# Local Computation with Valuations from a Commutative Semigroup

S. L. Lauritzen*   F. V. Jensen†

October 23, 1996

**Abstract**

This paper studies a variant of axioms originally developed by Shafer and Shenoy (1988). It is investigated which extra assumptions are needed to perform the local computations in a HUGIN-like architecture (Jensen *et al.* 1990) or in the architecture of Lauritzen and Spiegelhalter (1988). In particular it is shown that propagation of belief functions can be performed in these architectures.

**Keywords:** artificial intelligence, belief function, constraint propagation, expert system, probability propagation, valuation-based system.

## 1 Introduction

An important development in artificial intelligence is associated with an abstract theory of local computation known as the Shafer–Shenoy axioms (Shafer and Shenoy 1988; Shenoy and Shafer 1990). These describe in a very general setting how computations can be performed efficiently and locally in a variety of problems, just if a few simple conditions are satisfied. Even though the axioms were developed to formalize computation with belief functions (Shafer 1976), the general approach was valuable for a wide range of problems, including dynamic programming, constraint satisfaction, probabilistic inference, Spohn's belief revision, and others. Further, it enhanced flexibility in programming these local computation algorithms.

Lauritzen and Spiegelhalter (1988) developed a related algorithm for probability calculations and this algorithm was improved in the implementation of the HUGIN shell (Andersen *et al.* 1989; Jensen *et al.* 1990).

No abstract framework has been made that places the latter algorithms in a generality similar to that of Shafer and Shenoy although Dawid (1992) and Cowell and Dawid (1992) show that the HUGIN algorithm certainly has a generalization beyond standard probability calculations.

*Department of Mathematics, Aalborg University, Denmark
†Department of Computer Science, Aalborg University, Denmark

The present paper is concerned with the development of such an abstract framework. The crucial point is that the algorithms involve a division operation, whereas this is not the case for the computations in the Shafer–Shenoy framework.

A thorough comparison of the various algorithms in terms of computational efficiency would be interesting. This would involve a discussion of specific implementation issues, and we emphasize that such a comparison is outside the scope of the present paper, which is concerned with validity rather than efficiency.

# 2 Commutative Semigroups of Valuations

## 2.1 Basic examples

Following Shenoy and Shafer (1990) we study a commutative semigroup $(\mathcal{V}, \otimes)$ of *valuations* and assume that this has a unit $1 \in \mathcal{V}$. Thus the composition rule $\otimes$ satisfies for all $u$, $v$, and $w$ in $\mathcal{V}$

$$1 \otimes u = u, \quad u \otimes v = v \otimes u, \quad (u \otimes v) \otimes w = u \otimes (v \otimes w). \tag{1}$$

There is an abundance of examples of such semigroups that play an important role for local computations and the reader is referred to the above paper as well as other papers by P. Shenoy and collaborators for a large spectrum. In this paper we concentrate on a few of these that are central to artificial intelligence applications, or well suited to illustrate the basic algebraic issues.

**Example 1** An important example is $(\mathbf{P}(\mathcal{X}), \cdot)$, the semigroup of non-negative real functions on a finite product space $\mathcal{X} = \times_{v \in V} \mathcal{X}_v$ (where $V$ and $\mathcal{X}_v$ are finite sets) under multiplication, i.e.

$$(f \cdot g)(x) = f(x)g(x).$$

This semigroup is the basic semigroup for probability calculations. Usually we suppress the space $\mathcal{X}$ and write just $(\mathbf{P}, \cdot)$ or even $\mathbf{P}$ when this does not lead to ambiguities. □

**Example 2** Another important example is the semigroup $(\mathcal{B}(\mathcal{X}), \oplus)$, where $\mathcal{B}(\mathcal{X})$ is the set of *belief functions* over $\mathcal{X}$, and the combination is *Dempster's rule* (Shafer 1976). Here $\mathcal{X}$ is a finite product space as above.

Such a belief function $B \in \mathcal{B}(\mathcal{X})$ can for example be represented by its *mass function* $m$. This is any function defined on the set of subsets of $\mathcal{X}$ which satisfies

$$m(\emptyset) = 0, \quad m(A) \geq 0, \quad \sum_{A : A \subseteq \mathcal{X}} m(A) = 1. \tag{2}$$

The *belief* $B$ for any given $A \subseteq \mathcal{X}$ is then

$$B(A) = \sum_{C: C \subseteq A} m(C).$$

The subsets $A$ with $m(A) > 0$ are called the *focal elements* of $B$.

Dempster's rule for combination of two belief functions $B_1$ and $B_2$ can be expressed in terms of their mass functions $m_1$ and $m_2$ as

$$(m_1 \oplus m_2)(A) = \sum_{(A_1, A_2): A_1 \cap A_2 = A} m_1(A_1) m_2(A_2)/\kappa_{12} \quad \text{for } A \neq \emptyset$$

where $\kappa_{12}$ denotes a normalization constant

$$\kappa_{12} = \sum_{(A_1, A_2): A_1 \cap A_2 \neq \emptyset} m_1(A_1) m_2(A_2).$$

An alternative representation of a belief function is through its *commonality function* $Q$ defined as

$$Q(A) = \sum_{C: C \supseteq A} m(C). \tag{3}$$

The commonality function has $Q(\emptyset) = 1$ and is clearly non-increasing, i.e.

$$A \subseteq C \implies Q(A) \geq Q(C).$$

For simplicity we ignore the normalization and the condition that $\sum_A m(A) = 1$. Then Dempster's combination rule has the following simple expression in terms of commonality functions

$$(Q_1 \oplus Q_2)(A) = Q_1(A) Q_2(A) \text{ for } A \neq \emptyset.$$

This fact is well-known, but can also easily be seen by the calculation

$$\begin{aligned}
Q_1(A) Q_2(A) &= \left\{ \sum_{A_1: A_1 \supseteq A} m_1(A_1) \right\} \left\{ \sum_{A_2: A_2 \supseteq A} m_2(A_2) \right\} \\
&= \sum_{A_1: A_1 \supseteq A} \sum_{A_2: A_2 \supseteq A} m_1(A_1) m_2(A_2) \\
&= \sum_{D: D \supseteq A} \sum_{(A_1, A_2): A_1 \cap A_2 = D} m_1(A_1) m_2(A_2) \\
&= \sum_{D: D \supseteq A} (m_1 \oplus m_2)(D) \\
&= (Q_1 \oplus Q_2)(A).
\end{aligned}$$

However, the semigroup is not isomorphic to the multiplicative semigroup of non-negative functions on $\mathcal{P}(\mathcal{X})$, as not any non-negative function corresponds to a mass function that satisfies the constraints (2). The mass function can be expressed in terms of the commonality function through the inverse Möbius transform as

$$m(A) = \sum_{C:C\supseteq A} (-1)^{|C\setminus A|} Q(C) \quad \text{for } A \neq \emptyset \tag{4}$$

and only those non-negative functions $Q$ on $\mathcal{P}(\mathcal{X})$ whose inverse Möbius transforms are non-negative, correspond to belief functions. □

**Example 3** A third example is associated with *constraint satisfaction* problems. Here the semigroup is $(\mathcal{P}(\mathcal{X}), \cap)$, where $\mathcal{P}(\mathcal{X})$ is the set of subsets of the space $\mathcal{X}$. This semigroup is *idempotent,* i.e. all elements satisfy $v \otimes v = v$. The same holds for the semigroup $(\mathbf{C}(\mathcal{X}), \cap)$ of convex sets of probability measures, i.e. general convex subsets of $\mathbf{P}(\mathcal{X})$, or when $\mathbf{C}(\mathcal{X})$ is replaced with the set of convex polytopes in $\mathbf{P}(\mathcal{X})$. □

## 2.2 Semigroups which are unions of groups

An essential issue involved with using the Lauritzen–Spiegelhalter and HUGIN architectures is associated with the existence of a division operation. Semigroups are not in general equipped with such an operation, but if the semigroup is a union of groups, such a division can be partially defined. We now assume that our semigroup $\mathcal{V}$ of valuations is a disjoint union of groups;

$$\mathcal{V} = \bigcup_{\lambda \in \mathcal{L}} G_\lambda,$$

where $G_\lambda$ are groups and $G_\lambda \cap G_\mu = \emptyset$ if $\lambda \neq \mu$. If we let $\sigma(u)$ denote the label of the group to which $u$ belongs and $e_\lambda$ the unit in $G_\lambda$ we get

$$(e_\lambda \otimes e_\mu) \otimes (e_\lambda \otimes e_\mu) = (e_\lambda \otimes e_\mu)$$

whereby $(e_\lambda \otimes e_\mu)$ must be the unit in the group with label $\sigma(e_\lambda \otimes e_\mu)$. If we now let

$$\lambda \wedge \mu = \sigma(e_\lambda \otimes e_\mu),$$

it easily follows that $(\mathcal{L}, \wedge)$ defines an idempotent semigroup. In fact, if the semigroup $\mathcal{V}$ is idempotent as in Example 3, then we have $(\mathcal{L}, \wedge) = (\mathcal{V}, \otimes)$ and the label mapping is the identity. In general, the label map is a semigroup homomorphism so that we have

$$\sigma(u \otimes v) = \sigma(u) \wedge \sigma(v).$$

Also, a natural reflexive partial order on the labels $\mathcal{L}$ is defined by

$$\lambda \preceq \mu \iff \lambda \wedge \mu = \lambda.$$

As a consequence, if $\sigma(u) \preceq \lambda$ then

$$u \otimes e_\lambda = u \otimes e_{\sigma(u)} \otimes e_\lambda = u \otimes e_{\sigma(u)} = u. \tag{5}$$

**Example 4** The semigroup $(\mathbf{P}, \cdot)$ of Example 1 is a union of groups where $f$ and $g$ are in the same group if and only if they are positive on the same region of $\mathcal{X}$. Thus here $\sigma(f)$ can be identified with the support of $f$ as a subset of $\mathcal{X}$. And this holds true even with respect to the composition rule for the labels

$$\sigma(f) \wedge \sigma(g) = \sigma(f \cdot g) = \text{supp}(f \cdot g) = \text{supp}(f) \cap \text{supp}(g).$$

Hence, in this example the semigroup $(\mathcal{L}, \wedge)$ is isomorphic to $(\mathcal{P}(\mathcal{X}), \cap)$. $\square$

Because of this example, we will in general use the term *support function* for the map $\sigma$ and say that $\sigma(u)$ is the *support* of $u$ also in a general semigroup $(\mathcal{V}, \otimes)$.

Generally, division cannot be suitably defined globally, even in a semigroup which is a union of groups, corresponding to the fact that division with zero must be avoided in the semigroup $\mathbf{P}$ of Example 1. But we can define it partially as

$$u \oslash v = u \otimes v^{-1} \quad \text{if } \sigma(u) \preceq \sigma(v), \tag{6}$$

where $v$ is the inverse of $v$ within the group that contains $v$. It then holds that

$$(u \oslash v) \otimes v = u \tag{7}$$

since

$$(u \oslash v) \otimes v = u \otimes v^{-1} \otimes v = u \otimes e_{\sigma(v)} = u,$$

where we have used (5).

If now $w \in \mathcal{V}$ we have that $(u \otimes w) \oslash v$ is well defined, but not necessarily $u \otimes (w \oslash v)$ which only makes sense if also $\sigma(w) \preceq \sigma(v)$. If both expressions make sense, they are equal. In particular we always have

$$(u \otimes v) \oslash v = u \otimes (v \oslash v) = u \otimes e_{\sigma(v)}$$

but note that this is not necessarily equal to $u$. For example, if the semigroup is the set of pairs of non-negative numbers with pointwise multiplication, then

$$\{(2,0) \otimes (0,3)\} \oslash (0,3) = (2,0) \otimes \{(0,3) \otimes (0,1/3)\} = (2,0) \otimes (0,1) = (0,0) \neq (2,0).$$

Alternatively, one could have defined division globally by the expression in (6), but then (7) would then still only hold for $\sigma(u) \preceq \sigma(v)$.

An object satisfying a relation similar to (7) has been termed a *continuer* by Shafer (1991) in a setting which is more general than this.

**Example 5** The semigroup $(\mathcal{P}, \cap)$ of Example 3 is itself idempotent. In a trivial fashion this is a union of groups each having only one element. Thus, as then $v^{-1} = v$, the division $u \oslash v$ is defined for $u \preceq v$ and then $u \oslash v = u \otimes v = u$. $\square$

## 2.3 Separative semigroups

If a semigroup is not by itself a union of groups, it might be possible to embed it homomorphically into a semigroup which is, and then perform the division in the larger semigroup.

Hewitt and Zuckermann (1956) say that a semigroup $(\mathcal{V}, \otimes)$ is *separative* if it holds that

$$u \otimes v = u \otimes u = v \otimes v \implies u = v.$$

We refer to this paper and Chapter 4.3 of Clifford and Preston (1961) for proofs in this section that are not given here.

All semigroups that so far have been considered in the propagation literature, in particular the semigroups in the previous examples, are indeed separative. We abstain from showing this in any detail.

It was shown by Hewitt and Zuckermann (1956) that *a semigroup $\mathcal{V}$ is separative if and only if it can be homomorphically embedded into a union of groups.* Hence we assume in the following that the semigroup of valuations $\mathcal{V}$ is separative and embedded into a semigroup $\bar{\mathcal{V}}$ which is a union of groups.

**Example 6** The semigroup $\mathcal{B}$ of belief functions over a given space is indeed separative, which is easily seen using the commonality function representation. However it is not by itself a union of groups. The ratio between two belief functions, represented by their commonality functions $Q_1$ and $Q_2$, should obviously be defined as

$$R(A) = [Q_1 \ominus Q_2](A) \propto \begin{cases} Q_1(A)/Q_2(A) & \text{if } Q_2(A) \neq 0 \\ 0 & \text{otherwise} \end{cases}$$

and this is only well-defined if

$$Q_2(A) = 0 \implies Q_1(A) = 0.$$

In general $R$ is not the commonality function of a belief function, because its Möbius transform may have negative values.

$R$ is the commonality function of what Kong (1986) has called a quasi-belief function. A *quasi-belief function* is simply the Möbius transform of any set function taking values in the unit interval. Under Dempster's rule of combination — which corresponds to multiplication of commonality functions — the set of quasi-belief functions form a semigroup $\mathcal{Q}$ which is a union of groups, and the semigroup $\mathcal{B}$ of belief functions is a subsemigroup of $\mathcal{Q}$, homomorphically embedded into it. See also Thoma (1989) for further details. □

For a general separative semigroup, the embedding into a union of groups can be made in a variety of ways. In particular it can be made using the so-called *Archimedean components* of a semigroup. We refer to Clifford and Preston (1961) for details.

# 3 Valuations with Domains

The notion that relates valuations to local computation is that of a *domain map*

$$d : \mathcal{V} \to \mathcal{P}(V)$$

that to each valuation $u \in \mathcal{V}$ associates its *domain* $d(u)$, a subset of a finite set $V$. We assume that the domain map satisfies

$$d(u \otimes v) \subseteq d(u) \cup d(v), \quad d(1) = \emptyset. \tag{8}$$

The intuitive idea behind the notion of a domain is that a valuation with a small domain can be easily represented and manipulated in a computer, whereas for a valuation with a large domain this may be difficult or even impossible.

**Example 7** The model example for the domain map is Example 1. Let $\mathbf{P}_A$ denote the elements of $\mathbf{P}$ that only depend effectively on $x$ through its $A$-coordinates, i.e. $f \in \mathbf{P}_A$ if and only if

$$x_v = y_v \text{ for all } v \in A \implies f(x) = f(y).$$

The domain of $f$ is then the smallest $A$ such that $f \in \mathbf{P}_A$:

$$d(f) = \bigcap_{A: f \in \mathbf{P}_A} A.$$

Thus, if $d(f) = B$, $f$ can be represented in a computer as a table of no more than $| \times_{v \in B} \mathcal{X}_v|$ numbers. Also we clearly have the relation (8) satisfied. □

The second fundamental idea related to local computation is the $A$-*marginal* of a valuation $u \in \mathcal{V}$. This is an operation that forces $u$ to have domain no larger than $A$. More precisely, the $A$-marginal $u^{\downarrow A}$ of $u$ satisfies

$$d(u^{\downarrow A}) \subseteq A \cap d(u). \tag{9}$$

Further, we assume that the marginalization has the following basic property

$$1^{\downarrow A} = 1, \quad (u^{\downarrow A})^{\downarrow B} = u^{\downarrow A \cap B} \tag{10}$$

and finally, the key to local computation is the assumption

$$(u \otimes v)^{\downarrow A} = u \otimes v^{\downarrow A} \quad \text{if } d(u) \subseteq A. \tag{11}$$

Note that (10) and (11) imply that

$$u^{\downarrow A} = u \quad \text{if } d(u) \subseteq A.$$

The assumption that the valuations form a commutative semigroup (1) with a domain map and marginalization satisfying the above properties (9), (10), and (11), is a variant of the Shafer–Shenoy axioms for local computation. The main difference is that we have chosen to work with valuations defined over the same space, thus representing the domain less explicitely. So our marginalizations are rather like projections.

**Example 8** The standard marginalization in the probability example is the *sum-marginal* defined for all $x \in \mathcal{X}$ as

$$f^{+A}(x) = \frac{1}{|\mathcal{X}_{V \setminus A}|} \sum_{y_{V \setminus A} \in \mathcal{X}_{V \setminus A}} f(x_A, y_{V \setminus A}).$$

But another marginal of interest is the *max-marginal* defined as

$$f^{\wedge A}(x) = \max_{y_{V \setminus A} \in \mathcal{X}_{V \setminus A}} f(x_A, y_{V \setminus A}).$$

Also, associated with retraction of evidence in expert systems (Cowell and Dawid 1992) we have the *out-marginal* defined as

$$f^{\diamond A}(x) = \frac{\sum_{y_{V \setminus A} \in \mathcal{X}_{V \setminus A}} \left\{ f(x_A, y_{V \setminus A}) \times \prod_{v \in V \setminus A} h_v(y_v) \right\}}{\sum_{y_{V \setminus A} \in \mathcal{X}_{V \setminus A}} \left\{ \prod_{v \in V \setminus A} h_v(y_v) \right\}},$$

where $h_v$ are functions on $\mathcal{X}_v$ with values in $\{0, 1\}$. A function $h_v$ represents the *evidence* that $x_v$ has been observed to be in a particular subset $E_v$ of $\mathcal{X}_v$ (the subset where $h_v = 1$). It is easy to verify that all these marginalizations satisfy the conditions (9), (10), and (11).

Note that the sum- and out-marginals are defined slightly different than usual as they involve a normalization. This is done to ensure that the marginal of the identity is the identity. The need for this variation is a consequence of the fact that we have represented domains implicitely as mentioned above. □

**Example 9** In the case of constraint propagation in Example 3, the marginalization is the set-projection

$$C^{\downarrow U} = C_U \times \mathcal{X}_{V \setminus U}$$

where

$$C_U = \{x_U \in \mathcal{X}_U \mid x_U = y_U \text{ for some } y \in C\}$$

and the domain of a constraint set $C \subseteq \mathcal{X}$ is the smallest set $U$ such that $C$ is a cylinder set, i.e. such that $C = C^{\downarrow U}$. Also here the conditions are easy to verify. □

**Example 10** In the case of belief functions described in Example 2, the marginalization and the domain map are more subtly defined. For $U \subseteq V$ we let $\mathcal{B}_U$ denote the elements of $\mathcal{B}$ with all focal elements being cylinder sets over $U$, i.e. having the form

$$C = C_U \times \mathcal{X}_{V \setminus U}.$$

The domain of $B$ is then the smallest $U$ such that $B \in \mathcal{B}_U$:

$$d(B) = \bigcap_{U : B \in \mathcal{B}_U} U.$$

The $U$-marginal of a belief function is most conveniently expressed in terms of the mass functions as

$$m^{\downarrow U}(A) = \begin{cases} \sum_{C : C_U = A_U} m(C) & \text{if } A = A_U \times \mathcal{X}_{V \setminus U} \\ 0 & \text{otherwise.} \end{cases} \tag{12}$$

Here it is less trivial to verify that the conditions are fulfilled. In particular, the condition (11) requires some algebraic manipulation. However, this fact is well-known, see for example Shafer and Shenoy (1988) or Hájek *et al.* (1992). □

If $\mathcal{V}$ is a union of groups we further assume that $d(u) = d(u^{-1})$, and if $\mathcal{V}$ is a separative semigroup embedded into a union of groups $\bar{\mathcal{V}}$, we assume that the domain map is extended to $\bar{\mathcal{V}}$, whereas this is not necessarily so for the marginalization. The marginalization is then assumed to satisfy the following modified version of (11): for all $u, v \in \mathcal{V}$ and $x \in \bar{\mathcal{V}}$ we assume that

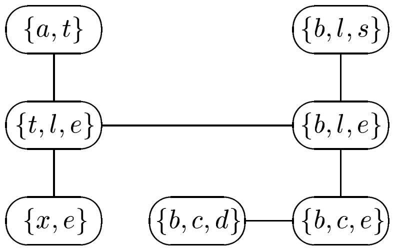
$$u = x \otimes v \text{ and } d(x) \subseteq A \implies u^{\downarrow A} = x \otimes v^{\downarrow A}. \tag{13}$$

# 4 Architectures for Local Computation

The basic structure behind local computation is that of a *junction tree* of subsets of a finite set $V$: an undirected tree $\mathcal{T}$ with subsets of $V$ as nodes, which has the special property that for any two nodes $A$ and $B$ of $\mathcal{T}$, the intersection $A \cap B$ is contained in all sets that lie on the path between $A$ and $B$. This implies the existence of an undirected, triangulated graph $\mathcal{G}$ over $V$ such that the nodes $\mathcal{C}$ of $\mathcal{T}$ include the maximal cliques of $\mathcal{G}$ and possibly subsets of these. Without loss of generality we can then assume that $V$ is the vertex set of $\mathcal{G}$ and that

$$V = \bigcup_{C \in \mathcal{C}} C.$$

A junction tree has also been termed join tree, Markov tree, clique tree, acyclic hypergraph, and other names in the literature.



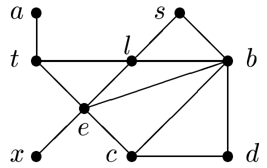




Figure 1: Junction tree with corresponding triangulated graph for an example with nine vertices: $V = \{a, b, c, d, e, l, s, t, x\}$.

Each subset $C \in \mathcal{C}$ is carrying a valuation $u_C$ with domain $d(u_C) \subseteq C$. The objects of interest to be computed are the marginals $u^{\downarrow C}, C \in \mathcal{C}$ of the *joint valuation*

$$u = \bigotimes_{C \in \mathcal{C}} u_C. \tag{14}$$

An example of a junction tree with corresponding triangulated graph is shown in Figure 1. The problem to be faced is that, typically, $d(u) = V$ and straight-forward computation and manipulation within such a large domain is inefficient or impossible, whereas computations that are *local,* i.e. concerned with valuations with domains within the cliques of $\mathcal{G}$, are possible.

We abstain here from explaining in detail how the initial junction tree with valuations is constructed from a knowledge base, although this in itself is not a trivial task. There is a considerable body of literature associated with this aspect. See for example the collections Oliver and Smith (1990) and Shafer and Pearl (1990), the book by Neapolitan (1990), or the review paper by Spiegelhalter *et al.* (1993). Also recent books by Almond (1995) and Jensen (1996) are recommended for further reading.

The algorithms have been modified and generalized in a variety of ways, to include continuous variables based on the conditional Gaussian distributions (Lauritzen 1992), to optimize decisions in influence diagrams (Shenoy 1992; Jensen *et al.* 1994), etc. These algorithms fall outside the framework considered in the present paper as the valuations do not form semigroups or the marginalization operations do not fulfil the assumptions.

A number of different but related architectures for computing the marginal of a joint valuation have been studied. Here we shall investigate three in further detail. All these involve successive local operations between nodes in the junction tree, also known as *message passing.* As pointed out by Shafer (1996), the main differences between the architectures lie in the form of messages passed, and in the scheduling of messages.

## 4.1 The Shafer–Shenoy architecture

This architecture for local computation is the most general of those we study, as it enables local computation for any semigroup of valuations with marginalization and domain map satisfying the Shafer–Shenoy axioms in the form (9), (10), and (11).

The messages are passed via a pair of *mailboxes* placed on each edge of the junction tree. If the edge connects $A$ and $B$, one mailbox is for messages that are $A$-outgoing and $B$-incoming, and one mailbox is for the reverse. The mailboxes can hold messages in the form of valuations with domains contained in $A \cap B$.

### 4.1.1 Scheduling of messages

All mailboxes are initialized as *empty.* When a message has been placed in a mailbox, the box is *full.*

A node $A$ in the junction tree is allowed to send a message to its neighbour $B$ if it has not done so before and if all $A$-incoming mailboxes are full except possibly the one which is for $B$-outgoing messages. So, initially only leaves of the junction tree are allowed to send messages. But as the message passing proceeds, other nodes will have their turn and eventually all mailboxes will be full, i.e. exactly two messages will have been passed along each branch of the junction tree.

### 4.1.2 Message types

There is only one message type in the Shafer–Shenoy architecture. The node $A$ sends a message to $B$ by computing the valuation $u_{A \to B}$ and storing it in the mailbox which is $B$-incoming and $A$-outgoing. Here $u_{A \to B}$ is given as

$$u_{A \to B} = \left\{ u_A \otimes \left( \bigotimes_{C \in \mathrm{ne}(A) \setminus B} u_{C \to A} \right) \right\}^{\downarrow B},$$

where $u_{C \to A}$ are the messages in the mailboxes which are $A$-incoming and $C$-outgoing and $\mathrm{ne}(A)$ are the neighbours of $A$ in the junction tree. In words, the valuation stored in $A$ is first combined with all messages that are incoming from directions other than $B$, and then marginalized to $B$. Clearly

$$d\left(u_{A \to B}\right) \subseteq \left\{ A \cup \left( \bigcup_{C \in \mathrm{ne}(A) \setminus B} (C \cap A) \right) \right\} \cap B \subseteq A \cap B$$

so the message passing operation makes sense.

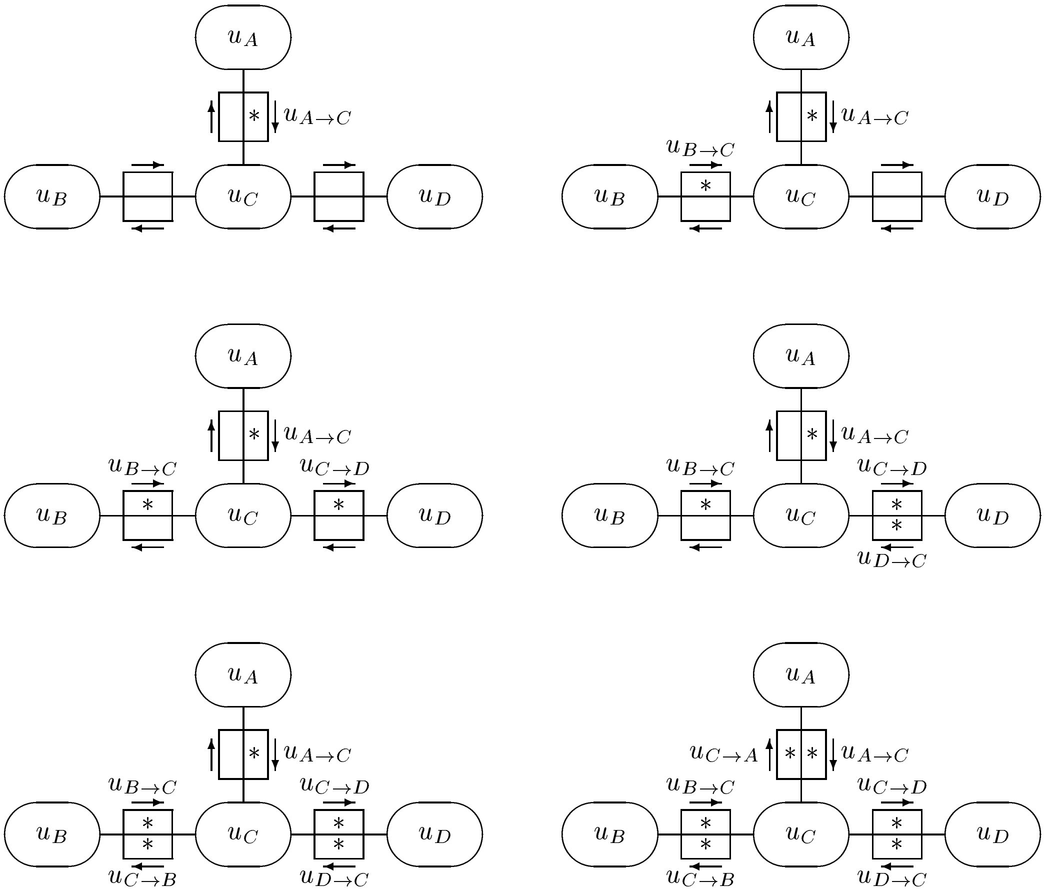



Figure 2: Message flow in the Shafer–Shenoy architecture. The clique $D$ plays the role of root. The figures should be read from left to right, top to bottom, beginning in the upper left corner. A message such as $u_{C\to D}$ is calculated via the formula $u_{C\to D} = \{u_C \otimes u_{A\to C} \otimes u_{B\to C}\}^{\downarrow D}$. The arrows mark the direction of each mailbox, represented as rectangular boxes along the edges of the junction tree. An asterisk marks that a mailbox is full.

It can be shown that — under the general assumptions made — when all mailboxes eventually are full, it holds for all nodes $A \in \mathcal{T}$ that

$$u^{\downarrow A} = u_A \otimes \left( \bigotimes_{C \in \mathrm{ne}(A)} u_{C\to A} \right).$$

Hence the goal of the local computation procedure has been achieved. The flow of messages is illustrated in Figure 2.

### 4.2 The Lauritzen–Spiegelhalter architecture

This architecture, which can be abstracted from the algorithm described by Lauritzen and Spiegelhalter (1988), differs from the previous architecture both in the scheduling of messages, but also in the types of messages passed.

#### 4.2.1 Scheduling of messages

Here the message passing is scheduled in two phases. One phase where the messages are *collected* towards a chosen *root* $R$ of the junction tree, and a second phase where the messages are *distributed* away from the root. More precisely, in the COLLECT phase a node $A$ is allowed to send to its neighbour towards the root $R$ if it has received messages from all neighbours that are further away from $R$ than $A$. When the root $R$ has received messages from all its neighbours, the DISTRIBUTE phase begins. In this phase the node $A$ sends messages to all its other neighbours as soon as it has received a message from its neighbour towards the root.

The difference between the schedulings is not as great as it might appear, as the Shafer–Shenoy scheduling effectively leads to choice of a root $R$ (the first node which has all incoming mailboxes full), and the messages are then virtually passed in two phases; see Dawid (1992) for this argument. However, in a specific implementation of the computations, say on a parallel computer, the two schedulings could perform quite differently.

It should be pointed out that it needs no extra assumptions to implement the Lauritzen–Spiegelhalter algorithm with Shafer–Shenoy scheduling. The only modification needed is to be aware that the first message along a branch is sent in the COLLECT phase, and the second message along a given branch is sent in the DISTRIBUTE phase.

#### 4.2.2 Message types

A more fundamental difference is that the messages passed in the two phases are different and different from those in the Shafer–Shenoy architecture. When a message is sent from $A$ to $B$ in the COLLECT phase, the valuations $u_A$ and $u_B$ change to $u_A^*$ and $u_B^*$ as follows

$$u_A^* = u_A \oslash u_A^{\downarrow B}, \quad u_B^* = u_B \otimes u_A^{\downarrow B}. \tag{15}$$

To ensure that the division in (15) is well-defined, we introduce the additional assumption that the support $\sigma(u)$ is never reduced under marginalization. More precisely, we assume that for all $u \in \mathcal{V}$ and all $D \subseteq V$ we have

$$\sigma(u) \preceq \sigma\left(u^{\downarrow D}\right) \tag{16}$$

in the partial order defined in Section 2.2. Note that

$$d(u_A^*) \subseteq d(u_A) \cup d(u_A^{\downarrow B}) \subseteq A.$$

Since we have

$$u_A^* \otimes u_B^* = u_A \otimes u_B,$$

the joint valuation is not affected by message passing

$$u = \bigotimes_{C \in \mathcal{C}} u_C = \bigotimes_{C \in \mathcal{C}} u_C^*,$$

apart from the fact that $u_C^*$ might be in $\bar{\mathcal{V}}$ unless $C = R$. When a message is sent from $B$ to $A$ in the DISTRIBUTE phase, the valuation only changes in the receiving node. The valuation $u_A$ changes to $u_A^*$ as

$$u_A^* = u_A \otimes u_B^{\downarrow A}, \tag{17}$$

whereas the valuation $u_B$ remains unchanged. It can be shown from (13) and the Shafer–Shenoy axioms that then we must have $u_A^* = u^{\downarrow A}$.

It is a consequence that when all messages have been sent, each node $C$ contains the marginal $u^{\downarrow C}$ of the joint valuation. The flow of messages is illustrated in Figure 3.

The main differences between this and the previous architecture are thus that the valuations stored in the nodes of the junction tree change during message passing, and that division is used in the COLLECT phase. The division needs extra assumptions and the Lauritzen–Spiegelhalter architecture is therefore not applicable to all problems where the Shafer–Shenoy architecture works.

On the other hand, the flexibility associated with two different message types makes this architecture suitable for generalization to such cases as those of optimizing decisions in influence diagrams, or calculating marginal moments of distributions with both discrete and continuous variables. As mentioned earlier, these cases are not immediately covered by the Shafer–Shenoy framework.

**Example 11** In constraint propagation we have $u^{\downarrow D} = u_D \times \mathcal{X}_{V \setminus D}$ as described in Example 9. As the support map in an idempotent semigroup is the identity, (16) becomes equivalent to the condition

$$u \cap \left(u^{\downarrow D}\right) = u,$$

which obviously holds. Hence, constraint propagation can be performed in a Lauritzen–Spiegelhalter architecture. □

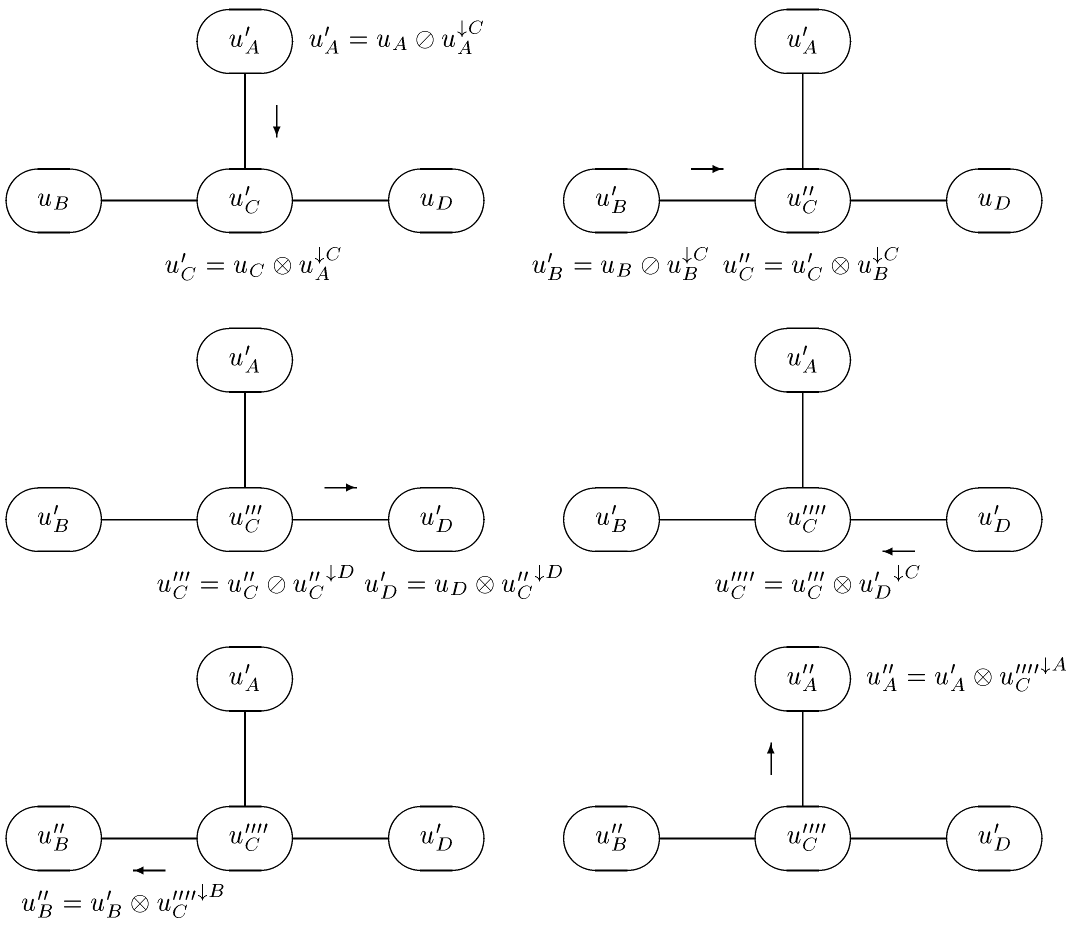



Figure 3: Message flow in the Lauritzen–Spiegelhalter architecture. The clique $D$ plays the role of root. The figures should be read from left to right, top to bottom, beginning in the upper left corner. An arrow indicates at each stage which message is being sent. The first three messages are sent in the COLLECT phase. Only the receiving node changes in the DISTRIBUTE phase.

**Example 12** For marginalization of belief functions with $\mathcal{B}$ considered as a subsemigroup of the quasi-belief functions $\mathcal{Q}$, the condition (16) is fulfilled. To see this, we use the expressions (3) and (12) to obtain that

$$\begin{aligned} Q^{\downarrow U}(D) &= \sum_{C:C\supseteq D} m^{\downarrow U}(C) \\ &= \sum_{C:C\supseteq D, C=C_U\times\mathcal{X}_{V\setminus U}} \sum_{E:E_U=C_U} m(E) \\ &\geq \sum_{E:E\supseteq D} m(E) = Q(D). \end{aligned}$$

The inequality sign follows from the fact that, for an arbitrary $E \supseteq D$ we have

$$E_U \times \mathcal{X}_{V\setminus U} \supseteq E \supseteq D$$

and therefore a term $m(E)$ is represented in the sum for all $E \supseteq D$.

Hence it follows that if $Q^{\downarrow U}(D) = 0$, we must also have $Q(D) = 0$ and $B^{\downarrow U}$ therefore divides $B$ within $\mathcal{Q}$. Also, the extended distributivity (13) can be shown to hold. As a consequence, propagation of belief functions can be implemented in a Lauritzen–Spiegelhalter architecture. □

**Example 13** For the out-marginal described in Example 8 the situation is more difficult. It may happen that (16) is violated. For example, if

$$\mathcal{X} = \{0,1\}^2, \quad f(x) = x_1 + x_2 - x_1x_2, \quad h_2(1) = 0,\ h_2(0) = 1,$$

then the out-marginal $f^{\diamond 1}(0) = 0$, but $f(0,1) = 1$. However, if we restrict the valuations to be strictly positive, corresponding to a (semi)group $\mathbf{P}^+$, the support condition is fulfilled as also exploited in Cowell and Dawid (1992). So retraction of evidence can typically only be implemented in a Lauritzen–Spiegelhalter architecture if all configurations in $\mathcal{X}$ have positive probability. If this is not the case, the Shafer–Shenoy architecture must be used. □

## 4.3 The HUGIN architecture

The final architecture to be considered is the one implemented in HUGIN for the probability case. It differs from the previous two in the representation of the joint valuation, and also in the messages passed.

The scheduling of messages can be made either by local control rules, as in the Shafer–Shenoy case, or in two phases, as in the Lauritzen–Spiegelhalter case. The semigroup of valuations and the corresponding marginalizations and domain maps are supposed to satisfy the same conditions as in the Lauritzen–Spiegelhalter case, in particular the condition (16).

### 4.3.1 Representation of the joint valuation

With each pair of neighbours $A$ and $B$ in $\mathcal{T}$ we associate their *separator* $S = A \cap B$. The set of separators $\mathcal{S}$ play an explicit role in the HUGIN architecture as they themselves hold valuations $u_S, S \in \mathcal{S}$.

Initially, we assume that the assignment of valuations to nodes in the junction tree is *supportive,* i.e.

$$\sigma\left(u_A^{\downarrow S}\right) \preceq \sigma\left(u_S\right) \quad \text{if } S \in \mathcal{S} \text{ is next to } A \text{ in } \mathcal{T}. \tag{18}$$

The typical assignment of valuations to universes has $u_S$ equal to the identity for all separators. This assignment is always supportive. The combination of (16) with (18) implies

$$\sigma(u_A) \preceq \sigma(u_S) \quad \text{if } S \in \mathcal{S} \text{ is next to } A \text{ in } \mathcal{T},$$

which then further implies

$$\sigma(\otimes_{C\in\mathcal{C}} u_C) = \bigwedge_{C\in\mathcal{C}} \sigma(u_C) \preceq \bigwedge_{S\in\mathcal{S}} \sigma(u_S) = \sigma(\otimes_{S\in\mathcal{S}} u_S). \tag{19}$$

The joint valuation is assumed to factorize over the junction tree as

$$u = \frac{\otimes_{C\in\mathcal{C}} u_C}{\otimes_{S\in\mathcal{S}} u_S},$$

which is well-defined because of (19). When the junction tree is initialized with identity valuations in all separators, the joint valuation is in fact given by the expression (14) as in the other architectures.

Finally, if $\mathcal{V}$ is not itself a union of groups we assume that for all connected subtrees $\mathcal{T}'$ with nodes $\mathcal{C}'$ and separators $\mathcal{S}'$ it holds that

$$u_{\mathcal{T}'} = \frac{\otimes_{C\in\mathcal{C}'} u_C}{\otimes_{S\in\mathcal{S}'} u_S} \in \mathcal{V}. \tag{20}$$

so that each such subtree defines a proper valuation and not just an element of the extended semigroup $\bar{\mathcal{V}}$. Also (20) is satisfied if the tree has identities in all separators.

### 4.3.2 Message types

The messages to be sent differ from the messages in the previous architectures by exploiting the separator valuations directly. On the other hand, there is only one type of message, whether this message be sent during the COLLECT or during the DISTRIBUTE phase. When a message is sent from $A$ to $B$ with separator $S = A \cap B$, the valuations change as

$$u_A^* = u_A, \quad u_S^* = u_A^{\downarrow S}, \quad u_B^* = u_B \otimes (u_S^* \oslash u_S). \tag{21}$$

The condition (18) ensures that the last expression in (21) has a meaning. The joint valuation is unchanged under message passing. This is seen by the following argument, where we have let $e_S^*$ be the unit in the group with label $\sigma(u_S^*)$:

$$\begin{aligned} u_A^* \otimes (u_B^* \oslash u_S^*) &= u_A \otimes [\{u_B \otimes (u_S^* \oslash u_S)\} \oslash u_S^*] \\ &= u_A \otimes e_S^* \otimes (u_B \oslash u_S) \\ &= u_A \otimes (u_B \oslash u_S). \end{aligned}$$

The last equality exploits (5) and (16).

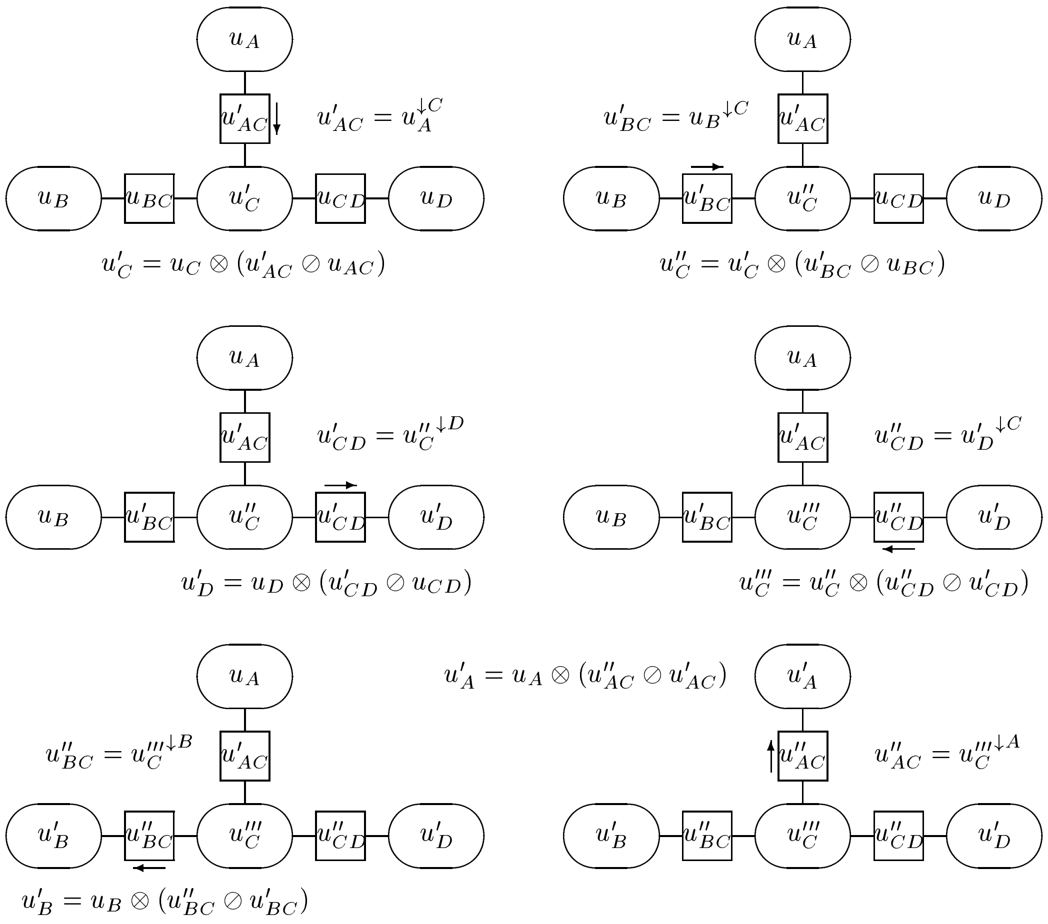



Figure 4: Message flow in the HUGIN architecture. The clique $D$ plays the role of root. Separators are represented by squares. The figures should be read from left to right, top to bottom, beginning in the upper left corner. The arrows indicate at each stage which message is being sent.

Obviously, the new separator valuation is a proper valuation in $\mathcal{V}$ as it is the marginal of a proper valuation. To see that the valuation $u_B^*$ in the receiving node is indeed also a member of $\mathcal{V}$ and not just in the extension $\bar{\mathcal{V}}$, we first use the condition (20) to get that $u_A \otimes (u_B \oslash u_S)$ — and hence its $B$-marginal — is in $\mathcal{V}$. Next, (13) gives

$$\{u_A \otimes (u_B \oslash u_S)\}^{\downarrow B} = u_A^{\downarrow B} \otimes (u_B \oslash u_S) = u_B \otimes (u_S^* \oslash u_S) = u_B^*$$

and thus $u_B^*$ is in $\mathcal{V}$. By a similar argument, the condition (20) can be seen to remain satisfied after the passing of a message. This ensures that computation can proceed.

Note that in effect, the extended semigroup $\bar{\mathcal{V}}$ is used only during the intermediate calculations; to represent the update ratios $u_S^* \oslash u_S$. Both separators and cliques hold proper valuations at all times.

The standard proofs in Jensen *et al.* (1990) or Dawid (1992) can now be used directly to show that after COLLECT and DISTRIBUTE, each node in the junction tree will hold the corresponding marginal of the original joint valuation. This is true for separators as well as for the original nodes in $\mathcal{T}$. The flow of messages is illustrated in Figure 4.

In the case of max-marginals in $\mathbf{P}$ or out-marginals in $\mathbf{P}^{+}$ the semigroup needs no extension and both of these cases are therefore directly amenable to the HUGIN architecture and, in fact, also implemented as standard propagation options in HUGIN. However, out-marginals in $\mathbf{P}$ cannot be implemented in a HUGIN architecture, see Example 13.

As the remarks in Example 9 and Example 12 show, constraint propagation as well as propagation of belief functions can be implemented in a HUGIN architecture. It is an issue for further research whether this can be exploited to improve the computational efficiency in these cases.


## Acknowledgements

This research was supported in part by the Danish Research Councils through the PIFT programme and by ESPRIT Basic Research Action 6156 (DRUMS II). The authors have benefited from communications with Glenn Shafer, Prakash Shenoy, and Nic Wilson, as well as comments from two anonymous referees.


## References


Almond, R. (1995). *Graphical belief modelling.* Chapman and Hall, London.

Andersen, S. K., Olesen, K. G., Jensen, F. V., and Jensen, F. (1989). HUGIN – A shell for building Bayesian belief universes for expert systems. In *Proceedings of the 11th international joint conference on artificial intelligence*, pp. 1080–5. Morgan Kaufmann, San Mateo. Also reprinted in Shafer and Pearl (1990).

Clifford, A. H. and Preston, G. B. (1961). *The algebraic theory of semigroups.* American Mathematical Society, Providence, RI.

Cowell, R. G. and Dawid, A. P. (1992). Fast retraction of evidence in a probabilistic expert system. *Statistics and Computing*, **2**, 37–40.

Dawid, A. P. (1992). Applications of a general propagation algorithm for probabilistic expert systems. *Statistics and Computing*, **2**, 25–36.

Hájek, P., Havránek, T., and Jiroušek, R. (1992). *Uncertain information processing in expert systems.* CRC Press, Boca Raton.

Hewitt, E. and Zuckermann, H. S. (1956). The $l_1$-algebra of a commutative semigroup. *Transactions of the American Mathematical Society*, **83**, 70–97.

Jensen, F., Jensen, F. V., and Dittmer, S. L. (1994). From influence diagrams to junction trees. In *Proceedings of the 10th conference on uncertainty in artificial intelligence*, (ed. R. L. de Mantaras and D. Poole), pp. 367–73. Morgan Kaufmann, San Mateo.

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

Jensen, F. V., Lauritzen, S. L., and Olesen, K. G. (1990). Bayesian updating in causal probabilistic networks by local computation. *Computational Statistics Quarterly*, **4**, 269–82.

Kong, A. (1986). *Multivarate belief functions and graphical models.* PhD thesis, Harvard University, Department of Statistics.

Lauritzen, S. L. (1992). Propagation of probabilities, means and variances in mixed graphical association models. *Journal of the American Statistical Association*, **86**, 1098–108.

Lauritzen, S. L. and Spiegelhalter, D. J. (1988). Local computations with probabilities on graphical structures and their application to expert systems (with discussion). *Journal of the Royal Statistical Society, Series B*, **50**, 157–224.

Neapolitan, E. (1990). *Probabilistic reasoning in expert systems.* John Wiley and Sons, New York.

Oliver, R. M. and Smith, J. Q. (1990). *Influence diagrams, belief nets and decision analysis.* John Wiley and Sons, Chichester.

Shafer, G. (1976). *A mathematical theory of evidence.* Princeton University Press, Princeton, N.J.

Shafer, G. (1991). An axiomatic study of computation in hypertrees. Technical Report WP–232, School of Business, University of Kansas.

Shafer, G. (1996). *Probabilistic expert systems.* Society for Industrial and Applied Mathematics, Philadelphia.

Shafer, G. and Pearl, J. (ed.) (1990). *Readings in uncertain reasoning.* Morgan Kaufmann, San Mateo.

Shafer, G. and Shenoy, P. P. (1988). Local computation in hypertrees. Technical Report WP–201, School of Business, University of Kansas.

Shenoy, P. P. (1992). Valuation-based systems for Bayesian decision analysis. *Operations Research*, **40**, (3), 463–84.

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

Spiegelhalter, D. J., Dawid, A. P., Lauritzen, S. L., and Cowell, R. G. (1993). Bayesian analysis in expert systems (with discussion). *Statistical Science*, **8**, 219–247 and 204–283.

Thoma, H. M. (1989). *Factorization of belief functions.* PhD thesis, Harvard University, Department of Statistics.