valuation-based system for a small additional cost.

If a valuation-based system is inconsistent, then it is useful to isolate a minimal set of valuations that are inconsistent. This will help a knowledge engineer to modify the knowledge-base so as to make it consistent. We propose a method for isolating a minimal set of inconsistent valuations. This method is a minor extension of the fusion algorithm for detecting inconsistencies.

All three methods described in this paper are based on the framework of valuation-based systems.[16, 22] This framework is very general and applies to many domains such as probability theory,[25, 22] Dempster-Shafer theory of belief functions,[23, 22] Spohn's theory of epistemic beliefs,[17, 22] possibility theory,[7, 19] discrete optimization,[18] constraint satisfaction,[24] and Bayesian decision theory.[20, 21]

An outline of this manuscript is as follows. In section 1, we define valuations and proper valuations which are used to represent knowledge. In section 2, we define two operations on valuations called combination and marginalization. These operations are used to make inferences from the knowledge-base. In section 3, we give a formal definition of a consistent valuation-based system. In section 4, we describe a fusion algorithm for checking for inconsistency in a valuation-based system and describe its computational complexity. In section 5, we describe a method for generating a model for consistent valuation-based systems. In section 6, we describe a method for identifying a minimal set of inconsistent valuations in an inconsistent valuation-based system. In section 7, we make some concluding remarks. Finally, in section 8, we provide proofs for the three main theorems in the paper.

1. Knowledge Representation

We represent knowledge by functions, called valuations, from the space of configurations to the space of values.

Consider a variable X. We use the symbol $\mathscr{W}_X$ for the set of possible values of X. We assume that one and only one of the elements of $\mathscr{W}_X$ can be the true value of X. We call $\mathscr{W}_X$ the *frame for* X. For example, suppose we are interested in determining whether Dick is a crook or not. We construct a variable IS_DICK_A_CROOK whose frame has two elements: "yes" and "no."

We assume $\mathscr{W}_X$ is defined such that the propositions regarding X that are of interest are precisely those of the form "The true value of X is in A," where A is a subset of $\mathscr{W}_X$. Thus, the propositions regarding X that are of interest are in a one-to-one correspondence with the subsets of $\mathscr{W}_X$; see p. 36 of [15].

The correspondence between subsets and propositions is useful since it translates the logical notions of conjunction, disjunction, implication and negation into the set-theoretic notions of intersection, union, inclusion and complementation, respectively; see pp. 36–37 of [15]. Thus, if A and B are two subsets of $\mathscr{W}_X$, and A' and B' are the corresponding propositions, then $A \cap B$ corresponds to the conjunction of A' and B', $A \cup B$ corresponds to the disjunction of A' and B', $A \subseteq B$ if and only if A' implies B', and A is the set-theoretic complement of B with respect to $\mathscr{W}_X$ if and only if A' is the negation of B'. Notice also that the proposition that corresponds to $\emptyset$ is false and the proposition that corresponds to $\mathscr{W}_X$ is true.

Let $\mathscr{X}$ denote the set of all variables. In this paper, we are concerned only with the case where $\mathscr{X}$ is finite. Also, we assume that all the variables in $\mathscr{X}$ have finite frames.

We often deal with non-empty subsets of variables in $\mathscr{X}$. Given a non-empty subset h of $\mathscr{X}$, let $\mathscr{W}_h$ denote the Cartesian product of $\mathscr{W}_X$ for X in h, i.e., $\mathscr{W}_h = X\{\mathscr{W}_X | X \in h\}$. We can think of the set $\mathscr{W}_h$ as the set of possible values of the joint variable h. Accordingly, we call $\mathscr{W}_h$ the *frame for* h. Also, we call elements of $\mathscr{W}_h$ *configurations of* h. We use this terminology even when h consists of a single variable, say X. Thus we call elements of $\mathscr{W}_X$ *configurations of* X. We use lower-case, bold-faced letters such as $\mathbf{x}$, $\mathbf{y}$, etc. to denote configurations. Also, if $\mathbf{x}$ is a configuration of g, and $\mathbf{y}$ is a configuration of h, and $g \cap h = \emptyset$, then $(\mathbf{x}, \mathbf{y})$ denotes a configuration of $g \cup h$.

It is convenient to extend this terminology to the case where the set of variables h is empty. We adopt the convention that the frame for the empty set $\emptyset$ consists of a single configuration, and we use the symbol $\blacklozenge$ to name that configuration; $\mathscr{W}_\emptyset = \{\blacklozenge\}$. To be consistent with our notation above, we adopt the convention that if $\mathbf{x}$ is a configuration for g, then $(\mathbf{x}, \blacklozenge) = \mathbf{x}$.

In (categorical) rule-based systems, knowledge is represented by rules. For example, we may have a rule relating two variables X and Y: *If* $X = x$ *then* $Y = y$. In our framework, we represent knowledge using functions called valuations.

Suppose $h \subseteq \mathscr{X}$. A *valuation for* h is a function $H: \mathscr{W}_h \to \{1, 0\}$. We call the elements of the set $\{1, 0\}$ *values*. The value 1 represents true, and the value 0 represents false. Thus, a valuation for h is a function from the set of configurations of h to the set of values. If H is a valuation for h and $X \in h$, then we say H *bears on* X. Also, if H is a valuation for h, then we say the *domain of* H *is* h.

Consider the rule *If* $X = x$ *then* $Y = y$ that relates two variables X and Y whose frames are, respectively, $\mathscr{W}_X = \{x, \sim x\}$ and $\mathscr{W}_Y = \{y, \sim y\}$. This rule can be represented by the valuation V for $\{X, Y\}$ defined as follows: $V(x, y) = 1$, $V(x, \sim y) = 0$, $V(\sim x, y) = 1$, $V(\sim x, \sim y) = 1$. Note that this is basically the truth-table representation of the rule interpreted as a conditional.

In order to represent the notion of consistent knowledge, we define proper valuations. Suppose $h \subseteq \mathscr{X}$. A valuation H for h is said to be *proper* if there exists a configuration $\mathbf{x}$ of h such that $H(\mathbf{x}) = 1$, and *improper* otherwise. Thus a proper valuation cannot be identically equal to 0 for all configurations.

The motivation behind the above definition is clear. Earlier, we had defined a frame as a set of configurations exactly one of which is true. Thus, it should not be possible for a consistent piece of knowledge to rule out all configurations.

2. Combination and Marginalization

In this section, we define two basic operations for valuations called combination and marginalization. Combination corresponds to logical conjunction and represents aggrega-

tion of knowledge. Marginalization corresponds to logical disjunction and represents coarsening of knowledge.

Before we define the two operations, we need to introduce some notation. *Projection* of configurations simply means dropping extra coordinates; if (w, x, y, z) is a configuration of $\{W, X, Y, Z\}$, for example, then the projection of (w, x, y, z) to $\{W, X\}$ is simply (w, x), which is a configuration of $\{W, X\}$. Formally, if g and h are sets of variables, $h \subseteq g$, and $\mathbf{x}$ is a configuration of g, then we let $\mathbf{x}^{\downarrow h}$ denote the projection of $\mathbf{x}$ to h. The projection $\mathbf{x}^{\downarrow h}$ is always a configuration of h. If $h = \varnothing$, then of course $\mathbf{x}^{\downarrow h} = \blacklozenge$.

Combination

Suppose G and H are valuations for g and h, respectively. The *combination of G and H*, denoted by $G \otimes H$, is the valuation for $g \cup h$ defined as follows:

$$(G \otimes H)(\mathbf{x}) = G(\mathbf{x}^{\downarrow g})H(\mathbf{x}^{\downarrow h}) \tag{2.1}$$

for all $\mathbf{x} \in \mathscr{W}_{g \cup h}$.

Suppose $\mathbf{x} \in \mathscr{W}_{g \cup h}$. Then $\mathbf{x}$ can be regarded as a proposition about variables in $g \cup h$. Note that proposition $\mathbf{x}$ is the same as the logical conjunction of propositions $\mathbf{x}^{\downarrow g}$ and $\mathbf{x}^{\downarrow h}$. Since the truth table for logical conjunction is simply pointwise multiplication, this explains the definition in (2.1).

Intuitively, combination corresponds to aggregation of knowledge. The valuation $G \otimes H$ represents the aggregation of the knowledge in G and H. Note that if either G or H is improper, then $G \otimes H$ is improper. However, if both G and H are proper, then $G \otimes H$ may be proper or improper. Whether $G \otimes H$ is proper or not depends on if the knowledge represented by G and H are consistent or not when aggregated.

Note that the combination operator is commutative and associative; $G \otimes H = H \otimes G$, and $(G \otimes H) \otimes K = G \otimes (H \otimes K)$. Thus, when we combine several valuations $G_1, \ldots, G_k$, we can write $G_1 \otimes \cdots \otimes G_k$ or simply $\otimes\{G_i \mid i = 1, \ldots, k\}$ without indicating the order in which the combination is carried out.

Consider two variables X and Y whose frames are, respectively, $\mathscr{W}_X = \{x, \sim x\}$ and $\mathscr{W}_Y = \{y, \sim y\}$. Let G_1, G_2, and G_3 be three valuations for $\{X\}$, $\{X, Y\}$, and $\{X, Y\}$, respectively, given as follows:

$$G_1(x) = 1, G_1(\sim x) = 0;$$

$$G_2(x, y) = 1, G_2(x, \sim y) = 0, G_2(\sim x, y) = 1, G_2(\sim x, \sim y) = 1;$$

$$G_3(x, y) = 0, G_3(x, \sim y) = 1, G_3(\sim x, y) = 1, G_3(\sim x, \sim y) = 1.$$

G_1 represents the proposition $X = x$; G_2 represents the conditional *If* $X = x$ *then* $Y = y$; and G_3 represents the conditional *If* $X = x$ *then* $Y = \sim y$. Table I shows that $G_1 \otimes G_2 \otimes G_3$ is an improper valuation. Therefore, the set of valuations $\{G_1, G_2, G_3\}$ is inconsistent.

Marginalization

Suppose G is a valuation for g, and suppose $h \subseteq g$. Then the *marginal of G for h*, denoted by $G^{\downarrow h}$, is the valuation for h defined as follows:

$$G^{\downarrow h}(\mathbf{x}) = \mathrm{MAX}\{G(\mathbf{x}, \mathbf{y}) \mid \mathbf{y} \in \mathscr{W}_{g-h}\} \tag{2.2}$$

for all $\mathbf{x} \in \mathscr{W}_h$.

Suppose $\mathbf{x} \in \mathscr{W}_h$. Then $\mathbf{x}$ can be regarded as a proposition about variables in h. Note that proposition $\mathbf{x}$ is the same as the disjunction of propositions in the set $\{(\mathbf{x}, \mathbf{y}) \mid \mathbf{y} \in \mathscr{W}_{g-h}\}$. Since the truth table for disjunction is simply maximization, this explains the definition in (2.2).

Intuitively, marginalization corresponds to coarsening of knowledge. If G is a valuation for g representing some knowledge about variables in g, and $h \subseteq g$, then $G^{\downarrow h}$ represents knowledge about variables in h implied by G if we disregard variables in $g - h$.

It follows from the above definition that $G^{\downarrow h}$ is proper if and only if G is proper. Also, if $X_1, X_2 \in g$, and G is a valuation on g, then $(G^{\downarrow(g-\{X_1\})})^{\downarrow(g-\{X_1, X_2\})} = (G^{\downarrow(g-\{X_2\})})^{\downarrow(g-\{X_1, X_2\})}$. In words, the order of deletion of variables in the marginalization operation does not matter.

Consider the valuations G_1 and G_2 as defined above representing proposition $X = x$, and conditional *If* $X = x$ *then* $Y = y$, respectively. If we combine G_1 and G_2 and marginalize the combination for $\{Y\}$, the resulting valuation $(G_1 \otimes G_2)^{\downarrow\{Y\}}$ is given by $(G_1 \otimes G_2)^{\downarrow\{Y\}}(y) = 1$ and $(G_1 \otimes G_2)^{\downarrow\{Y\}}(\sim y) = 0$, i.e., $(G_1 \otimes G_2)^{\downarrow\{Y\}}$ represents the proposition $Y = y$ (see Table II). Thus, combination and marginalization give the same result as the *modus ponens* form of inference in propositional logic ($X = x$, and *If* $X = x$ *then* $Y = y$, $\therefore Y = y$).

Similarly, other forms of logical inference such as *modus tollens* ($Y = \sim y$, and *If* $X = x$ *then* $Y = y$, $\therefore X = \sim x$), *disjunctive syllogism* ($X = x$ *or* $Y = y$, and $X = \sim x$, $\therefore Y = y$), and *disjunctive elimination* ($X = x$ *or* $Y = y$, *If* $X = x$ *then* $Z = z$, and *If* $Y = y$ *then* $Z = z$, $\therefore Z = z$) are shown to be

Table I. The Combination of G_1, G_2, and G_3

w		$G_1(\mathbf{w}^{\downarrow\{X\}})$	$G_2(\mathbf{w})$	$G_3(\mathbf{w})$	$(G_1 \otimes G_2 \otimes G_3)(\mathbf{w})$
x	y	1	1	0	0
x	$\sim y$	1	0	1	0
$\sim x$	y	0	1	1	0
$\sim x$	$\sim y$	0	1	1	0

Table II. Modus Ponens Represented as Combination and Marginalization

$\mathscr{W}_{\{Y, X\}}$		G_1	G_2	$G_1 \otimes G_2$	$(G_1 \otimes G_2)^{\downarrow\{Y\}}$	Ψ_X
y	x	1	1	1	1	x
y	$\sim x$	0	1	0	-	-
$\sim y$	x	1	0	0	0	x
$\sim y$	$\sim x$	0	1	0	-	-

G_1 represents the proposition $X = x$, and G_2 represents the conditional *If* $X = x$ *then* $Y = y$. $(G_1 \otimes G_2)^{\downarrow\{Y\}}$ represents the conclusion $Y = y$. Ψ_X is a solution for X with respect to $G_1 \otimes G_2$.

Table III. Modus Tollens Represented as Combination and Marginalization

$\mathscr{W}_{\{X,Y\}}$		G_4	G_2	$G_4 \otimes G_2$	$(G_4 \otimes G_2)^{\downarrow\{X\}}$
x	y	0	1	0	0
x	$\sim y$	1	0	0	-
$\sim x$	y	0	1	0	1
$\sim x$	$\sim y$	1	1	1	-

G_4 represents the proposition $Y = \sim y$, and G_2 represents the conditional *If* $X = x$ *then* $Y = y$. $(G_4 \otimes G_2)^{\downarrow\{X\}}$ represents the conclusion $X = \sim x$.

Table IV. Disjunctive Syllogism Represented as Combination and Marginalization

$\mathscr{W}_{\{Y,X\}}$		G_5	G_6	$G_5 \otimes G_6$	$(G_5 \otimes G_6)^{\downarrow\{Y\}}$
y	x	0	1	0	1
y	$\sim x$	1	1	1	-
$\sim y$	x	0	1	0	0
$\sim y$	$\sim x$	1	0	0	-

G_5 represents the proposition $X = \sim x$, and G_6 represents the disjunction $X = x$ *or* $Y = y$. $(G_5 \otimes G_6)^{\downarrow\{Y\}}$ represents the conclusion $Y = y$.

special cases of combination and marginalization in Tables III, IV, and V, respectively.

In consistent valuation-based systems, it is useful to have a model, i.e., a configuration of all variables that is consistent with all valuations in the system. This motivates the following definition.

Model for a Valuation

Suppose G is a proper valuation for g. We call $\mathbf{x} \in \mathscr{W}_g$ *a model for* G if $G(\mathbf{x}) = 1$.

Solution for a Variable

As we will see, once we have shown that a VBS is consistent, generating a model for the VBS is a matter of bookkeeping. Each time we marginalize a variable out of a valuation using maximization, we store a table of values of the variable where the maximums are achieved. We can think of this table as a function. We call this function " a solution for the variable." Formally, we define a solution as follows. Suppose g is a subset of variables, suppose $X \in g$, and suppose G is a valuation for g. A function $\Psi_X: \mathscr{W}_{g-\{X\}} \rightarrow \mathscr{W}_X$ is called a *solution for* X *with respect to* G if

$$G^{\downarrow(g-\{X\})}(\mathbf{y}) = G(\mathbf{y}, \Psi(\mathbf{y}))$$

for all $\mathbf{y} \in \mathscr{W}_{g-\{X\}}$. Table II shows a solution for X with respect to $G_1 \otimes G_2$.

3. Consistent Valuation-Based Systems

In this section, we formally define what we mean by a consistent valuation-based system.

A *valuation-based system* consists of a finite set of variables $\mathscr{X}$, a finite frame $\mathscr{W}_X$ for each variable X in $\mathscr{X}$, and a finite collection of valuations $\{V_1, \ldots, V_k\}$ where each valuation V_i is for some subset h_i of $\mathscr{X}$. We assume that $\cup\{h_1, \ldots, h_k\} = \mathscr{X}$. (If not, we can always disregard variables that are not included in the domain of some valuation.) Thus, a valuation-based system (VBS) can be denoted formally by the 3-tuple $\{\mathscr{X}, \{\mathscr{W}_X\}_{X \in \mathscr{X}}, \{V_1, \ldots, V_k\}\}$ representing variables, frames, and valuations, respectively.

Suppose $\rho = \{\mathscr{X}, \{\mathscr{W}_X\}_{X \in \mathscr{X}}, \{V_1, \ldots, V_k\}\}$ is a valuation-based system. We say that ρ is *consistent* if $\otimes\{V_1, \ldots, V_k\}$ is a proper valuation and *inconsistent* otherwise. We call $\otimes\{V_1, \ldots, V_k\}$ the *joint valuation*. Note that the joint valuation is a valuation for $\mathscr{X}$.

If a valuation-based system has, say, 50 variables, and each variable has, say, 2 configurations, then the frame of the joint variable $\mathscr{X}$ has 2^{50} configurations. Thus, in this case, it is computationally intractable to explicitly compute the joint valuation in order to verify whether it is consistent or not. In the next section, we describe a fusion algorithm for verifying if a VBS is consistent or not without explicitly computing the joint valuation.

4. A Fusion Algorithm for Consistency Checking

In section 2, we saw that a marginal $G^{\downarrow h}$ is proper if and only if G is proper. Thus, one way of checking whether or

Table V. Disjunctive Elimination Represented as Combination and Marginalization

$\mathscr{W}_{\{Z,X,Y\}}$			G_6	G_7	G_8	$G_6 \otimes G_7 \otimes G_8$	$(G_6 \otimes G_7 \otimes G_8)^{\downarrow\{Z\}}$
z	x	y	1	1	1	1	1
z	x	$\sim y$	1	1	1	1	-
z	$\sim x$	y	1	1	1	1	-
z	$\sim x$	$\sim y$	0	1	1	0	-
$\sim z$	x	y	1	0	0	0	0
$\sim z$	x	$\sim y$	1	0	1	0	-
$\sim z$	$\sim x$	y	1	1	0	0	-
$\sim z$	$\sim x$	$\sim y$	0	1	1	0	-

G_6 represents the disjunction $X = x$ *or* $Y = y$. G_7 represents the conditional *If* $X = x$ *then* $Z = z$. G_8 represents the conditional *If* $Y = y$ *then* $Z = z$. $(G_6 \otimes G_7 \otimes G_8)^{\downarrow\{Z\}}$ represents the conclusion $Z = z$.

not a VBS is proper is to verify if the marginal of the joint valuation for the empty set is proper or not. In this section, we describe a method for computing exactly the marginal of the joint valuation for the empty set without explicitly computing the joint valuation.

Suppose $\rho = \{\mathscr{X}, \{\mathscr{W}_X\}_{X \in \mathscr{X}}, \{V_1, \ldots, V_k\}\}$ is a valuation-based system. We will describe a fusion algorithm for explicitly computing the marginal $(V_1 \otimes \cdots \otimes V_k)^{\downarrow \varnothing}$.

The basic idea of the method is to successively delete all variables from the VBS. Any sequence may be used. All deletion sequences lead to the same answers. But different deletion sequences may involve different computational costs. We will comment on good deletion sequences at the end of this section.

When we delete a variable, we have to do a "fusion" operation on the valuations. Consider a set of m valuations $\{A_1, \ldots, A_m\}$. Suppose A_i is a valuation for a_i for $i = 1, \ldots, m$. Let $\text{Fus}_{X_j}\{A_1, \ldots, A_m\}$ denote the collection of valuations after fusing the valuations in the set $\{A_1, \ldots, A_m\}$ with respect to variable X_j. Then

$$\text{Fus}_{X_j}\{A_1, \ldots, A_m\} = \{A^{\downarrow (g_j - \{X_j\})}\} \cup \{A_i \mid X_j \notin a_i\} \quad (4.1)$$

where $A = \otimes\{A_i \mid X_j \in a_i\}$, and $g_j = \cup\{a_i \mid X_j \in a_i\}$. After fusion, the set of valuations is changed as follows. All valuations that bear on X_j are combined, and the resulting valuation is marginalized such that X_j is eliminated from its domain. The valuations that do not bear on X_j remain unchanged.

When we compute the marginal $A^{\downarrow (g_j - \{X_j\})}$ in (4.1), assume that we store a solution for X_j with respect to A, $\Psi_{X_j}: \mathscr{W}_{g_j - \{X_j\}} \to \mathscr{W}_{X_j}$. In the next section, we describe a method for constructing a model for a consistent VBS using these solutions.

We are ready to state the main theorem which describes the fusion algorithm.

Theorem 1 (Fusion Algorithm). *Suppose* $\rho = \{\mathscr{X}, \{\mathscr{W}_X\}_{X \in \mathscr{X}}, \{V_1, \ldots, V_k\}\}$ *is a valuation-based system. Suppose* $X_1 X_2 \cdots X_n$ *is a sequence of variables in* $\mathscr{X}$. *Then*

$$\otimes \text{Fus}_{X_n}\{\cdots \text{Fus}_{X_2}\{\text{Fus}_{X_1}\{V_1, \ldots, V_k\}\}\} = (V_1 \otimes \cdots \otimes V_k)^{\downarrow \varnothing}.$$

The essence of the fusion algorithm is to perform combinations of valuations on smaller frames instead of combining all valuations on the global frame associated with $\mathscr{X}$.

If the VBS ρ is consistent, then $(V_1 \otimes \cdots \otimes V_k)^{\downarrow \varnothing}(\blacklozenge) = 1$. If ρ is inconsistent, then $(V_1 \otimes \cdots \otimes V_k)^{\downarrow \varnothing}(\blacklozenge) = 0$. Depending on which subsets of valuations are inconsistent, inconsistency may be detected earlier during a fusion operation.

Example 1. (*Does Fred Dislike Dick?*). Consider a knowledge-base consisting of four rules and two facts as follows:

Rule 1: If Fred is a gullible citizen, then Fred is a citizen.
Rule 2: If Dick is an elected crook, then Dick is a crook.
Rule 3: If Fred is a citizen and Dick is a crook, then Fred dislikes Dick.
Rule 4: If Fred is a gullible citizen and Dick is an elected crook, then Fred does not dislike Dick.
Fact 1: Fred is a gullible citizen.
Fact 2: Dick is an elected crook.

Table VI. The Proper Valuations Corresponding to the Four Rules and the Two Facts

$\mathscr{W}_{\{C,G\}}$		R_1	$\mathscr{W}_{\{K,E\}}$		R_2
c	g	1	k	e	1
c	$\sim g$	1	k	$\sim e$	1
$\sim c$	g	0	$\sim k$	e	0
$\sim c$	$\sim g$	1	$\sim k$	$\sim e$	1

$\mathscr{W}_{\{C,K,D\}}$			R_3	$\mathscr{W}_{\{G,E,D\}}$			R_4
c	k	d	1	g	e	d	0
c	k	$\sim d$	0	g	e	$\sim d$	1
c	$\sim k$	d	1	g	$\sim e$	d	1
c	$\sim k$	$\sim d$	1	g	$\sim e$	$\sim d$	1
$\sim c$	k	d	1	$\sim g$	e	d	1
$\sim c$	k	$\sim d$	1	$\sim g$	e	$\sim d$	1
$\sim c$	$\sim k$	d	1	$\sim g$	$\sim e$	d	1
$\sim c$	$\sim k$	$\sim d$	1	$\sim g$	$\sim e$	$\sim d$	1

$\mathscr{W}_{\{G\}}$	F_1	$\mathscr{W}_{\{E\}}$	F_2
g	1	e	1
$\sim g$	0	$\sim e$	0

One representation of this knowledge-base is as follows. Let $C = c$, $G = g$, $K = k$, $E = e$, and $D = d$ be five variables and their respective configurations representing Fred is a citizen, Fred is a gullible citizen, Dick is a crook, Dick is an elected crook, and Fred dislikes Dick, respectively. Suppose all five of these variables have two configurations in their respective frames.

Rules 1, 2, 3, and 4 are represented by proper valuations R_1, R_2, R_3, and R_4 on $\{C, G\}$, $\{K, E\}$, $\{C, K, D\}$, and $\{G, E, D\}$, respectively, as shown in Table VI. Facts 1 and 2 are represented by proper valuations F_1 and F_2 on $\{G\}$ and $\{E\}$, respectively, as also shown in Table VI. Note that the VBS representation of this knowledge-base is essentially a truth-table representation if we replace 1 by T and 0 by F. Thus, the VBS consists of the 3-tuple $\{\mathscr{X} = \{C, G, K, E, D\}$, $\{\mathscr{W}_C = \{c, \sim c\}$, $\mathscr{W}_G = \{g, \sim g\}$, $\mathscr{W}_K = \{k, \sim k\}$, $\mathscr{W}_E = \{e, \sim e\}$, $\mathscr{W}_D = \{d, \sim d\}\}$, $\{R_1, R_2, R_3, R_4, F_1, F_2\}\}$.

A graphical depiction of a valuation-based system is called a *valuation network*. In a valuation network, variables are represented by circular nodes, valuations are represented by rectangular nodes, and each valuation is connected by an undirected edge to each variable node that it bears on. The valuation network for Example 1 is shown in Figure 1.

Figure 2 depicts the fusion algorithm graphically. The first network is the initial VBS. The second network is the VBS resulting from fusion with respect to G. The third network is the VBS resulting from fusion with respect to K. The fourth network is the VBS resulting from fusion with respect to C. The fifth network is the VBS resulting from

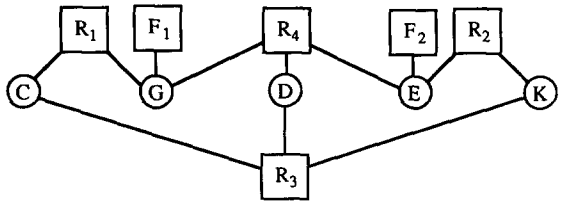

Figure 1. A valuation network for Example 1.

fusion with respect to D. Finally, the sixth network is the VBS resulting from fusion with respect to E. Theorem 1 tells us that $(((R_1 \otimes F_1 \otimes R_4)^{\downarrow\{E,D,C\}} \otimes (R_2 \otimes R_3)^{\downarrow\{E,D,C\}})^{\downarrow\{E\}} \otimes F_2)^{\downarrow\emptyset} = (R_1 \otimes R_2 \otimes R_3 \otimes R_4 \otimes F_1 \otimes F_2)^{\downarrow\emptyset}$. The details of the computations are shown in Tables VII, VIII, and IX. As seen from the last column in Table IX, the valuation $((R_1 \otimes F_1 \otimes R_4)^{\downarrow\{E,D,C\}} \otimes (R_2 \otimes R_3)^{\downarrow\{E,D,C\}})^{\downarrow\{E\}} \otimes F_2$ is improper. Therefore, the VBS is inconsistent.

Deletion Sequences

The sequence in which we delete variables in the fusion algorithm is called the *deletion sequence*. Which deletion sequence should one use? First, note that all deletion sequences lead to the same final result. This is implied in the statement of Theorem 1. Second, different deletion sequences may involve different computational efforts. For example, consider the VBS shown in Figure 1. In this example, all deletion sequences starting with variable D involve more computational effort than sequences that do not start with D as the former involve combinations on the frame of all five variables only, whereas the latter involve combinations on the frame of only four variables. Finding an optimal deletion sequence is a secondary optimization problem that has been shown to be NP-complete.[1] But there are several heuristics for finding good deletion sequences.[11, 13, 27]

One such heuristic is called one-step-look-ahead.[11] This heuristic tells us which variable to delete next. As per this heuristic, the variable that should be deleted next is one that leads to combination over the smallest frame. For example, in the VBS of Figure 1, for first deletion this heuristic would pick either C, G, E, or K over D since deletion of D involves combination over the frame of all five variables whereas deletion of C, G, E, or K involves combination over the frame of only four variables. After the first deletion, any remaining variables can be used for successive deletions as they all lead to combinations over frames of equal sizes.

Computational Complexity

Let us examine the worst-case computational complexity of the fusion algorithm. Suppose there are n variables and k valuations in the valuation-based system. In the fusion algorithm, we do exactly $k - 1$ binary combinations and exactly n marginalizations. Let s denote the largest cardinality of the frames on which we do the combinations. (In Example 1, $s = 16$.) A binary combination on a frame of size s involves at most s multiplications. Marginalizing a variable out of a valuation for a subset whose frame is of size s involves at most $s - 1$ comparisons. Thus the worst-case complexity of the fusion algorithm is $O((n + k)s)$.

Regarding the worst-case complexity of finding a good deletion sequence, if a heuristic such as one-step-look-ahead is used, then this involves at most nk multiplications and n comparisons for determining the first variable in the deletion sequence, at most $(n - 1)k$ multiplications and $n - 1$ comparisons for determining the second variable, etc. Therefore, the worst-case complexity of this algorithm is $O(n^2k)$.

The key parameter here is s. In the fusion algorithm, if the largest subset of variables on which we do a combination has, say, p variables, and each variable has 2 configurations in its frame, then $s = 2^p$. A lower bound for p is of course the size of the largest subset for which we have a valuation. Therefore if we have even one valuation in ρ

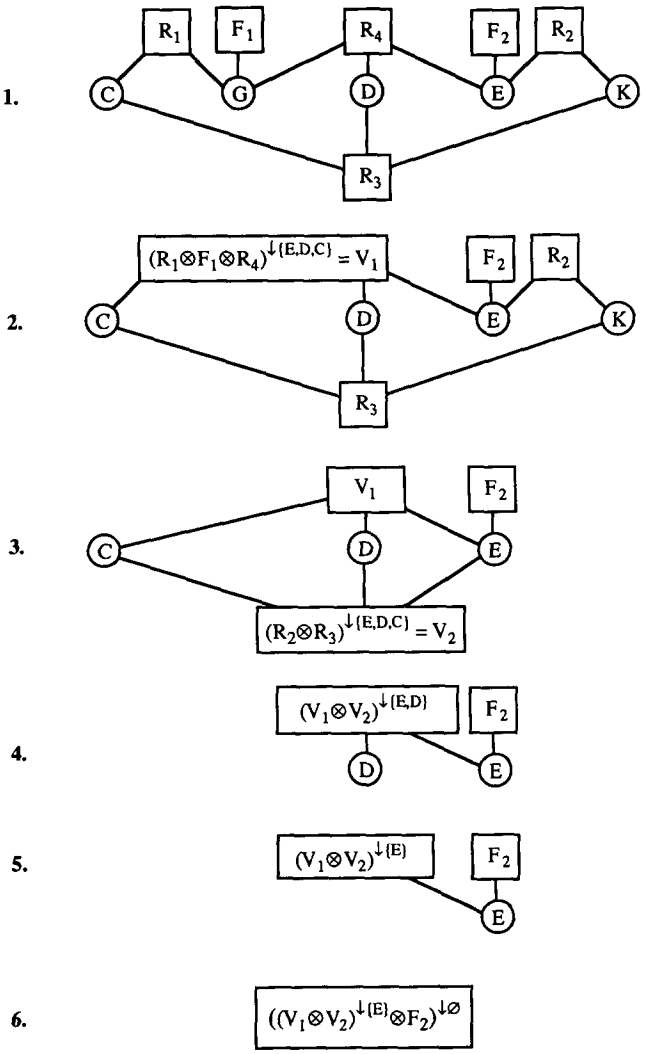

Figure 2. The fusion algorithm applied to the valuation network of Figure 1 using deletion sequence *GKCDE*.

Table VII. The Computation of Valuation $V_1 = (R_1 \otimes F_1 \otimes R_4)^{\downarrow\{E,D,C\}}$

$\mathscr{W}_{\{E,D,C,G\}}$				R_1	F_1	R_4	$R_1 \otimes F_1 \otimes R_4$	$(R_1 \otimes F_1 \otimes R_4)^{\downarrow\{E,D,C\}} = V_1$
e	d	c	g	1	1	0	0	0
e	d	c	$\sim g$	1	0	1	0	-
e	d	$\sim c$	g	0	1	0	0	0
e	d	$\sim c$	$\sim g$	1	0	1	0	-
e	$\sim d$	c	g	1	1	1	1	1
e	$\sim d$	c	$\sim g$	1	0	1	0	-
e	$\sim d$	$\sim c$	g	0	1	1	0	0
e	$\sim d$	$\sim c$	$\sim g$	1	0	1	0	-
$\sim e$	d	c	g	1	1	1	1	1
$\sim e$	d	c	$\sim g$	1	0	1	0	-
$\sim e$	d	$\sim c$	g	0	1	1	0	0
$\sim e$	d	$\sim c$	$\sim g$	1	0	1	0	-
$\sim e$	$\sim d$	c	g	1	1	1	1	1
$\sim e$	$\sim d$	c	$\sim g$	1	0	1	0	-
$\sim e$	$\sim d$	$\sim c$	g	0	1	1	0	0
$\sim e$	$\sim d$	$\sim c$	$\sim g$	1	0	1	0	-

relating all n variables, then $s = 2^n$. Of course, when n is large, it is extremely unlikely that this will ever happen in real life. In most rule-based systems, for example, rules generally relate the values of small subsets of variables.

Furthermore, the value of s depends on the "structure" of the subsets in $\mathscr{H} = \{h_1, \ldots, h_k\}$. If $\mathscr{H}$ is an acyclic hypergraph,[2] then all combinations are done on subsets in $\mathscr{H}$ if a reasonable heuristic (such as one-step-look-ahead) is used to select a deletion sequence.[11] On the other hand, if $\mathscr{H}$ is not acyclic, then some combinations will be on bigger subsets for certain, and depending on the nature of the acyclicity, these subsets could be large.

Table VIII. The Computation of Valuation $V_2 = (R_2 \otimes R_3)^{\downarrow\{E,D,C\}}$

$\mathscr{W}_{\{E,D,C,K\}}$				R_2	R_3	$R_2 \otimes R_3$	$(R_2 \otimes R_3)^{\downarrow\{E,D,C\}} = V_2$
e	d	c	k	1	1	1	1
e	d	c	$\sim k$	0	1	0	-
e	d	$\sim c$	k	1	1	1	1
e	d	$\sim c$	$\sim k$	0	1	0	-
e	$\sim d$	c	k	1	0	0	0
e	$\sim d$	c	$\sim k$	0	1	0	-
e	$\sim d$	$\sim c$	k	1	1	1	1
e	$\sim d$	$\sim c$	$\sim k$	0	1	0	-
$\sim e$	d	c	k	1	1	1	1
$\sim e$	d	c	$\sim k$	1	1	1	-
$\sim e$	d	$\sim c$	k	1	1	1	1
$\sim e$	d	$\sim c$	$\sim k$	1	1	1	-
$\sim e$	$\sim d$	c	k	1	0	0	1
$\sim e$	$\sim d$	c	$\sim k$	1	1	1	-
$\sim e$	$\sim d$	$\sim c$	k	1	1	1	1
$\sim e$	$\sim d$	$\sim c$	$\sim k$	1	1	1	-

5. Generating a Model for a Consistent VBS

Suppose $\rho = \{\mathscr{X}, \{\mathscr{W}_X\}_{X \in \mathscr{X}}, \{V_1, \ldots, V_k\}\}$ is a consistent valuation-based system. In the previous section, we described a method for explicitly computing the marginal $(V_1 \otimes \cdots \otimes V_k)^{\downarrow\varnothing}$. When we implement the fusion algorithm, each time we marginalize a variable, assume that we store a solution for that variable. If we use deletion sequence $X_1 X_2 \cdots X_n$, then at the end of the fusion algorithm, we have for each variable X_j a solution $\Psi_{X_j}: \mathscr{W}_{g_j - \{X_j\}} \to \mathscr{W}_{X_j}$, where g_j is as defined in (4.1). Note that $g_1 = \cup\{h_i \mid X_1 \in h_i\}$. The precise definition of g_2 will depend on the valuations in the $\mathrm{Fus}_{X_1}\{V_1, \ldots, V_k\}$. However, since X_1 has been deleted, $g_2 \subseteq \{X_2, \ldots, X_n\}$ and $X_2 \in g_2$. In general, $g_i \subseteq \{X_i, \ldots, X_n\}$, and $X_i \in g_i$ for $i = 1, \ldots, n$. Note that $g_n = \{X_n\}$.

Theorem 2 describes a recursive method for constructing a model for the joint valuation. The model is constructed piecemeal starting with the component corresponding to X_n and working sequentially opposite to the deletion sequence.

Theorem 2. *Suppose $\rho = \{\mathscr{X}, \{\mathscr{W}_X\}_{X \in \mathscr{X}}, \{V_1, \ldots, V_k\}\}$ is a consistent valuation-based system. Suppose $X_1 X_2 \cdots X_n$ is a sequence of variables in $\mathscr{X}$. Suppose $\Psi_{X_j}: \mathscr{W}_{g_j - \{X_j\}} \to \mathscr{W}_{X_j}$ is a solution for X_j computed during the fusion operation $\mathrm{Fus}_{X_j}\{\cdots \mathrm{Fus}_{X_2}\{\mathrm{Fus}_{X_1}\{V_1, \ldots, V_k\}\}\}$, for $j = 1, \ldots, n$. Then $\mathbf{z} \in \mathscr{W}_{\mathscr{X}}$ given by*

$$\mathbf{z}^{\downarrow\{X_j\}} = \Psi_{X_j}(\mathbf{z}^{\downarrow(g_j - \{X_j\})}) \text{ for } j = n, n-1, \ldots, 1$$

is a model for $V_1 \otimes \cdots \otimes V_k$.

Table IX. The Computation of Valuations $(V_1 \otimes V_2)^{\downarrow\{E,D\}}$, $(V_1 \otimes V_2)^{\downarrow\{E\}}$, and $(V_1 \otimes V_2)^{\downarrow\{E\}} \otimes F_2$

$\mathscr{W}_{\{E,D,C\}}$			V_1	V_2	$V_1 \otimes V_2$	$(V_1 \otimes V_2)^{\downarrow\{E,D\}}$	$(V_1 \otimes V_2)^{\downarrow\{E\}}$	F_2	$(V_1 \otimes V_2)^{\downarrow\{E\}} \otimes F_2$
e	d	c	0	1	0	0	0	1	0
e	d	$\sim c$	0	1	0	-	-	-	-
e	$\sim d$	c	1	0	0	0	-	-	-
e	$\sim d$	$\sim c$	0	1	0	-	-	-	-
$\sim e$	d	c	1	1	1	1	1	0	0
$\sim e$	d	$\sim c$	0	1	0	-	-	-	-
$\sim e$	$\sim d$	c	1	1	1	1	-	-	-
$\sim e$	$\sim d$	$\sim c$	0	1	0	-	-	-	-

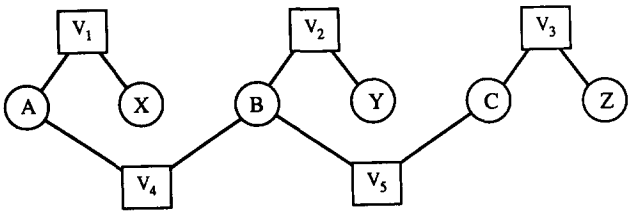

Figure 3. The valuation network for Example 2.

Example 2. Consider a VBS in which there are six variables A, B, C, X, Y, and Z. The frame for each variable has 3 configurations: 0, 1, and 2. Thus we have $3^6 = 729$ configurations of all 6 variables. Our knowledge about these variables can be expressed as five relations as follows: $A = X$, $B = Y$, $C = Z$, $A \neq B$, and $B \neq C$. $A = X$ means that variables A and X have the same value (0, 1, or 2), and so on. These five relations can be encoded as valuations $V_1, \ldots, V_5$ as shown in Table X. The valuation network for this example is shown in Figure 3.

If we apply the fusion algorithm with deletion sequence $XYZACB$, then we get the result $(((V_1^{\downarrow\{A\}} \otimes V_4)^{\downarrow\{B\}}) \otimes ((V_3^{\downarrow\{C\}} \otimes V_5)^{\downarrow\{B\}}) \otimes V_2^{\downarrow\{B\}})^{\downarrow\varnothing} = (V_1 \otimes \cdots \otimes V_5)^{\downarrow\varnothing}$. The details of the computation are shown in Table XI. The VBS is consistent. A model for the joint valuation is constructed as follows. Since $\Psi_B(\blacklozenge) = 0$, $\Psi_C(0) = 1$, $\Psi_A(0) = 1$, $\Psi_Z(1) = 1$, $\Psi_Y(0) = 0$, and $\Psi_X(0) = 0$, using Theorem 2, $B = 0$, $C = 1$, $A = 1$, $Z = 1$, $Y = 0$, $X = 1$ is a model for the joint valuation $V_1 \otimes \cdots \otimes V_5$.

6. Identifying a Minimal Set of Inconsistent Valuations

If a valuation-based system is inconsistent, then it is useful to isolate a minimal subset of inconsistent valuations. This will help the knowledge engineer to modify the knowledge-base so that it is consistent. In this section we describe a method for identifying a minimal subset.

Suppose $\{V_1, \ldots, V_k\}$ is a collection of proper valuations in a VBS ρ. Suppose ρ is inconsistent. Suppose $I_m \subseteq \{V_1, \ldots, V_k\}$. We say I_m is a *minimal set of inconsistent valuations* if $\otimes\{V_i \mid V_i \in I_m\}$ is improper, and for every proper subset I' of I_m, $\otimes\{V_i \mid V_i \in I'\}$ is proper. Since each valuation V_i is proper, a minimal set of inconsistent valuations will have at least two valuations.

As we mentioned in section 4, if a set of proper valuations is inconsistent, then this is manifested in the fusion algorithm when we do the combinations. Suppose the combination operation is implemented in a binary fashion, i.e., we combine valuations two at a time and stop whenever we get an improper valuation as a result. Also, suppose that when we combine two valuations, we keep track of the

Table X. The Valuations $V_1, \ldots, V_5$ in Example 2

$\mathscr{W}_A \times \mathscr{W}_X$		V_1	$\mathscr{W}_B \times \mathscr{W}_Y$		V_2	$\mathscr{W}_C \times \mathscr{W}_Z$		V_3	$\mathscr{W}_B \times \mathscr{W}_A$		V_4	$\mathscr{W}_B \times \mathscr{W}_C$		V_5
0	0	1	0	0	1	0	0	1	0	0	0	0	0	0
0	1	0	0	1	0	0	1	0	0	1	1	0	1	1
0	2	0	0	2	0	0	2	0	0	2	1	0	2	1
1	0	0	1	0	0	1	0	0	1	0	1	1	0	1
1	1	1	1	1	1	1	1	1	1	1	0	1	1	0
1	2	0	1	2	0	1	2	0	1	2	1	1	2	1
2	0	0	2	0	0	2	0	0	2	0	1	2	0	1
2	1	0	2	1	0	2	1	0	2	1	1	2	1	1
2	2	1	2	2	1	2	2	1	2	2	0	2	2	0

Table XI. The Details of Computations in Example 2

$\mathscr{W}_A \times \mathscr{W}_X$		V_1	$V_1^{\downarrow\{A\}}$	Ψ_X	$\mathscr{W}_B \times \mathscr{W}_Y$		V_2	$V_2^{\downarrow\{B\}}$	Ψ_Y	$\mathscr{W}_C \times \mathscr{W}_Z$		V_3	$V_3^{\downarrow\{C\}}$	Ψ_Z
0	0	1	1	0	0	0	1	1	0	0	0	1	1	0
0	1	0	-	-	0	1	0	-	-	0	1	0	-	-
0	2	0	-	-	0	2	0	-	-	0	2	0	-	-
1	0	0	1	1	1	0	0	1	1	1	0	0	1	1
1	1	1	-	-	1	1	1	-	-	1	1	1	-	-
1	2	0	-	-	1	2	0	-	-	1	2	0	-	-
2	0	0	1	2	2	0	0	1	2	2	0	0	1	2
2	1	0	-	-	2	1	0	-	-	2	1	0	-	-
2	2	1	-	-	2	2	1	-	-	2	2	1	-	-

$\mathscr{W}_B \times \mathscr{W}_A$		V_4	$V_1^{\downarrow\{A\}}$	$V_4 \otimes V_1^{\downarrow\{A\}}$	$(V_4 \otimes V_1^{\downarrow\{A\}})^{\downarrow\{B\}}$	Ψ_A
0	0	0	1	0	1	1
0	1	1	1	1	-	-
0	2	1	1	1	-	-
1	0	1	1	1	1	0
1	1	0	1	0	-	-
1	2	1	1	1	-	-
2	0	1	1	1	1	0
2	1	1	1	1	-	-
2	2	0	1	0	-	-

$\mathscr{W}_B \times \mathscr{W}_C$		V_5	$V_3^{\downarrow\{C\}}$	$V_5 \otimes V_3^{\downarrow\{C\}}$	$(V_5 \otimes V_3^{\downarrow\{C\}})^{\downarrow\{B\}}$	Ψ_C
0	0	0	1	0	1	1
0	1	1	1	1	-	-
0	2	1	1	1	-	-
1	0	1	1	1	1	0
1	1	0	1	0	-	-
1	2	1	1	1	-	-
2	0	1	1	1	1	0
2	1	1	1	1	-	-
2	2	0	1	0	-	-

$\mathscr{W}_B$	$(V_4 \otimes V_1^{\downarrow\{A\}})^{\downarrow\{B\}}$	$(V_5 \otimes V_3^{\downarrow\{C\}})^{\downarrow\{B\}}$	$V_2^{\downarrow\{B\}}$	$(V_4 \otimes V_1^{\downarrow\{A\}})^{\downarrow\{B\}} \otimes (V_5 \otimes V_3^{\downarrow\{C\}})^{\downarrow\{B\}} \otimes V_2^{\downarrow\{B\}} = V$	$V^{\downarrow\varnothing}(\blacklozenge)$	$\Psi_B(\blacklozenge)$
0	1	1	1	1	1	0
1	1	1	1	1	-	-
2	1	1	1	1	-	-

collection of initial valuations V_i which the combination represents. Thus, if inconsistency is detected during the combination operation associated with a fusion, then we can isolate the subset of valuations that are inconsistent. This subset of inconsistent valuations will depend on the sequence in which the valuations are combined. In particular, it will depend on the last valuation, say V_1, that would have been combined during the fusion algorithm if we had continued the fusion algorithm to the very end. Let $I(V_1)$ denote this set of inconsistent valuations. Of course, $I(V_1)$ need not be minimal. Note that if inconsistency is first detected during the last combination, then $V_1 \in I(V_1)$, else $V_1 \notin I(V_1)$.

Suppose we repeat the fusion algorithm by making three changes. First, we only fuse the valuations in $I(V_1)$. Second, we pick a valuation in $I(V_1)$, say V_2, that is different from V_1, and we pick a deletion sequence such that the variables in h_2 (the domain of V_2) succeed all other variables. Third, in implementing the fusion algorithm, valuation V_2 is picked to be the last valuation to be combined assuming we combine all valuations in $I(V_1)$. Since $I(V_1)$ is an inconsistent set of valuations, the fusion algorithm will stop after some combination results in an improper valuation. Let $I(V_2)$ denote the subset of initial valuations that is detected to be inconsistent during this second step. Note that $I(V_2) \subseteq I(V_1)$. As in the first step, V_2 may or may not be in-

cluded in $I(V_2)$. We repeat this process until we are left with a set $I(V_m)$ of inconsistent valuations such that we have no more valuations to pick as the last valuation, i.e., every valuation in $I(V_m)$ has already served as a last valuation. The following theorem asserts that $I(V_m)$ is a minimal inconsistent set.

Theorem 3. *Under the assumptions of the last paragraph, $I(V_m)$ is a minimal set of inconsistent valuations.*

Since the complexity of each propagation is $O((n + k)s)$, and in the worst case every valuation may be included in $I(V_m)$, the worst-case complexity of our technique for finding a minimal set of inconsistent valuations in an inconsistent VBS is $O((n + k)ks)$.

7. Conclusion

We have described a method for checking whether a VBS is consistent or not. We have described a method for constructing a model for a consistent VBS. And we have described a method for isolating a minimal set of valuations in an inconsistent VBS.

The method for checking whether a VBS is consistent or not is quite general. Although we have described the method for propositional logic, the method can be easily adapted to other domains such as probability theory, Dempster-Shafer theory of belief functions, and Zadeh's possibility theory. What is needed for the method to work are definitions of a valuation, a proper valuation, combination, and marginalization. Also, for the fusion algorithm to give correct results, combination and marginalization need to satisfy three axioms. These axioms are stated in [25, 18, 22].

In the fusion algorithm, just before we compute the marginal of the joint valuation for the empty set, we compute the marginal of the joint valuation for X_n, the last variable in the deletion sequence. Shenoy and Shafer[25] describe an efficient implementation of the fusion algorithm for computing the marginal of the joint for each variable for approximately twice the cost of computing the marginal for one variable. As we explained earlier in section 2, finding the marginal of the joint valuation is a form of logical inference. Thus the fusion algorithm can also be used to make inferences from a valuation-based system. Shenoy[16, 22] describes such a language that uses valuations to encode knowledge and uses combination and marginalization operations to make inferences from the knowledge-base.

8. Proofs

In this section, we give proofs for the three theorems in the paper stated in sections 4, 5, and 6, respectively.

Lemma 1. *Suppose $\rho = \{\mathscr{X}, \{\mathscr{W}_X\}_{X \in \mathscr{X}}, \{V_1, \ldots, V_k\}\}$ is a VBS. Suppose X is a variable in $\mathscr{X}$. Then*

$$(\otimes\{V_1, \ldots, V_k\})^{\downarrow(\mathscr{X}-\{X\})} = \otimes Fus_X\{V_1, \ldots, V_k\}.$$

Proof of Lemma 1. Without loss of generality, suppose that $V_1, \ldots, V_m$ are the only valuations that bear on X. Let $V = V_1 \otimes \cdots \otimes V_m$, let $g = g_1 \cup \cdots \cup g_m$, and let $\mathbf{c} \in \mathscr{W}_{\mathscr{X}-\{X\}}$. Then

$$\begin{aligned}
&(\otimes\{V_1, \ldots, V_k\})^{\downarrow(\mathscr{X}-\{X\})}(\mathbf{c}) \\
&\quad = \mathrm{MAX}\{[V_1(\mathbf{c}^{\downarrow g_1}, \mathbf{x}) \cdots V_m(\mathbf{c}^{\downarrow g_m}, \mathbf{x}) \\
&\quad \times V_{m+1}(\mathbf{c}^{\downarrow g_{m+1}}) \cdots V_k(\mathbf{c}^{\downarrow g_k})] \mid \mathbf{x} \in \mathscr{W}_X\} \\
&\quad = \mathrm{MAX}\{[V_1(\mathbf{c}^{\downarrow g_1}, \mathbf{x}) \cdots V_m(\mathbf{c}^{\downarrow g_m}, \mathbf{x})] \mid \mathbf{x} \in \mathscr{W}_X\} \\
&\qquad \times [V_{m+1}(\mathbf{c}^{\downarrow g_{m+1}}) \cdots V_k(\mathbf{c}^{\downarrow g_k})] \\
&\quad = \mathrm{MAX}\{V(\mathbf{c}^{\downarrow g-\{X\}}, \mathbf{x}) \mid \mathbf{x} \in \mathscr{W}_X\} \\
&\qquad \times [V_{m+1}(\mathbf{c}^{\downarrow g_{m+1}}) \cdots V_k(\mathbf{c}^{\downarrow g_k})] \\
&\quad = V^{\downarrow g-\{X\}}(\mathbf{c}^{\downarrow g-\{X\}})[V_{m+1}(\mathbf{c}^{\downarrow g_{m+1}}) \cdots V_k(\mathbf{c}^{\downarrow g_k})] \\
&\quad = V^{\downarrow g-\{X\}} \otimes [V_{m+1} \otimes \cdots \otimes V_k](\mathbf{c}) \\
&\quad = (\otimes \mathrm{Fus}_X\{V_1, \ldots, V_k\})(\mathbf{c}). \qquad \blacksquare
\end{aligned}$$

Proof of Theorem 1. A proof of this theorem is obtained by repeatedly applying the result of Lemma 1. At each step, we delete a variable and fuse the set of all valuations with respect to this variable. Using Lemma 1, after fusion with respect to X_1, the combination of all valuations in the resulting VBS is equal to $(\otimes\{V_1, \ldots, V_k\})^{\downarrow(\mathscr{X}-\{X_1\})}$. Again, using Lemma 1, after fusion with respect to X_2, the combination of all valuations in the resulting VBS is equal to $(\otimes\{V_1, \ldots, V_k\})^{\downarrow(\mathscr{X}-\{X_1, X_2\})}$, and so on. When all the variables have been deleted, using Lemma 1, the combination of all valuation left (there may be just one) will be $(\otimes\{V_1, \ldots, V_k\})^{\downarrow\varnothing}$. ■

Next, we state and prove a lemma we need to prove Theorem 2.

Lemma 2. *Suppose $\rho = \{\mathscr{X}, \{\mathscr{W}_X\}_{X \in \mathscr{X}}, \{V_1, \ldots, V_k\}\}$ is a consistent VBS. Suppose X is a variable in $\mathscr{X}$. Suppose Ψ_X: $\mathscr{W}_{g^-\{X\}} \to \mathscr{W}_X$ is a solution for X computed during $Fus_X\{V_1, \ldots, V_k\}$, and suppose $\mathbf{c} \in \mathscr{W}_{\mathscr{X}-\{X\}}$ is a model for $(\otimes\{V_1, \ldots, V_k\})^{\downarrow(\mathscr{X}-\{X\})}$. Then $(\mathbf{c}, \Psi(\mathbf{c}^{\downarrow(g^-\{X\})}))$ is a model for $\otimes\{V_1, \ldots, V_k\}$.*

Proof of Lemma 2. Without loss of generality, suppose that $V_1, \ldots, V_m$ are the only valuations that bear on X. Let $V = V_1 \otimes \cdots \otimes V_m$, let $g = g_1 \cup \cdots \cup g_m$, and let $\mathbf{c} \in \mathscr{W}_{\mathscr{X}-\{X\}}$. We need to prove that $(\otimes\{V_1, \ldots, V_k\})(\mathbf{c}, \Psi(\mathbf{c}^{\downarrow(g-\{X\})})) = 1$. We have

$$\begin{aligned}
&(\otimes\{V_1, \ldots, V_k\})(\mathbf{c}, \Psi(\mathbf{c}^{\downarrow(g-\{X\})})) \\
&\quad = V(\mathbf{c}^{\downarrow(g-\{X\})}, \Psi(\mathbf{c}^{\downarrow(g-\{X\})}))V_{m+1}(\mathbf{c}^{\downarrow g_{m+1}}) \cdots V_k(\mathbf{c}^{\downarrow g_k}) \\
&\qquad \text{(by definition of combination)} \\
&\quad = V^{\downarrow(g-\{X\})}(\mathbf{c}^{\downarrow(g-\{X\})})V_{m+1}(\mathbf{c}^{\downarrow g_{m+1}}) \cdots V_k(\mathbf{c}^{\downarrow g_k}) \\
&\qquad \text{(since } \Psi \text{ is a solution for } X\text{)} \\
&\quad = (\otimes \mathrm{Fus}_X\{V_1, \ldots, V_k\})(\mathbf{c}) \\
&\qquad \text{(by definition of } \mathrm{Fus}_X\{V_1, \ldots, V_k\}\text{)}
\end{aligned}$$

$= (\otimes\{V_1, \ldots, V_k\})^{\downarrow(\mathscr{X}-\{X\})}(\mathbf{c})$

(by Lemma 1)

$= 1$ (since $\mathbf{c}$ is a model for $\otimes\{V_1, \ldots, V_k\})^{\downarrow(\mathscr{X}-\{X\})}$ ■

Proof of Theorem 2. A proof of this theorem is obtained by repeatedly applying Lemma 2. Consider the VBS $\mathrm{Fus}_{X_n}\{\cdots \mathrm{Fus}_{X_2}\{\mathrm{Fus}_{X_1}\{V_1, \ldots, V_k\}\}\}$. There is only one valuation in this VBS and it is for the empty set. Since ρ is consistent, $(\mathrm{Fus}_{X_n}\{\cdots \mathrm{Fus}_{X_2}\{\mathrm{Fus}_{X_1}\{V_1, \ldots, V_k\}\}\})(\blacklozenge) = (V_1 \otimes \cdots \otimes V_k)^{\downarrow\varnothing}(\blacklozenge) = 1$. Since $\blacklozenge$ is a model for $(V_1 \otimes \cdots \otimes V_k)^{\downarrow\varnothing}$, by Lemma 2, $(\blacklozenge, \Psi_{X_n}(\blacklozenge)) = \Psi_{X_n}(\blacklozenge) = \mathbf{z}^{\downarrow\{X_n\}}$ is a model for $(V_1 \otimes \cdots \otimes V_k)^{\downarrow\{X_n\}}$.

Since $\mathbf{z}^{\downarrow\{X_n\}}$ is a model for $(V_1 \otimes \cdots \otimes V_k)^{\downarrow\{X_n\}}$, and $\Psi_{X_{n-1}}: \mathscr{W}_{g_{n-1}-\{X_{n-1}\}} \rightarrow \mathscr{W}_{X_{n-1}}$ is a solution for X_{n-1}, by Lemma 2, $(\mathbf{z}^{\downarrow\{X_n\}}, \Psi_{X_{n-1}}(\mathbf{z}^{\downarrow(g_{n-1}-\{X_{n-1}\})})) = (\mathbf{z}^{\downarrow\{X_n\}}, \mathbf{z}^{\downarrow\{X_{n-1}\}}) = \mathbf{z}^{\downarrow\{X_n, X_{n-1}\}}$ is a model for $(V_1 \otimes \cdots \otimes V_k)^{\downarrow\{X_n, X_{n-1}\}}$.

Continuing in this fashion, we get the result that $\mathbf{z}$ is a model for $V_1 \otimes \cdots \otimes V_k$. ■

Proof of Theorem 3. It is clear that the valuations in $I(V_m)$ are inconsistent. We will prove minimality of $I(V_m)$ by contradiction. Suppose $I(V_m)$ is not minimal. Then there exists a valuation $V_j \in I(V_m)$ such that $\otimes\{V_t \mid V_t \in I(V_m) - \{V_j\}\}$ is improper. Since V_j has served as a last valuation, consider the step in which V_j is the last valuation to be combined. Since $\otimes\{V_t \mid V_t \in I(V_m) - \{V_j\}\}$ is improper and since $I(V_m) \subseteq I(V_j)$, this means that inconsistency is detected in this step before V_j is combined since V_j is combined last. Therefore $V_j \notin I(V_j)$. This contradicts the assumption that $V_j \in I(h_m)$ since $I(V_m) \subseteq I(V_j)$. ■

Acknowledgments

This work was supported in part by the National Science Foundation under grant IRI-8902444. The paper has benefited a lot from the comments of three anonymous referees, an associate editor, and the area editor, John N. Hooker, Jr.

References

1. S. Arnborg, D.G. Corneil and A. Proskurowski, 1987. Complexity of Finding Embeddings in a k-Tree, *SIAM Journal of Algebraic and Discrete Methods 8*, 277–284.
2. C. Beeri, R. Fagin, D. Maier and M. Yannakakis, 1983. On the Desirability of Acyclic Database Schemes, *Journal of the ACM 30:3*, 479–513.
3. S. Cook, 1971. The Complexity of Theorem-Proving Procedures, in *Proceedings of the Third ACM Symposium on Theory of Computing*, 151–158.
4. M. Davis and H. Putnam, 1960. A Computing Procedure for Quantification Theory, *Journal of the ACM 7:3*, 201–215.
5. J. de Kleer, 1986. An Assumption-Based TMS, *Artificial Intelligence 28*, 127–162.
6. W.F. Dowling and J H. Gallier, 1984. Linear-Time Algorithms for Testing the Satisfiability of Propositional Horn Formulae, *Journal of Logic Programming 3*, 267–284.
7. D. Dubois and H. Prade, 1991. Inference in Possibilistic Hypergraphs, in Uncertainty in Knowledge Bases, B. Bouchon-Meunier, R.R. Yager and L.A. Zadeh (eds.), Lecture Notes in Computer Science No. 521, 250–259, Springer-Verlag, Berlin.
8. G. Gallo and G. Urbani, 1989. Algorithms for Testing the Satisfiability of Propositional Formulae, *Journal of Logic Programming 7*, 45–61.
9. A. Ginsberg, 1988. Knowledge-Base Reduction: A New Approach to Checking Knowledge Bases for Inconsistency and Redundancy, in *Proceedings of the Seventh National Conference on Artificial Intelligence (AAAI-88) 1*, St. Paul, MN, 585–589.
10. R.G. Jeroslow and J. Wang, 1990. Solving Propositional Satisfiability Problems, *Annals of Mathematics and Artificial Intelligence 1*, 167–187.
11. A. Kong, 1986. Multivariate Belief Functions and Graphical Models, Ph.D. dissertation, Department of Statistics, Harvard University, Cambridge, MA.
12. D.W. Loveland, 1978. *Automated Theorem Proving: A Logical Basis*, North-Holland, Amsterdam.
13. K Mellouli, 1987. On the Propagation of Beliefs in Networks Using the Dempster-Shafer Theory of Evidence, Ph.D. dissertation, School of Business, University of Kansas, Lawrence, KS.
14. T.A. Nguyen, W.A. Perkins, T.J. Laffey and D. Pecora, 1986. Checking an Expert Systems Knowledge Base for Consistency and Completeness, in *Proceedings of the 9th International Joint Conference on AI (IJCAI-85) 1*, 375–378, Los Angeles, CA.
15. G. Shafer, 1976. *A Mathematical Theory of Evidence*, Princeton University Press, Princeton, NJ.
16. P.P. Shenoy, 1989. A Valuation-Based Language for Expert Systems, *International Journal of Approximate Reasoning 3:5*, 383–411.
17. P.P. Shenoy, 1991. On Spohn's Rule for Revision of Beliefs, *International Journal of Approximate Reasoning 5:2*, 149–181.
18. P.P. Shenoy, 1991. Valuation-Based Systems for Discrete Optimization, in *Uncertainty in Artificial Intelligence 6*, P.P. Bonissone, M. Henrion, L.N. Kanal and J.F. Lemmer (eds.), 385–400, North Holland, Amsterdam.
19. P.P. Shenoy, 1992. Using Possibility Theory in Expert Systems, *Fuzzy Sets and Systems 52:2*, 129–142.
20. P.P. Shenoy, 1992. Valuation-Based Systems for Bayesian Decision Analysis, *Operations Research 40:3*, 463–484.
21. P P. Shenoy, 1993. A New Method for Representing and Solving Bayesian Decision Problems, in *Artificial Intelligence and Statistics III*, D.J. Hand (ed.), 119–138, Chapman & Hall, London.
22. P.P. Shenoy, 1992. Valuation-Based Systems: A Framework for Managing Uncertainty in Expert Systems, in *Fuzzy Logic for the Management of Uncertainty*, L.A. Zadeh and J. Kacprzyk (eds.), 83–104, Wiley, New York.
23. P.P Shenoy and G. Shafer, 1986. Propagating Belief Functions Using Local Computation, *IEEE Expert 1:3*, 43–52.
24. P.P. Shenoy and G. Shafer, 1988. Constraint Propagation, Working Paper No. 208, School of Business, University of Kansas, Lawrence, KS.
25. P.P Shenoy and G. Shafer, 1990. Axioms for Probability and Belief Function Propagation, in *Uncertainty in Artificial Intelligence 4*, R. Schachter, T. Levitt, J.F. Lemmer and L.N. Kanal (eds.), 169–198, North-Holland, Amsterdam. Reprinted in G. Shafer and J. Pearl (eds.), 1990. *Readings in Uncertain Reasoning*, Morgan Kaufmann, San Mateo, CA, 575–610.
26. M. Suwa, A.C. Scott and E.H. Shortliffe, 1982. An Approach to Verifying Completeness and Consistency in a Rule-Based Expert System, *AI Magazine 3:3*, 16–21.
27. L. Zhang, 1988. Studies on Finding Hypertree Covers of Hypergraphs, Working Paper No. 198, School of Business, University of Kansas, Lawrence, KS.