Efficient Maximum Projection of Database-Induced Multivariate Possibility Distributions

Christian Borgelt and Rudolf Kruse

Dept. of Information and Communication Systems
Otto-von-Guericke-University of Magdeburg
D-39106 Magdeburg, Germany
e-mail: christian.borgelt@cs.uni-magdeburg.de

Abstract

Current research in the domain of inference networks, probabilistic as well as possibilistic, focuses on learning such networks from data. Learning inference networks consists in finding a decomposition of a multivariate probability or possibility distribution that is induced by a database of sample cases. An operation to be carried out several times during the execution of common learning algorithms is the computation of the projection of the database-induced probability or possibility distribution to a subset of the database attributes. This operation is trivial for the probabilistic case, but turns out to be a problem for the possibilistic one, since ad hoc approaches lead to wrong results or are very inefficient. In this paper we suggest an efficient method to compute maximum projections of database-induced possibility distributions, making real world possibilistic network learning feasible in the first place.

1. Introduction

Inference networks, especially probabilistic networks, are already well established as powerful tools for reasoning under uncertainty. The idea underlying them is that reasoning in multi-dimensional domains, which tends to be infeasible in the domains as a whole — and the more so, if uncertainty and/or imprecision are involved —, can be made feasible by a decomposition of the available knowledge. The main advantage of such a decomposition is that it reduces the reasoning process to computations in lower-dimensional subspaces.

The theory of decomposition techniques for uncertain and imprecise knowledge is well developed. For example, decomposition based on dependence and independence relations between variables has extensively been studied in the field of graphical modeling [12]. Some of the best-known approaches are Bayesian networks [17], Markov networks [14], and the more general valuation-based networks [19]. But recently possibilistic networks [8] also gained some attention. All of these approaches led to the development of efficient implementations, for example HUGIN [1], PULCINELLA [18], PATHFINDER [10] and POSSINFER [8].

A large part of recent research in the domain of inference networks, probabilistic as well as possibilistic, has been devoted to learning them from data [4], [11], [7]. Basically, such learning consists in finding a decomposition of a database-induced multivariate probability or possibility distribution. Any algorithm for this task needs to compute several times the projection of this distribution to a subset of the database attributes, mainly in order to determine the distributions of the decomposition. This operation is trivial for the probabilistic case, but turns out to be a problem for the possibilistic one, since ad hoc approaches lead to wrong results or are very inefficient. In this paper we suggest an efficient method to compute maximum projections of database-induced possibility distributions, thus providing an important ingredient to make real world possibilistic network learning feasible.

2. Computing Maximum Projections

Before we can state clearly the problem we try to solve, we need some formal underpinnings. We start by defining a tuple (precise as well as imprecise), a relation, and projections of tuples as well as relations. We then introduce the notion of a database and that of a database-induced possibility distribution.

The definition of the induced possibility distribution is usually the first step in learning a possibilistic inference network from a given database. The decomposition part of the learning process involves computing maximum projections of the induced possibility distributions. Unfortunately it is not as easy as it appears at first sight to find an efficient algorithm for this task. Hence computing maximum projections is actually a problem, which we try to solve in this paper.

Definition 1: Let $X = \{A_i, \ldots, A_n\}$ be a set of attributes with domains $\mathrm{dom}(A_i) = \{a_1^{(i)}, \ldots, a_{m_i}^{(i)}\}$. A *tuple* t over X is a mapping

$$t : X \to \bigcup_{A \in X} 2^{\mathrm{dom}(A)}$$

with the constraint $\forall A \in X : t(A) \subseteq \mathrm{dom}(A) \wedge t(A) \neq \emptyset$. The set of all tuples over X is denoted $T(X)$. □

We write tuples similar to the usual vector notation. For example, a tuple t over $\{A, B, C\}$ which maps A to $\{a_1\}$, B to $\{b_2, b_4\}$ and C to $\{c_1, c_3\}$ is written $t = (A \to \{a_1\}, B \to \{b_2, b_4\}, C \to \{c_1, c_3\})$. If an implicit order is fixed, the attributes can be omitted.

With the above definition a tuple can assign a *set* of values to an attribute, instead of only one. The idea is that we want tuples to represent *imprecise* (i.e. multi-valued) information about the state of the world. We make this more precise below, where we discuss how a database can be interpreted as a description of a multivariate possibility distribution.

Definition 2: A tuple t over an attribute set X is called *precise*, iff $\forall A \in X : |t(A)| = 1$. Otherwise it is called *imprecise*. The set of all precise tuples over X is denoted $T_{\mathrm{prec}}(X)$. □

Note that a tuple, as defined in definition 1, can be seen as representing a set of possible precise tuples. We make use of this view when introducing the possibility distribution induced by a given dataset.

Since we head at describing decompositions of possibility distributions into distributions on lower-dimensional subspaces, we need to define the notion of a projection of a tuple to a subset of the set of attributes it is defined upon.

Definition 3: If t is a tuple over an attribute set X and $Y \subseteq X$, then $t_Y = \mathrm{proj}_Y^X(t)$ denotes the *projection* or *restriction* of the tuple t to Y. The mapping t_Y assigns (sets of) values only to the attributes in Y. (Hence t_Y is a tuple over Y.) □

We now turn from tuples to relations.

Definition 4: A *relation* R over an attribute set X is a set of tuples over X. □

Definition 5: If R is a relation over X and $Y \subseteq X$, then the *projection* $R_Y = \mathrm{proj}_Y^X(R)$ of R from X to Y is defined as

$$R_Y = \mathrm{proj}_Y^X(R) = \{s \in T(Y) \mid \exists t \in R : s \equiv \mathrm{proj}_Y^X(t)\}.$$

(Hence R_Y is a relation over Y.) □

For the type of problem we are dealing with, a simple relation is not enough. In a relation, as it is a *set* of tuples, each tuple can appear only once. In contrast to this, in a database of sample cases a given tuple can appear several times, reflecting the relative frequency of its occurrence. Since we cannot dispense with this frequency information, we need a mechanism to represent the number of occurrences of a tuple.

Definition 6: A *database* D over an attribute set X, is a tuple (R, w_R), where R is a relation over X and w_R is a function mapping each tuple in R to a natural number, i.e. $w_R : R \to \mathbb{N}$. □

The function w_R is meant to express the number of occurrences of a tuple $t \in R$ in a set of sample cases. We speak of $w_R(t)$ as indicating the *weight* of a tuple t.

We interpret a given database as a description of a multivariate possibility distribution. This interpretation is based on the context model [5], [13], which provides a well-founded justification of the semantics of a degree of possibility. In this model possibility distributions are seen as information-compressed representations of (not necessarily nested) random sets, a degree of possibility as the one-point coverage of a random set [16].

More precisely, let Ω be the set of all possible states of the world, $\omega_0 \in \Omega$ the actual (but unknown) state of the world, $(C, 2^C, P)$, $C = \{c_1, \ldots, c_k\}$, a finite probability space, and $\gamma : C \to 2^\Omega$ a set-valued mapping. C is seen as a set of contexts that have to be distinguished for a set-valued specification of ω_0. The contexts are supposed to describe different physical and observation-related frame conditions. $P(\{c\})$ is the (subjective) probability of the (occurrence or selection of the) context c.

A set $\gamma(c)$ is assumed to be the *most specific correct set-valued specification* of ω_0, which is implied by the frame conditions that characterize the context c. By "most specific set-valued specification" we mean that $\omega_0 \in \gamma(c)$ is guaranteed to be true for $\gamma(c)$, but is not guaranteed for any proper subset of $\gamma(c)$. The resulting *random set* $\Gamma = (\gamma, P)$ is an imperfect (i.e. imprecise *and* uncertain) specification of ω_0. Let π_Γ denote the *one-point coverage of* Γ (the *possibility distribution induced by* Γ), which is defined as

$$\pi_\Gamma : \Omega \to [0, 1], \quad \pi_\Gamma(\omega) \mapsto P\left(\{c \in C \mid \omega \in \gamma(c)\}\right).$$

In a complete modeling, the contexts in C must be specified in detail, so that the relationships between all contexts c_j and their corresponding specifications $\gamma(c_j)$ are made explicit. But if the contexts are unknown or ignored, then $\pi_\Gamma(\omega)$ is the total mass of all contexts c that provide a specification $\gamma(c)$ in which ω_0 is contained, and this quantifies the *possibility of truth* of the statement "$\omega = \omega_0$" [6], [8].

We apply the context model directly to a database. The set of all precise tuples over X is the set Ω of all possible states of the world. Each sample case is seen as associated with a context, each of which is assumed to have equal probability. Since the weight function

counts sample cases, a database already combines contexts, namely those which lead to the same tuple. The set of precise tuples represented by a normal tuple is seen as the most specific correct set-valued specification of the actual state ω_0 of the world. Thus we can define the possibility distribution that is induced by a given dataset D.

Definition 7: Let $D = (R, w_R)$ be a non-empty database ($R \neq \emptyset$) over an attribute set X. The (not necessarily normalized) possibility distribution π_D over X that is induced by D is defined as

$$\pi_D : T_{\text{prec}}(X) \to [0,1], \quad \pi_D(t) \mapsto \frac{\sum_{r \in \{s \in R \mid \forall A \in X : t(A) \subseteq s(A)\}} w_R(r)}{\sum_{r \in R} w_R(r)}. \quad \square$$

That is, the degree of possibility of a "point" t is the sum of the weights of those tuples (and thus of those sample cases) that cover it.

In learning possibilistic networks from data, it is tried to decompose such a database-induced possibility distribution into distributions on lower-dimensional subspaces. This is done in order to reduce the amount of data necessary to draw inferences in the underlying domain, since it can be shown that in a decomposition (as defined below) inferences can be carried out using only the distributions of the decomposition.

The distributions on lower-dimensional subspaces, of which a decomposition of a multivariate possibility distribution consists, are maximum projections of this possibility distribution.

Definition 8: Let π_D be a possibility distribution induced by a non-empty database $D = (R, w_R)$ over X and let $Y \subseteq X$. The *maximum projection* of π_D to Y, written $\pi_D^{(Y)} = \text{proj}_Y^X(\pi_D)$ is defined as

$$\pi_D^{(Y)} : T_{\text{prec}}(Y) \to [0,1], \quad \pi_D^{(Y)}(r) \mapsto \max_{t \in E_X(r)} \pi_D(t),$$

where $E_X(r)$, the extension of $r \in T_{\text{prec}}(Y)$ to X, is defined as $E_X(r) = \{s \in T_{\text{prec}}(X) \mid \text{proj}_Y^X(s) = r\}$. $\square$

A given possibility distribution π_D can be decomposed into distributions on lower dimensional subspaces, if it is possible to *reconstruct* π_D from the distributions on the subspaces.

Definition 9: Let π_D be a possibility distribution induced by a database $D = (R, w_R)$ over an attribute set X and let $\mathcal{Y} = \{Y_1, \ldots, Y_k\} \subseteq 2^X$ be a set of attribute sets, for which $\forall Y_1, Y_2 \in \mathcal{Y} : Y_1 \subseteq Y_2 \to Y_1 = Y_2$ (i.e. no attribute set Y_i is contained in any other).

π_D is called *decomposable* into a set of distributions on $Y \in \mathcal{Y}$, iff $\forall t \in T_{\text{prec}}(X)$:

$$\pi_D(t) = \min_{Y \in \mathcal{Y}} \pi_D^{(Y)}(t) = \min_{Y \in \mathcal{Y}} \text{proj}_Y^X(\pi_D)(t).$$

In this case the set $\{\pi_D^{(Y_1)}, \ldots, \pi_D^{(Y_k)}\}$ of maximum projections of π_D to the elements of $\mathcal{Y}$ is called a *decomposition* of π_D. $\square$

Often additional conditions are required to hold for the attribute sets in $\mathcal{Y}$, e.g. that the sets Y_i, if interpreted as the hyperedges of a hypergraph, form a hypertree, i.e. a hypergraph without cycles. Although this is a necessary condition for probabilistic decomposition, since cycles cannot be tolerated in undirected probabilistic networks (Markov networks), it can be neglected for possibilistic decomposition (due to the idempotence of the propagation operation).

In addition, since an exact decomposition is not what can be hoped for in practice, often approximations, resulting in a certain loss of information, are accepted. We neglect this possibility here.

We can now state clearly the problem this paper tries to solve. A learning algorithm for a possibilistic network usually consists, just like a learning algorithm for probabilistic networks, of two parts: an evaluation measure and a search method. The evaluation measure estimates the quality of a given decomposition candidate and the search method determines which decomposition candidates are inspected. But, of course, in order to evaluate a given decomposition candidate, one has to compute the maximum projections to the chosen set of subspaces. Therefore an important task for any learning algorithm for possibilistic networks is to efficiently compute the maximum projection of a database-induced possibility distribution to a given subset of the underlying set of attributes.

At first sight this seems to be a trivial task, but a closer look reveals that an *efficient* algorithm is not that easy to find. Especially, it is *not* possible, to compute the maximum projection simply as $\pi_D^{(Y)} = \max_{r \in E_X(t) \cap R} w_R(r)$, which could be computed in a single traversal of R. A simple example demonstrates this: Let $D = (R, w_R)$ be a dataset over $X = \{A, B\}$ with $R = \{(A \to \{a_1, a_2\}, B \to \{b_1, b_2\}), (A \to \{a_1\}, B \to \{b_2\})\}$ and $\forall t \in R : w_R(t) = 1$. From the above formula for $t = (A \to \{a_1\}, B \to \{b_2\})$ the degree of possibility $\pi_D(t) = 0.5$ would result, although the correct value is 1. The problem is that the tuples in R "intersect" on t and thus must be counted both.

3. Computation via the Support

Due to the problem mentioned above it seems to be necessary to construct the complete possibility distribution (or at least its "support", i.e. the set of precise tuples on which it is greater than zero) in order to compute the maximum projection of a database-induced possibility distribution. Of course, this is very inefficient, if possible at all, since the support of a database-

induced possibility distribution often contains a huge number of tuples (see section 5). Fortunately, there is a better method, based on the intersection property observed above and working with the "closure" of a relation, which we describe in the next section. We study the method based on the "support" of a relation first, since our proof, that the computation from the "closure" of a relation leads to the correct results, compares the results of the "closure" method to the results of the "support" method.

We start by defining the notion of "at least as specific as" for tuples. This notion is helpful in defining the support of a relation and also proves to be very useful when dealing with closures. We then define the support of a relation and of a dataset and demonstrate how a maximum projection of a database-induced possibility distribution can be computed from the support of a given (imprecise) database.

Definition 10: A tuple t_1 over an attribute set X is called *at least as specific as* a tuple t_2 over X, written $t_1 \sqsubseteq t_2$, iff $\forall A \in X : t_1(A) \subseteq t_2(A)$. □

Note that $\sqsubseteq$ is not a total ordering, since there are tuples that are incomparable. For example, $t_1 = (A \to \{a_1\}, B \to \{b_1, b_2\})$ and $t_2 = (A \to \{a_1, a_2\}, B \to \{b_1, b_3\})$ are not comparable, since neither $t_1 \sqsubseteq t_2$ nor $t_2 \sqsubseteq t_1$ holds.

Note also that $\sqsubseteq$ is obviously transitive, i.e. if t_1, t_2, t_3 are three tuples over an attribute set X with $t_1 \sqsubseteq t_2$ and $t_2 \sqsubseteq t_3$, then also $t_1 \sqsubseteq t_3$.

Finally, note that $\sqsubseteq$ is maintained by projection. That is, if t_1 and t_2 are two tuples over an attribute set X with $t_1 \sqsubseteq t_2$ and if $Y \subseteq X$, then $\text{proj}_Y^X(t_1) \sqsubseteq \text{proj}_Y^X(t_2)$. (We need this in the proof of theorem 1.)

Definition 11: Let R be a relation over an attribute set X. The support of R, written $\text{supp}(R)$, is the set of all precise tuples that are at least as specific as a tuple in R, i.e.

$$\text{supp}(R) = \{t \in T_{\text{prec}}(X) \mid \exists r \in R : t \sqsubseteq r\}. \quad \square$$

Obviously, $\text{supp}(R)$ is also a relation over X. Using the above definition we define the support of a dataset $D = (R, w_R)$ as $\text{supp}(D) = (\text{supp}(R), w_{\text{supp}(R)})$. In doing so, we fix $w_{\text{supp}(R)}$ in such a way, that any maximum projection of the possibility distribution π_D can be computed by taking maxima over $w_{\text{supp}(R)}$. This can be achieved by defining $w_{\text{supp}(R)}$ as

$$\begin{aligned} w_{\text{supp}(R)} : \text{supp}(R) &\to \mathbb{N}, \\ w_{\text{supp}(R)}(s) &\mapsto \textstyle\sum_{t \in R, s \sqsubseteq t} w_R(t) \end{aligned}$$

Comparing this definition to definition 7, we see that

$$\pi_D(t) = \begin{cases} \frac{1}{K} w_{\text{supp}(R)}(t), & \text{if } t \in \text{supp}(R), \\ 0, & \text{otherwise,} \end{cases}$$

where $K = \sum_{r \in R} w_R(r)$.

It follows, that any maximum projection of π_D can be computed easily from $w_{\text{supp}(R)}$ as follows (although the two are identical, we write $\pi^{(Y)}_{\text{supp}(D)}$ instead of $\pi^{(Y)}_D$ to indicate that it is computed via the support of D):

$$\pi^{(Y)}_{\text{supp}(D)} : T_{\text{prec}}(Y) \to [0, 1],$$
$$\pi^{(Y)}_{\text{supp}(D)}(r) \mapsto \begin{cases} \frac{1}{K} \max\limits_{t \in S_X(r)} w_{\text{supp}(R)}(t), & \text{if } S_X(r) \neq \emptyset, \\ 0, & \text{otherwise,} \end{cases}$$

where $S_X(r) = \{t \in \text{supp}(R) \mid r \sqsubseteq \text{proj}_Y^X(t)\}$ and $K = \sum_{s \in R} w_R(s)$. (Restricting $E_X(r)$ of definition 8 to $S_X(r) \subseteq E_X(r)$ is justified, since $\forall s \in E_X(r) \backslash S_X(r) : \pi_D(s) = 0$.)

4. Computation via the Closure

As already mentioned above, the computation of maximum projections via the support of a database is, in general, very inefficient, because of the usually vast number of tuples in $\text{supp}(R)$. In this section we introduce a computation of a maximum projection via a set of tuples that is usually much smaller then $\text{supp}(R)$, namely $\text{clos}(R)$. (That, in practice, it is indeed much smaller is demonstrated in section 5.)

We start by defining the intersection of two tuples and with it the closure of a relation under tuple intersection. By extending the notion of closure to a database and defining $w_{\text{clos}(R)}$ appropriately, any maximum projection of a database-induced possibility distribution can be computed by taking maxima over $w_{\text{clos}(R)}$. We prove that the degrees of possibility derived in this way are the same as those that result from a computation via the support of the given database.

Definition 12: Let X be a set of attributes. A tuple s over X is called the *intersection* of two tuples t_1 and t_2 over X, written $s = t_1 \sqcap t_2$, iff $\forall A \in X : s(A) = t_1(A) \cap t_2(A)$. □

Note that the intersection of two given tuples need not exist. For example, $t_1 = (A \to \{a_1\}, B \to \{b_1, b_2\})$ and $t_2 = (A \to \{a_2\}, B \to \{b_1, b_3\})$ do not have an intersection, since $t_1(A) \cap t_2(A) = \emptyset$, but a tuple may not map an attribute to the empty set (see definition 1).

Note also that the intersection s of two tuples t_1 and t_2 is at least as specific as both of them, i.e. $s \sqsubseteq t_1$ and $s \sqsubseteq t_2$. In addition, s is the least specific of all tuples s' for which $s' \sqsubseteq t_1$ and $s' \sqsubseteq t_2$, i.e. $\forall s' \in T(X) : (s' \sqsubseteq t_1 \wedge s' \sqsubseteq t_2) \to s' \sqsubseteq s \equiv t_1 \sqcap t_2$. This is important, since it also says that any tuple that is at least as specific as both of two given tuples is at least as specific as their intersection. (We need this in the proof of theorem 1.)

Finally, note that intersection is idempotent, i.e. $t \sqcap t = t$. (We need this below, where we collect some properties of closures.)

Definition 13: A relation R over an attribute set X is called *closed under tuple intersection*, iff

$$\forall t_1, t_2 \in R: \quad (\exists s \in T(X) : s = t_1 \sqcap t_2) \rightarrow s \in R,$$

i.e. iff R contains for any two tuples also their intersection (provided it exists). □

Definition 14: Let R be a relation over an attribute set X. The closure of R, written $\mathrm{clos}(R)$, is the set

$$\mathrm{clos}(R) = \{t \in T(X) \mid \exists S \subseteq R : t \equiv \bigsqcap_{s \in S} s\},$$

i.e. the relation R together with all possible intersections of tuples from R. □

Note that $\mathrm{clos}(R)$ is, obviously, also a relation and that it is closed under tuple intersection: If $t_1, t_2 \in \mathrm{clos}(R)$, then $\exists S_1 \in R : t_1 = \bigsqcap_{s \in S_1} s$ and $\exists S_2 \in R : t_2 = \bigsqcap_{s \in S_2} s$. If $\exists r \in T(X) : r = t_1 \sqcap t_2$, then $r = t_1 \sqcap t_2 = \bigsqcap_{s \in S_1} s \sqcap \bigsqcap_{s \in S_2} s = \bigsqcap_{s \in S_1 \cup S_2} s \in \mathrm{clos}(R)$. (The last equality in this sequence holds, since $\sqcap$ is idempotent, see above).

Note also that a direct implementation of the above definition is not the best way to compute $\mathrm{clos}(R)$. A better, because much more efficient way, is to start with a relation $R' = R$, to compute only pairwise intersections of tuples taken from R', and to add the results to R', until no new tuples can be added.

Just as for the support, we extend the notion of closure to databases and define the closure of a database $D = (R, w_R)$ as $\mathrm{clos}(D) = (\mathrm{clos}(R), w_{\mathrm{clos}(R)})$. In doing so, we fix $w_{\mathrm{clos}(R)}$ in such a way, that any maximum projection of the possibility distribution π_D can be computed by taking maxima over $w_{\mathrm{clos}(R)}$. This can be achieved by defining $w_{\mathrm{clos}(R)}$ as

$$\begin{array}{rcl} w_{\mathrm{clos}(R)} : \mathrm{clos}(R) & \rightarrow & \mathbb{N}, \\ w_{\mathrm{clos}(R)}(c) & \mapsto & \sum_{t \in R, c \sqsubseteq t} w_R(t). \end{array}$$

We assert that any maximum projection of π_D can be computed from $w_{\mathrm{clos}(R)}$ as follows (we write $\pi^{(Y)}_{\mathrm{clos}(D)}$ to indicate that it is computed via the closure of D):

$$\pi^{(Y)}_{\mathrm{clos}(D)} : T_{\mathrm{prec}}(Y) \rightarrow [0,1],$$
$$\pi^{(Y)}_{\mathrm{clos}(D)}(r) \mapsto \begin{cases} \frac{1}{K} \max\limits_{t \in C_X(r)} w_{\mathrm{clos}(R)}(t), & \text{if } C_X(r) \neq \emptyset, \\ 0, & \text{otherwise,} \end{cases}$$

where $C_X(r) = \{t \in \mathrm{clos}(R) \mid r \sqsubseteq \mathrm{proj}^X_Y(t)\}$ and $K = \sum_{s \in R} w_R(s)$. Since, as already mentioned, $\mathrm{clos}(R)$ usually contains much fewer tuples than $\mathrm{supp}(R)$, a computation based on the above formula is much more efficient. We establish our assertion, that any maximum projection can be computed in this way, by the following theorem.

Theorem 1: Let $D = (R, w_R)$ be a database over X and $Y \subseteq X$. Let $\mathrm{supp}(D) = (\mathrm{supp}(R), w_{\mathrm{supp}(R)})$ and $\mathrm{clos}(D) = (\mathrm{clos}(R), w_{\mathrm{clos}(R)})$ as well as $\pi^{(Y)}_{\mathrm{supp}(D)}$ and $\pi^{(Y)}_{\mathrm{clos}(D)}$ be defined as above. Then

$$\forall r \in T_{\mathrm{prec}}(Y): \quad \pi^{(Y)}_{\mathrm{clos}(D)}(r) = \pi^{(Y)}_{\mathrm{supp}(D)}(r),$$

i.e. computing the maximum projection of the possibility distribution π_D induced by D via the closure of D is equivalent to computing it via the support of D.

Proof: Let $r \in T_{\mathrm{prec}}(Y)$ be arbitrary. Let $K = \sum_{t \in R} w_R(t)$, $S_X = \{s \in \mathrm{supp}(R) \mid r \sqsubseteq \mathrm{proj}^X_Y(s)\}$, and $C_X = \{c \in \mathrm{clos}(R) \mid r \sqsubseteq \mathrm{proj}^X_Y(c)\}$. We prove the assertion of the theorem in two steps.

1) $\pi^{(Y)}_{\mathrm{clos}(D)}(r) \geq \pi^{(Y)}_{\mathrm{supp}(D)}(r)$:

a) $S_X = \emptyset$: $\pi^{(Y)}_{\mathrm{supp}(D)}(r) = 0 \leq \pi^{(Y)}_{\mathrm{clos}(D)}(r) \in [0,1]$.

b) $S_X \neq \emptyset$:

$$\begin{array}{rcl} \pi^{(Y)}_{\mathrm{supp}(D)}(r) & = & \frac{1}{K} \max_{s \in S_X} w_{\mathrm{supp}(R)}(s) \\ & = & \frac{1}{K} \max_{s \in S_X} \sum_{t \in R, s \sqsubseteq t} w_R(t). \end{array}$$

Let $\hat{s} \in S_X$ be (one of) the tuple(s) $s \in S_X$ for which $w_{\mathrm{supp}(R)}(s)$ is maximal. Let $V = \{t \in R \mid \hat{s} \sqsubseteq t\}$. Then it is $\pi^{(Y)}_{\mathrm{supp}(D)}(r) = \frac{1}{K} w_{\mathrm{supp}(R)}(\hat{s}) = \frac{1}{K} \sum_{t \in V} w_R(t)$. Since $V \subseteq R$, it is $t^* = \bigsqcap_{t \in V} t \in \mathrm{clos}(R)$. Since $\hat{s} \in S_X$, it is $r \sqsubseteq \mathrm{proj}^X_Y(\hat{s})$ and since $\forall t \in V : \hat{s} \sqsubseteq t$, it is $\hat{s} \sqsubseteq t^*$, hence $r \sqsubseteq \mathrm{proj}^X_Y(t^*)$. It follows that $t^* \in C_X$. Let $W = \{t \in R \mid t^* \sqsubseteq t\}$. Since $t^* = \bigsqcap_{t \in V} t$, it is $\forall t \in V : t^* \sqsubseteq t$ and hence $V \subseteq W$. Putting everything together we arrive at

$$\begin{array}{rcl} \pi^{(Y)}_{\mathrm{clos}(D)}(r) & = & \frac{1}{K} \max_{c \in C_X} w_{\mathrm{clos}(R)}(c) \\ & \geq & \frac{1}{K} w_{\mathrm{clos}(R)}(t^*) = \frac{1}{K} \sum_{t \in W} w_R(t) \\ & \geq & \frac{1}{K} \sum_{t \in V} w_R(t) = \pi^{(Y)}_{\mathrm{supp}(D)}(r) \end{array}$$

2) $\pi^{(Y)}_{\mathrm{clos}(D)}(r) \leq \pi^{(Y)}_{\mathrm{supp}(D)}(r)$:

a) $C_X = \emptyset$: $\pi^{(Y)}_{\mathrm{clos}(D)}(r) = 0 \leq \pi^{(Y)}_{\mathrm{supp}(D)}(r) \in [0,1]$.

b) $C_X \neq \emptyset$:

$$\begin{array}{rcl} \pi^{(Y)}_{\mathrm{clos}(D)}(r) & = & \frac{1}{K} \max_{c \in C_X} w_{\mathrm{clos}(R)}(s) \\ & = & \frac{1}{K} \max_{c \in C_X} \sum_{t \in R, c \sqsubseteq t} w_R(t). \end{array}$$

Let $\hat{c} \in C_X$ be (one of) the tuple(s) $c \in C_X$ for which $w_{\mathrm{clos}(R)}(c)$ is maximal. Let $W = \{t \in R \mid \hat{c} \sqsubseteq t\}$. Then it is $\pi^{(Y)}_{\mathrm{clos}(D)}(r) = \frac{1}{K} w_{\mathrm{clos}(R)}(\hat{c}) = \frac{1}{K} \sum_{t \in W} w_R(t)$. Let $U = \{t \in T_{\mathrm{prec}}(X) \mid t \sqsubseteq \hat{c}\}$. Since $r \in T_{\mathrm{prec}}(Y)$ and $r \sqsubseteq \mathrm{proj}^X_Y(\hat{c})$ (because of $\hat{c} \in C_X$), there must be a tuple $s^* \in U$, for which

$r \sqsubseteq \mathrm{proj}_Y^X(s^*)$. Since $s^* \in U$, $s^* \sqsubseteq \hat{c} \in \mathrm{clos}(R)$ and since $\forall c \in \mathrm{clos}(R) : \exists t \in R : c \sqsubseteq t$, it follows that $\exists t \in R : s^* \sqsubseteq t$ and hence $s^* \in \mathrm{supp}(R)$. Let $V = \{t \in R \mid s^* \sqsubseteq t\}$. Since $s^* \sqsubseteq \hat{c}$, it is $\forall t \in W : s^* \sqsubseteq t$ and hence $W \subseteq V$. Putting everything together we arrive at

$$\begin{aligned} \pi^{(Y)}_{\mathrm{supp}(D)}(r) &= \tfrac{1}{K} \max_{s \in S_X} w_{\mathrm{supp}(R)}(s) \\ &\geq \tfrac{1}{K} w_{\mathrm{supp}(R)}(s^*) = \tfrac{1}{K} \textstyle\sum_{t \in V} w_R(t) \\ &\geq \tfrac{1}{K} \textstyle\sum_{t \in W} w_R(t) = \pi^{(Y)}_{\mathrm{clos}(D)}(r) \end{aligned}$$

From 1) and 2) it follows that, since r is arbitrary, $\forall r \in T_{\mathrm{prec}}(Y) : \pi^{(Y)}_{\mathrm{clos}(D)}(r) = \pi^{(Y)}_{\mathrm{supp}(D)}(r)$. □

5. Experimental Results

We tested our method on three datasets, namely the Danish Jersey cattle blood type determination dataset (djc, 500 cases), the soybean diseases dataset (soybean, 683 cases), and the congress voting dataset (vote, 435 cases). (The latter two datasets are well known from the UCI Machine Learning Repository [15].) Each of these datasets contains a lot of missing values, which we treated as an imprecise attribute value. That is, for a missing value of an attribute A we assumed $\mathrm{dom}(A)$ as the set of values that the corresponding tuple maps A to. Unfortunately we could not get hold of any real world dataset containing "true" imprecise attribute values, i.e. datasets with cases in which for an attribute A a set $S \subset \mathrm{dom}(A)$ with $|S| > 1$ and $S \neq \mathrm{dom}(A)$ was possible. If anyone can direct us to such a dataset, we would be very grateful.

For each of the mentioned datasets we compared the reduction to a relation (keeping the number of occurrences in the tuple weight), the expansion to the support of this relation, and the closure of the relation. The results are as follows:

dataset	cases	tuples in R	tuples in $\mathrm{supp}(R)$	tuples in $\mathrm{clos}(R)$
djc	500	283	712818	291
soybean	683	631	n.a.	631
vote	435	342	98753	400

The entry "n.a." (not available) means that the resulting relation is too large to be computed. Hence we could not determine its size. It is obvious that using the closure instead of the support of a relation to compute the maximum projections leads to an enormous reduction in complexity, or, in some cases, makes it possible to compute a decomposition in the first place.

Applications of the considered method to actually learn possibilistic networks from data can be found in [2], [3], although the use of the presented method is not explicitly mentioned in these papers.

Acknowledgments

We are grateful to S.L. Lauritzen and L.K. Rasmussen for making the Danish Jersey cattle blood type determination example available for us.

References

[1] S.K. Andersen, K.G. Olesen, F.V. Jensen, and F. Jensen. HUGIN — A shell for building Bayesian belief universes for expert systems. *Proc. 11th Int. J. Conf. on Artificial Intelligence*, 1080–1085, 1989

[2] C. Borgelt and R. Kruse. Some Experimental Results on Learning Probabilistic and Possibilistic Networks with Different Evaluation Measures. *Proc. 1st Int. J. Conf. on Qualitative and Quantitative Practical Reasoning, ECSQARU-FAPR'97*, 71–85, Bad Honnef, Germany, 1997

[3] C. Borgelt and J. Gebhardt. Learning Possibilistic Networks with a Global Evaluation Method. *Proc. EUFIT'97*, Vol. 2, 1034–1038, Aachen, Germany, 1997

[4] G.F. Cooper and E. Herskovits. A Bayesian Method for the Induction of Probabilistic Networks from Data. *Machine Learning* 9:309–347, Kluwer 1992

[5] J. Gebhardt and R. Kruse. A Possibilistic Interpretation of Fuzzy Sets in the Context Model. *Proc. IEEE Int. Conf. on Fuzzy Systems*, 1089-1096, San Diego 1992.

[6] J. Gebhardt and R. Kruse. The context model — an integrating view of vagueness and uncertainty. *Int. Journal of Approximate Reasoning* **9** 283–314

[7] J. Gebhardt and R. Kruse. Learning Possibilistic Networks from Data. *Proc. 5th Int. Workshop on Artificial Intelligence and Statistics*, 233–244, Fort Lauderdale, 1995

[8] J. Gebhardt and R. Kruse. POSSINFER — A Software Tool for Possibilistic Inference. In: D. Dubois, H. Prade, and R. Yager, eds. *Fuzzy Set Methods in Information Engineering: A Guided Tour of Applications*, 407–418, Wiley, New York, 1996

[9] J. Gebhardt and R. Kruse. Tightest Hypertree Decompositions of Multivariate Possibility Distributions. *Proc. Int. Conf. on Information Processing and Management of Uncertainty in Knowledge-based Systems*, 1996

[10] D. Heckerman. *Probabilistic Similarity Networks*. MIT Press 1991

[11] D. Heckerman, D. Geiger, and D.M. Chickering. Learning Bayesian Networks: The Combination of Knowledge and Statistical Data. *Machine Learning* 20:197–243, Kluwer 1995

[12] R. Kruse, E. Schwecke, and J. Heinsohn. *Uncertainty and Vagueness in Knowledge-based Systems: Numerical Methods*. Series: Artificial Intelligence, Springer, Berlin 1991

[13] R. Kruse, J. Gebhardt, and F. Klawonn. *Foundations of Fuzzy Systems*, John Wiley & Sons, Chichester, England 1994

[14] S.L. Lauritzen and D.J. Spiegelhalter. Local Computations with Probabilities on Graphical Structures and Their Application to Expert Systems. *Journal of the Royal Statistical Society, Series B*, 2(50):157–224, 1988

[15] P.M. Murphy and D. Aha, UCI Repository of Machine Learning Databases, FTP from ics.uci.edu in the directory pub/machine-learning-databases, 1994

[16] H.T. Nguyen. Using Random Sets. *Information Science* 34:265–274, 1984

[17] J. Pearl. *Probabilistic Reasoning in Intelligent Systems: Networks of Plausible Inference (2nd edition)*. Morgan Kaufman, New York 1992

[18] A. Saffiotti and E. Umkehrer. PULCINELLA: A General Tool for Propagating Uncertainty in Valuation Networks. *Proc. 7th Conf. on Uncertainty in AI*, 323–331, San Mateo 1991

[19] G. Shafer and P.P. Shenoy. Local Computations in Hypertrees. Working Paper 201, School of Business, University of Kansas, Lawrence 1988