# Finding w ≥ 2 Probable Explanations in a Valuation Based Systems

S.T. Wierzchoń, M.A. Kłopotek
Institute of Computer Science
Polish Academy of Sciences
01-267 Warszawa, Ordona 21, Poland
e-mail: {stw, klopotek}@ipipan.waw.pl

**Abstract:** A method for finding a number of best explanations in so-called valuation based system is presented. Although the algorithm is mentally based on the idea proposed by Nillson [8], it can be used in much broader class of problems, and some savings in computations are discussed.

## 1. Introduction

The idea of graphical expert systems, called also bayesian networks (BN) was initiated by Pearl [11] and worked-out by Lauritzen and Spiegelhalter [7]. Its generalization to different uncertainty formalisms was proposed by Shenoy [14] under the name valuation based systems (VBS in short).

In the probabilistic context knowledge about interrelationships among variables from a fixed set $X$ is represented on the qualitative level by a directed acyclic graph $G$ and on the quantitative level by a set of conditional probabilities (see Section 2 for a rigorous definition).

Finding the $w \geq 1$ probable explanation in a BN means an optimization problem to identify a set of configurations $x^1,..., x^w$ such that $P(x^1) \geq ... \geq P(x^w)$ where $P$ is the joint probability distribution represented by the BN. We call $x^1$ the most probable explanation (MPE), and $x^2,..., x^w$ - probable explanations or probable configurations. Partial ordering of the probable configurations is important in several application domains like diagnosis, prognosis evaluation, or assessment of certain design methodologies [17], [6]. Its importance in such formalism like dynamical programmig (a special case of VBS) is obvious - cf. [19].

The problem of finding total order of configurations is NP-hard [22], [9]. Also the problem of finding several most probable configurations is generally NP-hard [2]. Previously two general approaches were proposed to handle the complexity of the task. One is to restrict the type of network to be dealt with such as single connected network [18], [4], or biparite graph [21]. Another approach is to shift the complexity to spatial domain in order to keep the computational complexity in a linear order. For example, Shimony and Charniak [16] convert a Bayesian belief network into a Weight Boolean Function Directed Acyclic Graph (WBFDAG) which permits the use of best-first search strategy in a WBFDAG. This method maintains a linear time with respect to the size of the graph, but the number of nodes in a WBFDAG could be exponential as compared to the original network. Several algorithms have been also

elaborated for the special task of finding only MPE, based on integer programming, and also on some modifications of the message passing algorithm of Shenoy and Shafer [13].

For singly connected networks, when no compromise is made between spatial and time complexity, an efficient algorithm for finding the first two probable configurations has been developed by Pearl [10]. The basic idea of Pearl's algorithm is that each node in a network is associated with a causal and diagnostic function, through which the largest values of these functions will propagate to each other for obtaining the information needed to compute $P(x^1)$. Sy, [18], proposed an efficient computational method for obtaining $w$ probable configurations in a singly connected network. The algorithm involves essentially a kind of message passing process.

However, methods developed for singly connected networks fail for multiply connected networks. One possible way out of this problem would be to create compound variables (combining several variables into one) in order to obtain a singly connected graph [11]. However, the computational load of processing the compound variable is exponentially increased. A similar approach would be to generalize Sy's approach to joint tree representation of a Bayesian network. (Consult [5] for appropriate definitions). To overcome the problems occuring when a junction tree must be computed, Wierzchoń, Kłopotek and Michalewicz [20] proposed another, genetic, approach to solve this problem within the general framework of VBS's. Some succesful solution to the problem of finding probable configurations in junction trees were proposed recently by Seroussi and Golmard [12] and Nillson [8]. Particulary the last solution seems to be specially attractive.

In this paper we study the problem of finding the first $w \geq 1$ probable explanations (configurations) in a VBS. To clarify the presentation we start from basic notions concerning Bayesian networks, being examples of VBS's. After recalling the algorithm for finding the MPE we extend it to the problem of finding the second probable explanation; by recursion this allows to find any $w$-th explanation. In our presentation we refer to the method proposed originally in [8] and we show that further improvements and generalizations can be achieved. First, important savings in computations are presented in Section 3. Second, a method for solving larger class of problems is discussed in Section 5. To be more precise, Nillson's algorithm was derived by referring to the HUGIN architecture [5] what implies the existence of so-called removal operator, dual to the combination operator (in case of probabilistic networks the removal is simple pointwise division). In our presentation we refer to the elementary architecture described by Shafer [13], hence the existence of the removal operator is not necessary.

## 2. Finding the Most Probable Explanation (MPE)

Let $X = \{X_1,...,X_n\}$ be a set of random variables with discrete domains $D_1,..., D_n$, respectively. Cartesian product of these domains, $D_1 \times ... \times D_n$, will be denoted as $D$; it represents the space of total configurations. A Bayesian network (BN for brevity) is

a pair <$G$, $P$> where $G$ is a directed acyclic graph whose nodes correspond to the variables, and $P = \{P_i\}_{i=1,...,n}$. Each $P_i$ is the conditional probability table, $P_i = \pi(X_i|pa_i)$, and $pa_i$ denotes the set of parents of $X_i$ in the graph $G$. The BN represents a joint probability distribution (*jpd*) over $X$

$$P(x_1,...,x_n) = \prod_{i=1}^{n} \pi(x_i|x.pa_i) \quad (1)$$

where $x = (x_1,...,x_n)$ abbreviates an assignment $(X_1=x_1,...,X_n=x_n)$, and $x.pa_i$ stands for the projection of the tuple $x$ over the subset of variables $pa_i$ (i.e. deletion of the elements of $x$ corresponding to the variables in $X\backslash pa_i$).

A common query to a BN is so-called belief assessment: find a marginal probability $P(x.S) = \sum\{P(y)|y \in D: y.S = x.S\}$, where $S \subseteq X$. Another very important problem is that of MPE: find such a tuple $x^*$ that $P(x^*) = \max_{x \in D} P(x)$.This last problem can be solved by using dynamic programming technique [3], and its solution consists of two parts: in the *backward part* the maximum value of $P$ is computed and in the *forward part* the configuration $x^*$ is identified.

To do so, original BN is usually transformed into the secondary structure, called *junction tree* by Jensen [5] or Markov tree by Shafer [13]. Its nodes, denoted $N_j$ , $j$ = 1,...,$m$, (in fact, subsets of $X$) can always be ordered in such a way that the running intersection property holds: $(\forall k \geq 2)(\exists j<k)$: $N_j \supseteq N_k \cap (N_1 \cup ... \cup N_{k-1})$. This property means that the variables of a node $N_k$ also contained in previous nodes $(N_1,..., N_{k-1})$ are all members of one previous node, $N_j$. The set $S_k = N_k \cap (N_1 \cup ... \cup N_{k-1})$ is said to be *separator*: it separates the *residual* $R_k = N_k\backslash S_k$ from $(N_1 \cup ... \cup N_{k-1})\backslash S_k$. Any node $N_j$ containng $S_k$ with $j < k$ is called a *parent* of $N_k$ in the tree; similarly the node $N_k$ is called a *child* of $N_j$. Hence $N_1$ is the root of the junction tree and $N_m$ is a leaf node in the tree.

Now, the jpd over $X$ can be expressed as

$$P(x_1,...,x_n) = \prod_{j=1}^{m} t_j(x.N_j) \quad (2)$$

where $t_j$ is a table associated to $j$-th node of a junction tree representing given BN. If $N_j$ can be represented in the form $N_j = \{X_j\} \cup pa_i$ then $t_j(x.N_j) = \pi(x_j|x.pa_j)$, otherwise $t_j$ = 1 for each configuration $x.N_j$.

Assuming that the nodes of the tree are numbered such that the running intersection property is satisfied, the backward and forward parts can be described as follows. To keep the presentation clearer, we assume that each node, besides $t_j$, stores another table, $\tau_j$, and initially $t_j = \tau_j$, $j$ = 1,...,$m$.

*BACKWARD PART*: It consists of a number of send-absorb steps described below.

*Send-step*. Mark all leaves of the junction tree as current leaves; particularly $N_m$ is a current leaf. Let $N_p$ be the parent of $N_m$ in the junction tree. The current leaf sends a message to $N_p$ defined as follows

$$M_{m\to p}(x.S_m) = \max\ \{\tau_m(y.N_m) | \ y \in D:\ y.S_m = x.S_m\} \tag{3}$$

*Absorb-step*. The message is absorbed by the parent, i.e. the table $\tau_p(x.N_p)$ changes to the table $\tau_p(x.N_p) = \tau_p(x.N_p)\cdot M_{m\to p}(x.S_m)$.

The process of sending-absorbing messages is repeated for all remaining children of $N_p$. When $N_p$ has absorbed the messages from all its children, it is marked as current leaf. Generally, the sequence of send-absorb steps bears a resemblance to evaluation of a parse tree. Hence, a node can send its messege only if it is labelled as current leaf.

It is easy to observe that when the root of the tree has absorbed messages from its children then the table $\tau_1$ represents maximum of $P$ over the set $X\backslash N_1$ of variables, and the maximal value of $\tau_1$ equals maximum of $P(x_1,...,x_n)$ over $X$.

Let $T(N_j)$ be the subtree of the junction tree $T$ rooted at the node $N_j$, and let $Ch(N_j)$ be the set of children of $N_j$. Then

**Lemma 1** After the backward part any node $N_j$ (excluding all leaf nodes in $T$) stores the table $\tau_j(x.N_j)$ with the values

$$\begin{aligned}\tau_j(x.N_j) &= t_j(x.N_j)\cdot \prod_{N_k \in Ch(N_j)} M_{k\to j}(x.S_k) \\ &= t_j(x.N_j)\cdot\max\{\prod_{N_k \in T(N_j),k\neq j} t_k(y.N_k) \mid y \in D:\ y.N_j = x.N_j\}\end{aligned} \tag{4}$$

**Corollary 1**. After the backward part the root of the junction tree computes the table

$$\begin{aligned}\tau_1(x.N_1) &= t_1(x.N_1)\cdot\max\{\prod_{k=2}^{m} t_k(y.N_k) \mid y \in D:\ y.N_j = x.N_j\} \\ &= \max\ \{P(y) | \ y \in D:\ y.N_1 = x.N_1\}\end{aligned} \tag{5}$$

*FORWARD PART*: Suppose we have found an optimal configuration $x^*.(N_1 \cup ... \cup N_{j-1})$. To extend it to the configuration over $x.(N_1 \cup ... \cup N_j)$ we search for such an assignment $z$ of variables in $N_j$ that $z.S_j = x.S_j$ and $z$ maximizes the table $\tau_j$.

**Corollary 2**. Let $x^1$ be the MPE. Then
a) $M_{m\to p}(x^1.S_m) = \tau_m(x^1.N_m)$,
b) $\tau_j(x^1.N_j) = \max\{\prod_{N_k \in T(N_j),k\neq j} t_k(y.N_k) \mid y \in D:\ y.N_j = x^1.N_j\}$

$$= \prod_{N_k \in T(N_j), k \neq j} \tau_k(x^1.N_k) \cdot t_j(x^1.N_j) \quad (6)$$

### 3. Finding the second probable explanation

Supose $x^1$ is the MPE, and $x^2$ is a second probable configuration, i.e. $P(x^2) = \max_{x \in D \setminus \{x1\}} P(x)$. Suppose that $x^1$ and $x^2$ differ on at least one position corresponding to the variable, say, $X_k$. Let $N_j$ be such a node that $X_k \in N_j$ and $X_k$ does not belong to any node outside $T(N_j)$. Obviously, if $X_k \in N_j = S_j \cup R_j$ then $X_k$ must be a member of the residual $R_j$. Indeed, if $X_k \in S_j$ then $X_k$ must belong to the parent of $N_j$ also. Similarly, if $X_k \in R_j$ then - by running intersection property - it cannot belong to any node in $T \setminus T(N_j)$. Hence to find $P(x^2)$ we must review the maximal values over the sets

$$D^j = \{y \in D: y.(N_1 \cup \ldots \cup N_{j-1}) = x^1.(N_1 \cup \ldots \cup N_{j-1}), y.R_j \neq x^1.R_j\}, j = 2..m \quad (7)$$

$$D^1 = \{y \in D: y.N_1 \neq x^1.R_1\}, \text{ since } S_1 = \varnothing \text{ and } R_1 = N_1 \setminus S_1 = N_1 \quad (8)$$

Suppose first, we are looking for the maximum of $P$ over $D^1$. To do so we simply take the maximum over all configurations $y \in D$ such that $y.N_1 \neq x^1.N_1$, and we reconstruct the full configuration $x^2$ analogously as in the forward part already described.

Now let us focus on finding the maximum over $D^j$, where $j$ is such an index that $N_j$ is a child of the root node $N_1$. As stated above, a candidate for $x^2$ must be such a configuration $y$ that $y.N_j = x^1.N_j$, and $y.R_j \neq x^1.R_j$. By Corrolary 1

$$P(x^1) = t_1(x^1.N_1) \cdot \prod_{N_k \in Ch(N_1)} \tau_k(x^1.N_k) \quad (9)$$

On the other hand, the maximum over $D^j$ can be computed as:

$$\max_{y \in D^j} P(y) = t_1(x^1.N_1) \cdot \prod_{N_k \in Ch(N_1) \setminus \{N_j\}} \tau_k(x^1.N_k) \times \max\{\tau_j(y.N_j) | y.S_j = x^1.S_j, y.R_j \neq x^1.R_j\} \quad (10)$$

Hence, combining (9) and (10) we state that

$$\max_{y \in D^j} P(y) = \frac{P(x^1)}{\tau_j(x^1.N_j)} \cdot \max\{\tau_j(y.N_j) | y.S_j = x^1.S_j, y.R_j \neq x^1.R_j\} \quad (11)$$

Equation (11) demonstrates that the maximum over $D^j$ can be computed locally, by reviewing (and manipulating on) the values stored in the already computed table $\tau_j$.

The reasoning used in the derivation of (11) can be extended to any node $N_j$ located at any level of the join tree. Suppose namely that $N_p$ is the parent of $N_j$ in the tree, and suppose $N_s$ is a children of the root such that $N_j$, $N_p \in T(N_s)$. Denote $\Pi(N_s, N_p)$ the path starting from $N_s$ and ending at $N_p$, and denote by $\Sigma(N_s)$ the subset of $T(N_s)$ consisting of the nodes in $\Pi(N_s, N_p)$ and childrens of these nodes. The values assigned to the table $\pi_s$ can be computed recursively expanding the equation (10):

$$\pi_s(x^1.N_s) = \prod_{N_k \in \Sigma(N_s)\setminus(\Pi(N_s,N_p)\cup N_j)} \tau_k(x^1.N_k) \times \prod_{N_l \in \Pi(N_s,N_p)} t_l(x^1.N_l)\cdot\tau_j(x^1.N_j) \quad (12)$$

To find the maximum over all configurations $y \in D^j$, one should observe that for any node $N_k \in T\setminus T(N_j)$, $y.N_k = x^1.N_k$, and, for the node $N_j$, $y.S_j = x^1.S_j$, $y.R_j \neq x^1.R_j$, Hence, replacing $\pi_s(x^1.N_s)$ in equation (10) by the formula (12) we obtain the result (11). This gives another proof to the Lemma 2 below, originally presented in [8]

**Lemma 2**. The maximal value of $P$ over all configurations in $D^j$ equals

$$\max_{y\in D^j} P(y) = \frac{P(x^1)}{\tau_j(x^1.N_j)} \cdot \max\{\tau_j(y.N_j) | y.S_j = x^1.S_j,\ y.R_j \neq x^1.R_j\} \quad (13)$$

Our aim, however, was not to offer another proof of an already known lemma only. The importance of formula (13) will be discussed in Section 5.

Denote $M(i,k,c_{i+1,k})$ the maximal value of $P$ over all configurations in $D^k$ derived from the solution $x^i$; $c_{i+1,k}$ stands for the number of entry in the table $\tau_k$ for which that maximum has been achieved. Reviewing all values $M(1,k,c_{i+1,k})$ we identify the value $P(x^2)$, that is $P(x^2) = \max\{M(1,k,c_{i+1,k})\colon k = 1,...,m\} = M(1,k^*,c_{2,k^*})$. This corresponds to the backward part of the problem. To solve the forwad part we choose the subtree $T(N_{k^*})$ and we identify configurations for all variables in this tree excluding the variables in $N_j$ as they are determined by the configuration number $c_{k^*}$. Particularly, when $N_{k^*}$ is a leaf in $T$, forward part is not necessary: we simple append to the configuration $x^1.(N_1 \cup ... \cup N_{k^*-1})$ the configuration pointed by $c_{2,k^*}$. Note that this algorithm is much faster than that one proposed by Nillson in [8]. In his approach the configurations are identified for all variables in $\{N_j,.., N_m\}$, and in our approach we restrict to the subtree $T(N_j)$ only. To be precise, note also that after determination of $k^*$-th position we simply block $M(1,k^*,c_{2,k^*})$, i.e. we set it to zero or we remove it from the list of all $M$'s values.

## 4. Finding $w > 2$ most probable explanations.

Suppose first that we want to find the third configuration $x^3$, i.e. such a configuration that $P(x^3) = \max_{x\in D-\{x1,x2\}} P(x)$, and suppose that $x^2$ maximizes $M(1,k, c_{2,k})$ for some $k$

= $k$*. This means that $x^2.(N_1 \cup ... \cup N_{j-1}) = x^1.(N_1 \cup ... \cup N_{j-1})$, and thus $M(2,k, c_{3,k}) = M(1,k, c_{2,k})$ for $k < k$* and it is not necessary to compute the values of $M(2,k, c_{3,k})$ for $k < k$*. To find $M(2,k, c_{3,k})$ for $k \geq k$* we use Lemma 2 again. This time however we know that $x^1.R_{k*} \neq x^2.R_{k*}$, and $x^2.S_{k*} = x^1.S_{k*}$; these configurations corresponds to the entries $c_{1,k*}$ and $c_{2,k}$. It means that to compute $M(2,k, c_{3,k})$ by formula (11) we must search maximum over only such $y$'s that $y.S_k = x^1.S_k$, $y.R_j \neq x^1.R_j$, $y.R_j \neq x^2.R_j$, i.e. we simply ignore the $c_{1,k*}$-th and $c_{2,k*}$-th entries in the corresponding table $\tau_k$.

To identify $P(x^3)$ we must find maximum over the values $M(1,k,c_{1,k})$ for $k = 1,...,m$, $k \neq k$* and the values $M(2,k,c_{2,k})$ for $k = j,...,m$. If this maximum is achieved for some $k = j$* in $M(1,k,c_{1,k})$ we assume that $x^3.((N_1 \cup ... \cup N_m)\backslash T(N_{j*}) = x^1.((N_1 \cup ... \cup N_m)\backslash T(N_{j*})$ and we perform forward part on the variables belonging to the tree $T(N_{j*})$. On the other hand, if the maximum is achieved for some $k = j$* in $M(2,k,c_{2,k})$, we assume that $x^3.((N_1 \cup ... \cup N_m)\backslash T(N_{j*}) = x^2.((N_1 \cup ... \cup N_m)\backslash T(N_{j*})$ and we perform forward part on the variables belonging to the tree $T(N_{j*})$. Of course the configuration $x^3.N_{j*}$ is determined by the $c_{s,j*}$ where $s = 1$ or 2.

Suppose, we have identified $w$ most probable configurations and we want to find ($w$+1)-th configuration. To solve this problem we need the next information:

- All *active* values of $M(i,k_i,c_{i+1,k_i})$.
- All current solutions $x^1,...,x^w$.

To identify $P(x^{w+1})$ we first compute the entries $M(w+1,k,c_{w+2,k})$ for $k = j_w,...,m$ ($j_w$ is the node number for which $w$-th solution achieved its maximum), and we add them to the set of active values of $M$. The maximum over *all* active values is blocked and we store the numbers: $i$ (of previous solution from which this new solution is derived), $k_i$ and $c_{i+1,k_i}$.

Suppose $P(x^{w+1}) = M(i*,j*,c_{i*,j*})$. It means that $x^{w+1}.((N_1 \cup ... \cup N_m)\backslash T(N_{j*}) = x^{i*}.((N_1 \cup ... \cup N_m)\backslash T(N_{j*})$ and we must perform forward part on the variables belonging to the tree $T(N_{j*})$ and $x^{w+1}.N_{j*}$ is determined by the $c_{i*j*}$.

**5. Application to valuation-based systems**

Valuation-based systems (VBS, in short) were proposed in [14] to cover a broad family of problems. Roughly speaking, in a VBS nodes of a junction tree represent microuniverses with knowledge represented by so-called valuations. A BN is a good example of a VBN: probability tables are just the valuations. The process of sending-absorbing messages described in Section 1 can be imagined here in terms of coarsening-aggregation of knowledge. Hence we need two operators of marginalization and combination, respectively. That is, we postulate that for each subset $S \subseteq X$ there is a set $V_S$, elements of which are said valuations for $S$. The set $V$ =

$\cup\{V_S \mid S \subseteq X\}$ stands for the set of all valuations. If $\sigma \in V_S$, then we say that $S$ is the domain of $\sigma$.
By combination we understand a mapping $\otimes$: $V \times V \rightarrow V$ satisfying the three axioms (for more details see [15])

C1) If $\rho \in V_R$ and $\sigma \in V_S$ then $\rho \otimes \sigma \in V_{R \cup S}$,
C2) $\rho \otimes (\sigma \otimes \upsilon) = (\rho \otimes \sigma) \otimes \upsilon$
C3) $\rho \otimes \sigma = \sigma \otimes \rho$

Note that in case of a BN, combination corresponds to pointwise multiplication (eventually followed by normalization).

Further, we postulate that for each nonempty subset $S \subseteq X$ and for each variable $Y \in S$ there is a mapping $\downarrow(S\backslash\{Y\})$: $V_S \rightarrow V_{S\backslash\{Y\}}$, called marginalization such that for any valuation $\sigma \in V_S$ and for any two variables $Y, Z \in S$, [15]:

$$(\sigma^{\downarrow(S\backslash\{\{Y\})})^{\downarrow(S\backslash\{Y,Z\})} = (\sigma^{\downarrow(S\backslash\{\{Z\})})^{\downarrow(S\backslash\{Y,Z\})}$$

Note that in case of a BN, both summation and maximization are examples of the marginalization.

Now the problem of probabilistic reasoning can be represented in VBS language as follows: Find a marginal valuation

$$\mu = (\sigma_1 \otimes ... \otimes \sigma_m)^{\downarrow S}, \text{ for some } S \subseteq X \tag{14}$$

where the pair $(\downarrow, \otimes)$ is set to $(\sum, \prod)$, and the problem of finding MPE can be expressed as

$$\mu = (\sigma_1 \otimes ... \otimes \sigma_m)^{\downarrow \varnothing} \tag{15}$$

where the pair $(\downarrow, \otimes)$ is set to $(\max, \prod)$.

Taking as $(\downarrow, \otimes)$ the pair $(\max, \sum)$ we obtain the definition of nonserial dynamic programming problem studied by Bertele and Brioschi in [1]. Here, during the absorb step, a node $N_p$ changes its original valuation $\sigma_p$ into the new valuation

$$\tau_p = \sigma_p \otimes (\otimes\{M_{k \rightarrow p} | N_k \in Ch(N_p)\}) \tag{16}$$

and the message which a node $N_k$ sends to its parent is defined as

$$M_{k \rightarrow p} = (\tau_k)^{\downarrow(N_k \cap N_p)} \tag{17}$$

Both the formulas are translation of the corresponding formulas given in Section 2.

Now the idea of identifying $w \geq 2$ best configuration can be easily adopted to VBS's. When $\otimes$ admits an inverse operation (called removal operation in [15], and denoted $\circledR$), the equation (11) can be rewritten in the form

$$\sigma^{\downarrow D^j} = (\sigma(x^1) \circledR \tau_j(x^1.N_j)) \otimes \tau_j^{\downarrow D^j}(x^1.N_j) \qquad (18)$$

However when combination has no removal operator (examples of such situaltion are given in [15]), to perform such a task we must combine partial information as described in Section 3 (cf. equation (12))

**References**:


[1] Bertele, U., and Brioschi, F., *Nonserial Dynamic Programming*, Academic Press, New York, 1972

[2] Cooper G. The computational complexity of probabilistic inference using Bayesian belief networks. *Artif. Intell*. **42**,1990, 395-405

[3] Dechter, R. Bucket elimination: A unifying framework for probabilistic inference algorithms. In Uncertainty in AI (*UAI-96*), pp. 211-219

[4] Henrion M. An introduction to algorithm for inference in belief nets, *Uncertainty in Artificial Intelligence* **5**(1990).

[5] Jensen, F.V. *An Introduction to Bayesian Networks*. University College Press, London, 1996

[6] Keramaris, V., and Sy, B.K. A syllable based approach towards the intelligibility study of impaired speech. *Proc. 12th Annual International Conference of IEEE-EMBS*, Philadelphia, Penn, 1990.

[7] Lauritzen, S.L., and Spiegelhalter, D.J. Local computations with probabilities on graphical structures and their application to expert systems. *J. R. Statist. Soc*. **B**-50(1988)157-224

[8] Nilsson, D. An efficient algorithm for finding the *M* most probable configurations in Bayesian networks. Institute for Electronic Systems, Aalborg University, Denmark, Technical Report R-96-2020. A revised version of this report will appear in *Statistics and Computing*.

[9] Pal, N., Bezdek, J., and Hemanisha, R. Uncertainty measures for evidential reasoning I: a review. *Int. J.Approx. Reas*. **7**(1992)162-183.

[10] Pearl, J. Fusion, propagation and structuring in Bayesian networks. *Artif. Intel.* **28**(1986), 241-288.

[11] Pearl, J. *Reasoning in Intelligent Systems: Networks of Plausible Inference*, Morgan Kaufmann Publishers, 1988.

[12] Seroussi, B., and Golmard, J.L. An algorithm for finding the K most probable configurations in Bayesian networks. *Int. J. Approx. Reasoning* **8**(1994)17-50

[13] Shafer, G. *Probabilistic Expert Systems*, SIAM CBMS-NSF Regional Conference Series in Applied Mathematics, Philadelphia, 1996, vol. 67

[14] Shenoy, P.P. A valuation-based language for expert systems. *Int. J. Approx. Reas*., **3**(1989)383-411
[15] Shenoy, P.P. Conditional independence in valuation-based systems. *Int. J. Approx. Reas*., **10**(1993)203-234
[16] Shimony, S.E., and Charniak, E. A new algorithm for finding MAP assignments to belief networks. *Proc. Conference on Uncertainty in AI*, Cambridge, MA, 1990, 98-103.
[17] Sy, B.K. An AI based CAD/CAM approach to assess design methodology of a user-specified nonvocal communication device. *Proc. 15th Northeast Bioengineering Conference*, Boston, Mass., 1989, 185-192
[18] Sy, B.K. A recurrence local computation approach towards ordering composite beliefs in Bayesian belief networks. *Int. J. Approx. Reasoning*, **11**(1993)205-233
[19] Wierzchoń, S.T. Constraint propagation over restricted space of configurations, and its use in optimization. In: R.R. Yager, J. Kacprzyk and M. Fedrizzi (eds.) *Advances in the Dempster-Shafer Theory of Evidence*, J. Wiley, New York, 1994, 375-394
[20] Wierzchoń, S.T., Kłopotek, M.A., Michalewicz, M. Reasoning and facts explanation in valuation based systems, *Fundamenta Informatica*, **30**(1997)359-371
[21] Wu, T. A problem decomposition method for efficient diagnosis and interpretation of multiple disorders, *Proc. 114th. Symp.Computer Appl. in Medical Care*, 1990, 86-92.
[22] Yen J. Generalizing Dempster-Shafer theory to fuzzy sets. IEEE Trans. Syst.Man.Cybern. 20(3), 1990, 559-570.