AN APPROXIMATE PROPAGATION ALGORITHM FOR PRODUCT-BASED POSSIBILISTIC NETWORKS

Amen Ajroud, Mohamed Nazih Omri
FSM, Université de Monastir, Boulevard de l'Environnement 5019, Monastir, Tunisia

Salem Benferhat
CRIL, Université d'Artois, Rue Jean Souvraz SP18, 62300, Lens, France

Habib Youssef
ISITCOM, Université de Sousse, Route principale n$^\circ$1, 4011, Hammam Sousse, Tunisia

Keywords: Possibilistic networks, possibility distributions, approximate inference, DAG multiply connected.

Abstract: Product-Based Possibilistic Networks appear to be important tools to efficiently and compactly represent possibility distributions. The inference process is a crucial task to propagate information into network when new pieces of information, called evidence, are observed. However, this inference process is known to be a hard task especially for multiply connected networks. In this paper, we propose an approximate algorithm for product-based possibilistic networks. More precisely, we propose an adaptation of the probabilistic approach "Loopy Belief Propagation" (LBP) for possibilistic networks.

1 INTRODUCTION

Graphical models are important tools to efficiently represent and analyze uncertain information. Among these graphical representations, Bayesian Networks are particulary well defined and well applied. Possibilistic networks appear to be an alternative approach to model both uncertainty and imprecision. There are two kinds of possibilistic networks: Min-Based Possibilistic Networks and Product-Based Possibilistic Networks (Fonck, 1994). These two kinds of possibilistic networks only differ on the definition of possibilistic conditioning.

One of the most critical issue in probabilistic and possibilistic networks is the propagation of information through the graph structure. Unfortunately, it is known that both probabilistic an possibilistic inference are hard tasks specially when graphs are multiply connected (Cooper, 1990) and (Dagum & Luby, 1993). Indeed, when the networks are simply connected, the propagation of possibility degrees is not very difficult and can be achieved in a polynomial time. When the networks are large and multiply connected, several problems emerge: the inference requires an enormous memory size and calculation becomes complex and even impossible.

The inference algorithms can be classified in two categories (Guo & Hsu, 2002):

- Exact algorithms: these methods mostly exploit the independencies relations present in the network and transform the initial graph into a new graphical structure such that a junction tree. These algorithms give the exact posterior possibility degree.
- Approximate algorithms: they are alternatives of exact algorithms when the networks become very complex. They estimate the posterior uncertain degree in various ways.

In possibility theory frameworks, Exact algorithms have been defined for both min-based and product-based graphs. Moreover, an anytime approximate algorithm has been proposed for min-based possibilistic networks (Ben Amor & al, 2003). However, to the best of our knowledge, no approximate algorithm exists for product-based possibilistic networks.

In this paper we focus on product-based possibilistic network and propose a possibilistic inference algorithm which allows to determine an approximation of possibility degree of any variable of interest given some evidence. This possibilistic algorithm is an adaptation of a well-known probabilistic algorithm

"Loopy Belief Propagation" (Murphy & al, 1999) and (Bishop, 2006). Conditions for exact inference in multiply connected graphs through LBP have been demonstrated (Heskes, 2003). Without any transformation of the initial graph, the basic idea of this adaptation is to propagate evidence into network by passing messages between nodes. More precisely, messages are exchanged between each node and its parents and its children. We keep passing messages in the network until a stable state is reached (if ever).

The rest of this paper is organized as follows: first, we give a brief background on possibility theory and product-based possibilistic networks (section 2). Then, we present our possibilistic adaptation of "Loopy Belief Propagation" algorithm for product-based possibilistic networks (Section 3). Section 4 gives some experimental results.

2 POSSIBILITY THEORY AND POSSIBILISTIC NETWORKS

This section presents a short summary of Possibility Theory; for more details see (Dubois & Prade, 1988).

Let $V = \{A_1, A_2, ..., A_n\}$ be a set of variables. We denote by D_{A_i} the finite domain associated with the variable A_i. a_i denotes any instance of variable A_i.

$\Omega = \times_{A_i \in V} D_{A_i}$ represents the universe of discourse and ω, an element of Ω, is called an *interpretation* or *state*. The tuple $(\alpha_1, \alpha_2, ..., \alpha_n)$ denotes the interpretation ω, where each α_i is an instance of A_i.

2.1 Possibility Distribution

A possibility distribution π is a mapping from Ω to the interval $[0,1]$. The degree $\pi(\omega)$ represents the compatibility of ω with available piece of information. In other words, if π is used as an imperfect specification of a current state ω_0 of a part of the world, then $\pi(\omega)$ quantifies the degree of possibility that the proposition $\omega = \omega_0$ is true. By convention, $\pi(\omega) = 0$ means that $\omega = \omega_0$ is impossible, and $\pi(\omega) = 1$ means that this proposition is regarded as being possible without restriction. Any intermediary possibility degree $\pi(\omega) \in]0,1[$ indicates that $\omega = \omega_0$ is somewhat possible. A possibility distribution π is said to be *normalized* if there exists at least one state which is consistent with available pieces of information. More formally,

$$\exists \omega \in \Omega, \quad such\ that \ \ \pi(\omega) = 1.$$

Given a possibility distribution π defined on Ω, we can define a mapping grading the *possibility measure* of an event $\varphi \subseteq \Omega$ to the interval $[0,1]$ by,

$$\Pi(\varphi) = max\{\pi(\omega) : \omega \in \varphi\}.$$

Possibilistic conditioning (Dubois & Prade, 1988) consists in modifying our initial knowledge, encoded by a possibility distribution π, by the arrival of a new piece of information $\varphi \subseteq \Omega$. We will focus only on product-based conditioning, defined by:

$$\pi(\omega|\varphi) = \begin{cases} \frac{\pi(\omega)}{\Pi(\varphi)} & if\ \omega \in \varphi \\ 0 & otherwise \end{cases}$$

2.2 Product-based Possibilistic Networks

Possibilistic networks (Fonck, 1994), (Borgelt & al,1998), (Gebhardt & Kruse, 1997) and (Kruse & Gebhardt, 2005), denoted by ΠG, are directed acyclic graphs (DAG). Nodes correspond to variables and edges encode relationships between variables. A node A_j is said to be a parent of A_i if there exists an edge from the node A_j to the node A_i. Parents of A_i are denoted by U_{A_i}.

Uncertainty is represented at each node by local conditional possibility distributions. More precisely, for each variable A:

If A is a root, namely $U_A = \emptyset$, then $\max(\pi(a_1), \pi(a_2)) = 1$.

If A has parents, namely $U_A \neq \emptyset$, then $\max(\pi(a_1|U_A), \pi(a_2|U_A)) = 1$, for each $u_A \in D_{U_A}$, where D_{U_A} is the domain of parents of A.

Possibilistic networks are also compact representations of possibility distributions. More precisely, joint possibility distributions associated with possibilistic networks are computed using a so-called possibilistic chain rule similar to the probabilistic one, namely :

$$\pi_{\Pi G}(a_1, ..., a_n) = \prod_{i=1..n} \Pi(a_i \mid u_{A_i}),$$

where a_i is an instance of A_i and $u_{A_i} \in D_{U_{A_i}}$ is an instance of domain of parents of node A_i.

Example 1. *Figure 1 gives an example of a possibilistic network. Table 1 and 2 provide local conditional possibility distributions of each node given its parents.*

Table 1: Initial possibility distributions.

a	$\pi(a)$	b	a	$\pi(b\|a)$	c	a	$\pi(c\|a)$
a_1	0.3	b_1	a_1	1	c_1	a_1	0.2
a_2	1	b_1	a_2	0.4	c_1	a_2	1
		b_2	a_1	0	c_2	a_1	1
		b_2	a_2	1	c_2	a_2	0.3

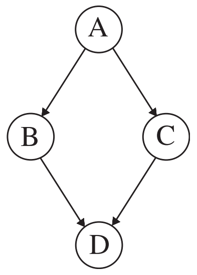

Figure 1: Example of a multiply connected DAG.

Table 2: Initial possibility distributions.

d	b	c	$\pi(d\|b,c)$	d	b	c	$\pi(d\|b,c)$
d_1	b_1	c_1	1	d_2	b_1	c_1	1
d_1	b_1	c_2	0	d_2	b_1	c_2	1
d_1	b_2	c_1	0.8	d_2	b_2	c_1	1
d_1	b_2	c_2	0.1	d_2	b_2	c_2	1

Using a possibilistic chain rule, we encode the joint distribution relative to A, B, C and D as follows :
$\forall a,b,c,d\ \ \pi(a,b,c,d) = \pi(a).\pi(b|a).\pi(c|a).\pi(d|b,c)$.
For instance,
$\pi(a_1,b_1,c_2,d_2) = \pi(a_1).\pi(b_1|a_1).\pi(c_2|a_1).\pi(d_2|b_1,c_2) = 0.3.1.1.1 = 0.3$

3 ADAPTATION OF LBP FOR POSSIBILISTIC NETWORKS

This section summarizes a direct adaptation of probabilistic LBP algorithm in the possibilistic framework. The probabilistic "Loopy Belief Propagation" is an approximate inference algorithm which applies the rules of Pearl Algorithm (Pearl, 1986) for multiply connected DAG. The basic idea of LBP is to propagate evidence into network by passing messages iteratively between nodes. We keep passing messages in the network until a stable state is reached (if ever). LBP is still a good alternative for exact inference methods specially when these latter meet difficulties to run. Let $Y_A = \{Y_1, Y_2, .., Y_n\}$, and $U_A = \{U_1, U_2, .., U_m\}$ be respectively the set of children and parents of node A.

With our adaptation, propagate an evidence E=e into Product-Based possibilistic algorithm is resumed to calculate the belief in a non-evidence node A, which is approximated by a conditional possibility degree of A, formally:

$$\forall a \in D_A,\ \ Bel(a) = \alpha.\lambda(a).\mu(a) \approx \pi(a \mid e), \qquad (1)$$

where α is a normalizing constant, $\lambda(a)$ and $\mu(a)$ are iteratively calculated.

At iteration *t*, $\lambda^{(t)}(a)$ and $\mu^{(t)}(a)$ represent the incoming messages to node A received from, respectively, children and parents of A:

$$\lambda^{(t)}(a) = \lambda_A(a).\prod_{j=1}^{n}\lambda_{Y_j}^{(t)}(a) \qquad (2)$$

$$\mu^{(t)}(a) = max_u \pi(a|u).\prod_{i=1}^{m}\mu_A^{(t)}(u_i) \qquad (3)$$

At iteration *t+1*, the node A sends outgoing messages (λ-messages and μ-messages) to, respectively, its parents and its children:

$$\lambda_A^{(t+1)}(u_i) = \beta\, max_a\, \lambda^{(t)}(a).[max_{u_{k:k\neq i}}\, \pi(a|u).\prod_{k\neq i}\mu_A^{(t)}(u_k)] \qquad (4)$$

$$\mu_{Y_j}^{(t+1)}(a) = \gamma\, \lambda_A(a).\prod_{i=1, i\neq j}^{n}\lambda_{Y_i}^{(t)}(a).\ \mu^{(t)}(a) \qquad (5)$$

Nodes are updated in parallel: at each iteration, all nodes compute their outgoing messages based on the input of their neighbors from the previous iteration. This procedure is said to converge if none of the beliefs in successive iterations changed by more than a small threshold (e.g. 10^{-3}) (Murphy and al, 1999).

We note that these formulas are similar to those corresponding to probabilistic framework but we use the maximum operator instead of the addition. The outline of our adaptation algorithm is as follows:

Algorithm: Product-Based possibilistic inference in multiply connected DAG

```
BEGIN
 N ← {nodes of network};
 for all nodes, initialization of :
 - $\lambda_{A_n}(a_n)$ by 1 or its observed value;
 - $\lambda$-messages and $\mu$-messages by vector 1 ;
 - Bel($a_m$) ← 0 ;
 M ← {number of non-observed nodes};
 t ← 1;
 convergence ← FALSE;
 while NOT convergence and t < max_itr
   for n ← 1 to length(N)
      calculate $\lambda^{(t)}(a_n)$ with the formula (2);
      calculate $\mu^{(t)}(a_n)$ with the formula (3);
   end
   for m ∈ M
      OldBel($a_m$) ← Bel($a_m$);
      calculate Bel($a_m$) with the formula (1);
   end
   if ∀ m ∈ M, Bel($a_m$)−OldBel($a_m$) < tol
      convergence ← TRUE;
```

```
    end
   if NOT convergence
      for n ← 1 to length(N)
         calculate for every parent $U_i$ of $A_n$; $\lambda_{A_n}^{(t+1)}(u_i)$
         with the formula (4);
         calculate for every child $Y_j$ of $A_n$; $\mu_{Y_j}^{(t+1)}(a_n)$
         with the formula (5);
      end
      t ← t + 1;
    end
  end
END
```

Example 2. *Given the Product-Based possibilistic network presented in Example 1, we now try to run our possibilistic adaptation. For example, we observe, as evidence, the node D=d_2 and we compute, progressively, possibility degree of each other node knowing the evidence. Table 3 shows for each iteration values obtained for A, B and C. Note that the algorithm converges after 3 iterations. We indicate, for comparison, the exact values given by an exact inference algorithm.*

Table 3: Posterior possibility degree.

iteration	t = 1	t = 2	t = 3	exact values
$\pi(a_1 \mid d_2)$	0.3	0.3	0.3	0.3
$\pi(a_2 \mid d_2)$	1	1	1	1
$\pi(b_1 \mid d_2)$	1	0.4	0.4	0.4
$\pi(b_2 \mid d_2)$	1	1	1	1
$\pi(c_1 \mid d_2)$	1	1	1	1
$\pi(c_2 \mid d_2)$	1	0.3	0.3	0.3

4 EXPERIMENTAL RESULTS

The implementation of possibilistic adaptation of LBP is based on the Bayes Net Toolbox (BNT) (Murphy & al, 1999) which is an open source Matlab package for directed graphical models. We used also the Possibilistic Networks Toolbox (PNT) which implements the exact inference algorithm : product-based Junction Tree (Ben Amor, 2002). The experimentation is performed on random possibilistic networks generated as follows:

Graphical Components. We used two DAGs structures generated as follows:

- STRUCTURE 1: In the first structure, the DAGs are multiply connected and generated randomly. We fixed the total number of nodes at 30. The cardinality of node instances is chosen randomly between 2 and 3. We can also change the number of parents for each node.
- STRUCTURE 2: In the second one, we choose special cases of DAGs where nodes are partitioned into two levels. This structure corresponds to well-known networks as the QMR (Quick Medical Reference) network (Jaakkola & Jordan, 1999). We have diseases level and findings level and we try to infer the distribution of diseases given a subset of findings. We chose specially this kind of network because it is known that in many cases, the probabilistic LBP did not converge to a stable state. We generate randomly this structure of graph with 30 nodes with instance cardinality $\in \{2, 3\}$.

Numerical Components. Once the DAG structure is fixed, we generate random conditional distributions of each node in the context of its parents. Then, we generate randomly the variable of interest.

4.1 Convergence

In this first experimentation we propose to evaluate the convergence of our approximate algorithm. Here, we focus on the number of iterations needed for the algorithm to converge. First, this experimentation is performed on 1000 random networks from STRUCTURE 1. Figure 2 shows that 46% of random networks generated need less than 5 iterations to converge. In the same way, 91% of networks generated need less than 20 iterations to reach convergence state. We consider that it is a satisfactory result with reasonable iterations number.

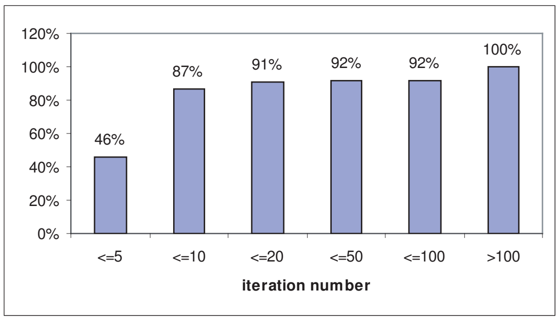

Figure 2: Iteration number for DAG of STRUCTURE 1.

We repeat the same experimentation with 1000 DAG having QMR structure (STRUCTURE 2). Figure 3 shows results of this experimentation. We note that for less than 20 iterations 85% of random networks converge. This result is slightly inferior to the first experimentation because QMR graph is less likely to converge than other structures.

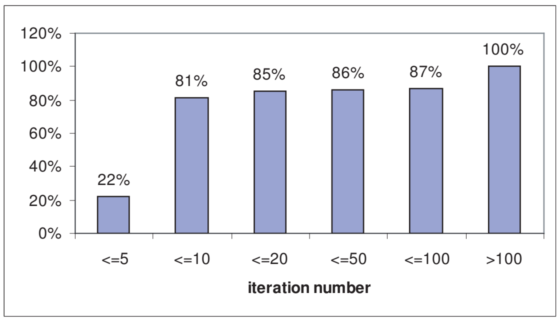

Figure 3: Iteration number for QMR DAG.

4.2 Exactness

We are now interested in the exactness of values generated by the approximate algorithm. Here, we measure the difference between approximate and exact values generated by respectively approximate and exact inference algorithm. This difference between values expresses how much our algorithm can approximate the exact one. More precisely, for one network, we compute possibility degrees obtained by our approximate algorithm and possibility degrees generated by an exact algorithm : the product-based junction tree algorithm (Ben Amor, 2002). Then we compute, for each node of the graph, the difference between exact and approximate possibility degrees. We consider that there is equality between 2 compared values if the difference between them is less than a fixed threshold (10^{-2}). finally, we count the number of equality and inequality cases and we obtain a percentage of each state.

Figure 4 summarizes the results of this experimentation applied to 1000 random graphs generated by STRUCTURE 1. 92% of numerical values issued from approximate algorithm coincide with those issued from exact algorithm. Note that here we did not discard the samples of non-convergence.

Figure 5 represents the result of the same experimentation using 1000 QMR graphs generated by STRUCTURE 2. In this case, only 83% of numerical values issued from approximate algorithm are considered to be exact. The results of these statistics for the two graph structures conclude that the approximate possibility degrees are closed with exact possibility degrees.

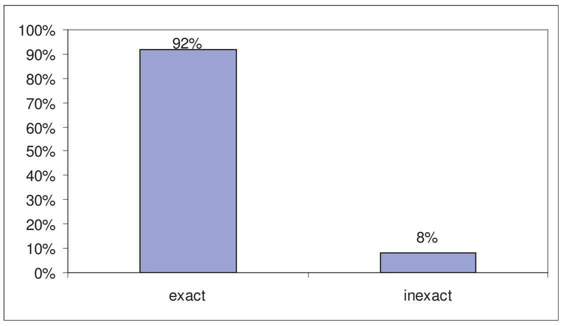

Figure 4: Average of compared values in randomly graph structure.

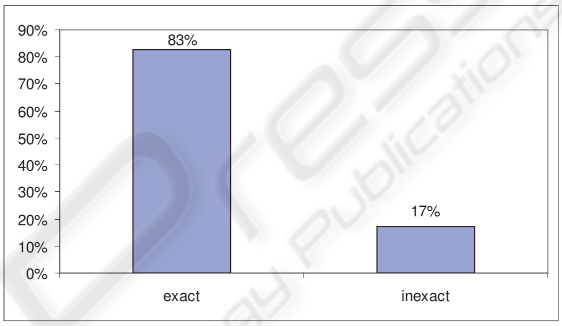

Figure 5: Average of compared values in QMR graph structure.

5 CONCLUSIONS

In this paper we proposed an approximate inference algorithm for Product-Based Possibilistic networks. It is an adaptation of probabilistic LBP algorithm in possibilistic framework. Without any structure transformation, the algorithm consists to propagate evidence, into multiply connected network, by passing messages between nodes. Our adaptation is an interesting alternative implementation for exact inference when this one fails on complex network (when sizes of cliques are large). Experimental results show that the possibility distributions generated by approximate algorithm are very near to exact possibility distributions. In a future work, we project to apply this new propagation algorithm by it's adaptation to handling interventions in possibilistic causal networks (Benferhat & Smaoui, 2007).

REFERENCES

Ben Amor, N. (2002). *Qualitative possibilistic graphical models : from independence to propagation algorithms*. Thèse de doctorat, Université de Tunis, ISG.

Ben Amor, N., Benferhat, S., & Mellouli, K. (2003) Anytime Propagation Algorithm for Min-based Possibilistic Graphs. *Soft Computing, A fusion of foundations, methodologies and applications, 8,* 150-161.

Benferhat, S., & Smaoui, S. (2007). Possibilistic Causal Networks for Handling Interventions: A New Propagation Algorithm. *The Twenty-Second AAAI Conference on Artificial Intelligence (AAAI'07),* pp.373-378.

Bishop, C. M., (2006). *Pattern Recognition and Machine Learning*. Springer.

Borgelt, C., Gebhardt, J., & Kruse, R. (1998). Possibilistic graphical models. *Proceedings of International School for the Synthesis of Expert Knowledge (ISSEK'98),* pp.51-68.

Cooper, G. F. (1990). Computational complexity of probabilistic inference using Bayesian belief networks. *Artificial Intelligence*, pp.393-405.

Dagum, P., & Luby, M. (1993). Approximating probabilistic inference in Bayesian belief networks is NP-hard. *Artificial Intelligence, 60,* 141-153.

Dubois, D., & Prade, H. (1988). *Possibility theory : An approach to computerized, Processing of uncertainty*. Plenium Press, New York.

Fonck, P. (1994). *Réseaux d'inférence pour le raisonnement possibiliste*. PhD thesis, Université de Liège, Faculté des Sciences.

Gebhardt, J., & Kruse, R. (1997). Background and perspectives of possibilistic graphical models, *Qualitative and Quantitative Practical Reasoning (ECSQARU/FAPR'97),* pp.108-121.

Guo, H., & Hsu, W. (2002). A survey of algorithms for real-time Bayesian network inference, *Joint Workshop on Real-Time Decision Support and Diagnosis Systems (AAAI/KDD/UAI-2002),* pp.1-12.

Heskes, T. (2003). Stable fixed points of loopy belief propagation are minima of the Bethe free energy. *Advances in Neural Information Processing Systems, 15,* 359-366.

Jaakkola, T.S., & Jordan, M.I. (1999). Variational probabilistic inference and the qmr dt network. *Journal of Artificial Intelligence Research, 10,* 291-322.

Kruse, R., & Gebhardt, J. (2005). Probabilistic Graphical Models in Complex Industrial Applications. *Fifth International Conference on Hybrid Intelligent Systems (HIS'05),* p.3.

Murphy, K. P., Weiss, Y., & Jordan, M. I. (1999). Loopy belief propagation for approximate inference: An empirical study. *Proceedings of the Fifteenth Conference on Uncertainty in Artificial Intelligence (UAI'99),* pp.467-475,

Pearl, J. (1986). Fusion, propagation and structuring in belief networks. *Artificial Intelligence, 29,* 241-288.