An Algebra for Structured Queries in Bayesian Networks

Jean-Noël Vittaut, Benjamin Piwowarski, and Patrick Gallinari

LIP6 Laboratoire d'Informatique de Paris 6,
8 rue du Capitaine Scott, 75015, Paris, France
{vittaut, bpiwowar, gallinar}@poleia.lip6.fr

Abstract. We present a system based on a Bayesian Network formalism for structured documents retrieval. The parameters of this model are learned from the document collection (documents, queries and assessments). The focus of the paper is on an algebra which has been designed for the interpretation of structured information queries and can be used within our Bayesian Network framework. With this algebra, the representation of the information demand is independent from the structured query language. It allows us to answer both vague and strict structured queries.

1 Introduction

Bayesian networks have been used by different authors for flat text information retrieval [2][9]. They have been shown to be theoretically well founded and different classical IR systems may be considered as particular cases of these BN-IR models. Recently, we proposed a BN model for structured IR retrieval [10][11][12]. This model allows taking into account local relations between the different elements of an XML document. It makes use of flat text IR models for computing local scores for document elements. BN inference is then used to compute a final score for the document elements. Inference on the BN variables allows combining in some way the relevance information computed for different document elements.

This paper describes the algebra we have developed for the interpretation of structured queries. It provides a representation of the query which is independent of any particular query language. The general algebra has been described in details in [13]. We show here how it can be adapted to the NEXI language used for INEX.

The paper is organized as follows. First we describe in section 2.1 an adaptation of the Okapi model which has been used as the local scorer for our BN system in the 2004 INEX evaluation. We briefly describe in section 2.2 the BN model and the use of Okapi within this BN. Results for CO queries are then presented in section 2.3. Section 3.1 describes the algebra used for the interpretation of the NEXI query language and its use for the VCAS queries of INEX. We finally present in section 3.2 the results obtained with this model on VCAS queries.

N. Fuhr et al. (Eds.): INEX 2004, LNCS 3493, pp. 100 – 112, 2005.

2 Content Only Queries

2.1 Okapi Model

We used Okapi as a standalone model and also as a local baseline model for Bayesian Networks. It allows us to compute a local score for each doxel (a document element) of the database. Then, this score is used to order the results (if we use the Okapi model alone) or as a source of evidence for Bayesian Networks.

In the Bayesian Network, the local scores provided by baseline models have to be interpreted as a probability (of relevance). So, we adapted Okapi [15] in order to:

– reach reasonable performances on the INEX corpus (and on a structured collection in general);
– compute a score which could be interpreted as a probability with this model.

The local score of a doxel x for a given query q , computed by the Okapi model, is defined by:

$$\text{Okapi}(q,x)=\sum_{j=1}^{\text{length}(q)}\omega_{j,x}\frac{(k_1+1)tf_{x,j}}{K_x+tf_{x,j}}\times\frac{(k_3+1)qtf_j}{k_3+qtf_j} \tag{1}$$

where k_1 and k_3 are constants, length (q) is the number of terms in query q. This formula is similar to classical Okapi except for the index x appearing in ω, K and tf. Okapi makes use of different statistics relative to the document collection such as term occurrences or mean document length. Since for Structured Information Retrieval (SIR) elements to be retrieved are doxels and not plain documents, these statistics have to be adapted. Values $\omega_{j,x}$ and K_X are defined as follows:

- $\omega_{j,x}=\log\left(\frac{N-n_j+0.5}{n_j+0.5}\right)$. In Okapi N is the number of documents in the collection and the n_j number of documents containing term j. There are different options for adapting these collection statistics to SIR. We will present here tests where these two values were defined respectively with respect to the classical document set ("document frequency") as in Okapi.
- $K_x=k_1\left((1-b)+b\frac{dl}{avdl}\right)$ where b is a constant and in Okapi dl is the document length and $avdl$ is the average document length. Here dl was replaced by the doxel length and one weighting scheme was tested for $avdl$: the average length taken respectively over all the doxels with the same tag ("tag").

We chose this peculiar weighting scheme as it allowed us to reach good performances when used by our BN model. As we said, we needed scores which can be interpreted as probabilities. Okapi score does not range between 0 and 1. The normalization of Okapi is discussed in [14] in the context of filtering, where it is proposed to make a regression of the original Okapi score via a logistic function. We used this idea here with the following transformation:

$$P\left(M_{\text{Okapi}}(x)=\mathrm{R} \mid q\right)=\frac{1}{1+e^{\alpha\times\text{Okapi}(q,x)/\text{length}(q)\text{-}\beta}} \tag{2}$$

This formula gives the normalized score for the local baseline variants of Okapi model. The α and β parameters were estimated on the whole INEX 2002 database. This score is dependant on the query length. Since the parameters of the logistic function should be valid for queries of varying length, this score was divided by the query length. We then computed the mean okapi score μ and the standard deviation σ for all the CO queries of INEX 2003. We then set α and β such that the probability $P\left(M_{\text{Okapi}}(x)=\mathrm{R} \mid q\right)$ is 0.5 when the score is μ and 0.75 when the score is $\mu+\sigma$. These values were chosen empirically.

This is different from [14] where the parameters of the regression are estimated for each query. This would not be realistic here because of the increased complexity of SIR.

2.2 Bayesian Networks

Model

Let us consider a hierarchically structured collection like the INEX corpus. Documents are organised in a category hierarchy with corpus as the root node, journal collections as its immediate descendents, followed by journals, articles etc. We view retrieval for such a collection as a stochastic process in which a user goes deeper and deeper in the corpus structure: the user starts its search at the "root node" of all categories, and then selects one or several categories in which relevant documents should be. For each category, he or she selects subcategories and/or documents within these categories. This process is iterated until the user has found relevant and highly specific doxels.

The BN structure we used directly reflects this document hierarchy and retrieval will follow the above stochastic process. We consider that each structural part within the hierarchy has an associated random variable. The root of the BN is thus a "corpus" variable, its children the "journal collection" variables, etc. The whole collection is thus modelled as a large BN which reflects the doxel hierarchy in the collection.

Each random variable in the BN can take its values in a finite set. Existing BN models for flat [2] or structured [8] documents use binary values $(R,\neg R)$. This is too limitative for SIR since quantifying an element relevance is more complex than for whole documents and should somewhat be related to the two dimensional scale (specificity, exhaustivity) proposed for INEX. We used a state space of cardinality 3, $V=\{I,B,E\}$ with:

1. I for Irrelevant when the element is not relevant;
2. B for Big when the element is marginally or fairly specific;
3. E for Exact when the element has a high specificity.

In this model, relevance is a local property in the following sense: if we knew that an element is relevant, not relevant or too big, the relevance value of its parent would not bring any new information on the relevance of one of its descendants.

For any element X and for a given query q, the probability $P(X = E, X\text{'s parent} = B \mid q)$ will be used as the final Retrieval Status Value (RSV) of this element. Using the simpler RSV $P(X = \mathrm{E} \mid q)$ led to poor performances with the BN. Our choice was partly motivated by the work of Crestani et al. [3][4] and by preliminary experiments.

Besides these variables, there are two more types of random variables in the BN. The first one corresponds to the information need, it is denoted Q and its realization is a query denoted q. Q is a vector of word frequencies taking its values in a multidimensional real space. This random variable is always observed (known). Document textual information is not directly modelled in this BN for complexity reasons. Instead a series of baseline IR models will be used to compute local relevance scores for each doxel given a query. For each local baseline model, this score will only depend on the doxel content and on the query. It is then independent on the context of the doxel in the XML tree. The global score for each doxel will then combine these local scores and will also depend on the doxel context in the BN – the parent's relevance. These local baseline models have been adapted from classical (flat) retrieval IR models. In the experiments presented here, one variant of the Okapi model was used for baseline: the okapi with standard document frequency and a length normalisation over elements with the same tag. In the BN model a random variable is associated to each local scorer and doxel. Let $M(X)$ denote the random variable associated to the local baseline model and doxel X and m its realization. As in classical IR this variable will take two values: R (relevant) and $\neg R$ (not relevant), i.e. $m \in \{R, \neg R\}$. The local relevance score at X given query q for the baseline model will be $P(M(X) = R \mid q)$. Note that it is straightforward to add new baseline models; in the following, all the formulas were adapted to the case where we have only one baseline model.

Based on the local scores $M(X)$ and on the BN conditional probabilities, BN inference is then used to combine evidence and scores for the different doxels in the document model. In our tree like model, the probability that element X is in state I, B or E depends on its parent state and on the fact that the local baseline models have judged the element as relevant or not relevant. The probability for X to be in a given state $v \in \mathrm{V}$ is then:

$$P(X = v \mid q) = \sum_{v_Y \in \mathrm{V}, m \in \{R, \neg R\}} P(X = v_X \mid Y = v_Y, M(X) = m) P(M(X) = m \mid q) \tag{3}$$

In this expression, the summation is over all the possible values of v_Y and m ($\mathrm{v_Y}$ can take any value in $V = \{I, B, E\}$, and m can take values $R, \neg R$). The conditional probability is expressed as follows:

$$P(X = v_X \mid Y = v_Y, M = m) = F_X(\Theta, v_X, v_Y, m) = \frac{e^{\theta_{c_X, v_X, v_Y, m}}}{\sum_{v \in V = \{I, E, B\}} e^{\theta_{c_X, v\ , v_Y, m}}} \tag{4}$$

where the $\theta_{c_X, v_X, v_Y, m}$ are real values to be learned. There is one such parameter for each tag category c_X and value set v_X , v_Y , m . All the doxels sharing the same value set c_X, v_X, v_Y, m will share this θ parameter. The denominator ensures that conditional probabilities sum to 1.

Training Algorithm

In order to learn the parameters and to fit to a specific corpus, we used a training criterion based on the relative order of elements. We used all the assessed CO topics from the INEX 2003 dataset. The criterion to be minimised was:

$$Q(\Theta) = \sum_q w(q) \sum_{i,j} e^{(RSV(i,q) - RSV(j,q)) s_q(i,j)} \tag{5}$$

where the weighted q summation is over the set of all training queries and the i and j ones are over the set of all doxels in the training set. $RSV(i,q)$ is the score of the element X_i and s_q is defined as follows:

$$s_q(i,j) = \begin{cases} -1 & \text{if } X_i \text{ is "better" than } X_j \text{ for query } q \left(X_i >_q X_j\right) \\ 1 & \text{if } X_j \text{ is "better" than } X_i \text{ for query } q \left(X_i <_q X_j\right) \\ 0 & \text{otherwise} \left(X_i =_q X_j\right) \end{cases} \tag{6}$$

The order ("better than") between the elements depends on the assessments for a given query *q*. For instance, a highly specific and exhaustive element is "better than" a fairly exhaustive and highly specific one. We used the following partial order between assessments (from "best" to "worst"):

1. Highly specific and highly exhaustive
2. Highly specific and fairly exhaustive
3. Highly specific and marginally exhaustive
4. All other assessment including "not assessed"

The score of an element in the criterion formula is either $P(X = \mathrm{E} \mid q)$ for the model BN1 or $P(X = E, X\text{'s parent} = B \mid q)$ for the model BN2. The latter is more complex but more related to our scoring algorithm. The weight *w(q)* was chosen in order to normalize the contribution of different topics: even if the number of assessments were different, this normalization ensured that each topic had the same influence on the criterion. The criterion is minimal when all the elements are ordered according to our partial order.

In order to optimize the criterion, different gradient algorithms could be used. For the experiments we used a simple gradient descent algorithm where the learning rate (epsilon) was automatically set by a line search; for this latter, we use the Armijo algorithm. The number of steps was chosen so as to optimize the performance with respect to the generalized recall metric. For BN1, a maximum was reached after 195 iterations while for BN2 a maximum was reached after 700.

2.3 Experiments

Three official runs were submitted to INEX'04:

- **Okapi.** In this run, we used the Okapi weighting scheme; every volume (and not every doxel) in the INEX corpus was considered as a document while the average document length used in the Okapi formula was local: for every doxel, the average document length was the average length of the doxels with the same tag.
- **BN1.** In this run, we submitted the doxel retrieved with the BN which is described in 0. The former Okapi model was used for local model, $P(X = \mathrm{E} \mid q)$ was used as the score of an element for the learning process, and $P\left(X = E, X\text{'s parent} = B \mid q\right)$ was used as the score of an element. INEX tags were grouped in categories.
- **BN2.** In this run, we also submitted the doxel retrieved with the BN which was learnt with a different grouping of tag names. $P\left(X = E, X\text{'s parent} = B \mid q\right)$ was used as the score of an element both for learning and testing.

With respect to the experiments we have done the two previous years [10][12], this ranking criterion seems the most promising one – in INEX 2002 and 2003 we used a maximum likelihood algorithm (EM) [5] which was not well fitted to this task. However, the partial order should be refined so as to be more close to the "ideal" user related criterion.

Table 1. CO official runs

	Generalized recall	Average of all RP measures
Okapi	40.74	0.10
BN1	46.45	0.04
BN2	45.97	0.05

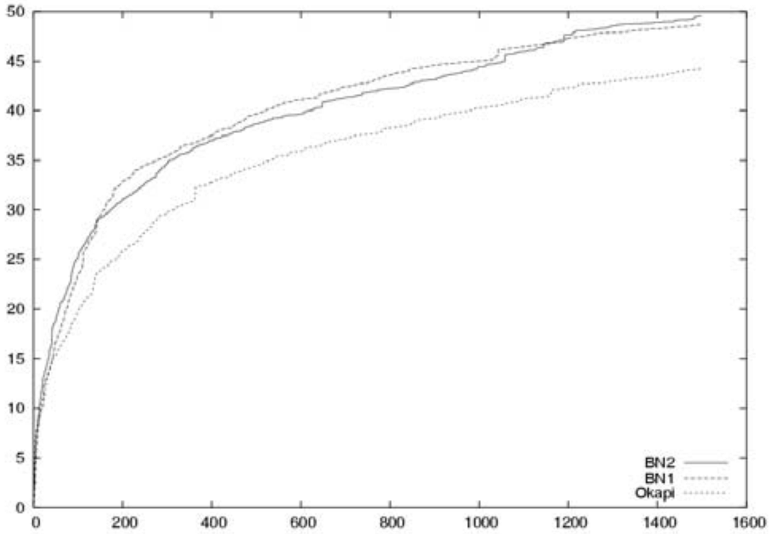

Fig. 1. CO official runs with generalized recall metric

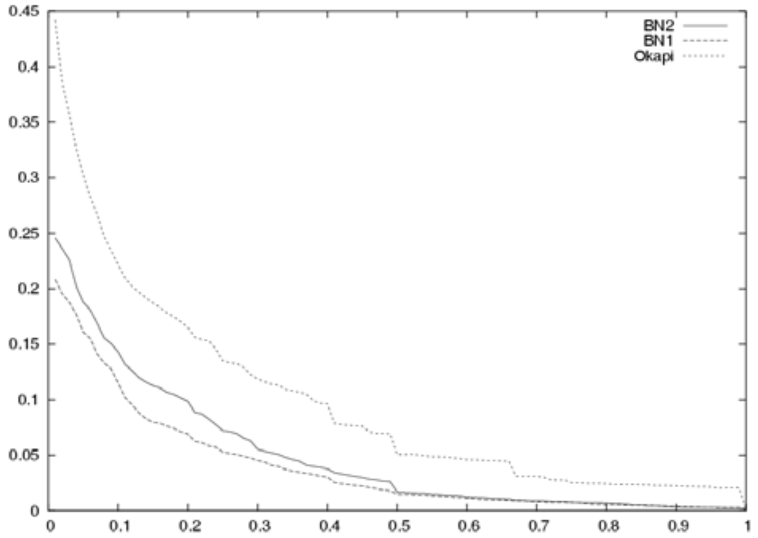

Fig. 2. CO official runs with the average of all RP measures

3 Content and Structure Queries

In this section, we present the algebra we have used to answer Vague Content and Structure Queries (VCAS) starting from the scores of BN Model or standalone Okapi model. We only give some elements to understand the way we use this algebra in the specific case of NEXI queries. A more detailed description of the algebra is given in [13]. At last, we give the results of the experiments on INEX 2004 for the Okapi model and Bayesian Networks.

3.1 Algebra

Introduction

In INEX, queries are expressed in a query language (NEXI) which is very similar to XPath in which a vague operator (about) is introduced in order to allow for queries in a similar fashion than in information retrieval. Such languages can be used to express query needs that mix possibly vague content and structure constraints. XPath is for XML documents what SQL is for databases: it is a language that describes which information should be retrieved from XML documents. In traditional databases, this request is usually mapped into an algebra which in turn is used as a query plan. This query plan is closely related to physical operations that will give the answers to the query. In databases, the result of a formula of the algebra is a set of tuples. In XML databases, the result is a set of elements.

Defining or choosing an algebra is very important to answer complex query needs. This is proved by the important number of works within the semi-structured database field, like for example [1][9]. Such approaches are also used in INEX [7]. Our algebra is closely related to the one defined by Fuhr and Grossjohan [6]. As in classical IR, SIR aim is to retrieve the set of document elements that fulfill a given query need. This query need is very often imprecise. The algebra we define here can be used to answers vague queries that have constraints on both content and structure and make use of the Bayesian Networks framework that we use for CO queries.

Algebra Description

Besides classical operators of the set theory like the intersection, the union and the complementary, our algebra uses structural operators like:

- $\overline{desc}/(x)$ (descendant or self)
- $\overline{anc}_{/}(x)$ (ancestor or self)
- other operators which are not mentioned here because they are useless with the specification of NEXI language.

We denote X the set of all doxels. We introduce three functions:

1. *R(q)* which returns the set of doxels which are answers to the query need *q*. A doxel is in the set with a given probability.
2. $[comp(t, comparison\_op)]$ which returns the set of doxels x where $comparison\_op(x,t)$ is true. We have used $=, \neq, \leq, \geq, <, >$ as comparison operators.
3. *label (x)* which returns the label of the doxel (the tag name). The function *label^{-1}(l)* returns the set of doxels which have a label *l* (This function is used for SCAS queries). In order to process VCAS queries, we can replace the latter function by a vague one called *invlabel (l)* which returns a set of labels with a given probability.

The algebra is defined on the set P(X) (the set of all the part of the set of doxels). We use the operator "$\circ$" to compose the different functions defined on P(X) which take values in P(X).

With all these operators and functions, we are able to answer structured queries.

Probabilistic Interpretation

In the previous section, *R(q)* returns the set of doxels that are answers to query *q*. In Information Retrieval (IR), the answers to a query are not well defined: the query is expressed in vague terms, and the real query need cannot be easily defined. We thus have to define $R(q)$ as a "*vague*" set in order to compute the answer to a query that contains predicates like *about*.

In our approach, as in the probabilistic interpretation of fuzzy sets [16], a set $A \subset \mathrm{X}$ is not anymore defined strictly. We denote such a set by A_v (*v* for *vague*). A_v is defined by a probability distribution on subsets of X. The case where probability $P(A_V = A) = 1$ means that the set A_v is strict and not vague (the concept of fuzzy set is thus more general than the concept of classical set). An element a belongs to A_v with a probability $P(a \in A_V)$ which is formally defined by:

$$P(a \in A_V) = \sum_{A \subset \mathrm{X}, a \in A} P(A_V = A) \tag{7}$$

We define recursively the fact that a doxel belongs to a vague set:

$$x \in \varphi(A_V) \equiv \bigvee_{x' \in X, x \in \varphi(\{x'\})} x' \in A_V \tag{8}$$

Lastly, intersection and union operators can also be transformed in logical formulas:

$$\begin{aligned} x \in A_V \cap B_V &\equiv (x \in A_V) \wedge (x \in B_V) \\ x \in A_V \cup B_V &\equiv (x \in A_V) \vee (x \in B_V) \end{aligned} \tag{9}$$

Algebraic Expression of a CAS Query

In order to convert a NEXI query into an algebraic expression, we briefly define the way we decompose the NEXI Queries used in INEX, which can be easily extended to XPath like queries.

A NEXI query is read from left to right. For instance, the NEXI query :

//article[about(., “bayesian networks”)]//section[about(., “learning structure”)]

could be used to express “in articles about Bayesian Networks, find sections that are about learning the structure of the network”.

The different components are separated by two slashes “$//$ ” which are not within brackets. The query $//L_0[F_0]//L_1[F_1]...//L_n[F_n]$ can be decomposed into $//L_i[F_i]$ elements, each such component being itself composed of three parts:

1. **The axis** ($//$)**.** This is an abbreviation of the $/descendant-or-self::$ denoted axis in XPath. It defines a set with respect to a given doxel *x*. For the first component of the XPath, this set is defined by the document *d*. For the first component of an XPath within a filter, the set of selected doxels is evaluated with respect to the document *d* or to the filtered doxel. For any another component, the selection is made with respect to the set of doxels selected by the previous component;
2. **The label** *(L_i)* It filters the set of doxels selected by the axis and keep only those which have the label L_i. When the label is *, the “axis” set is not filtered;
3. **The filter** *(F_i)* that expresses a boolean condition on doxels. It returns the subset of those doxels which fulfill the boolean conditions expressed in the filter. An XPath can be used in the filter: it is relative to the context path and take the form of $.//L_0//L_1...//L_n$. The filter is a condition which can be true or false for one doxel.

An algebraic expression is defined on $\mathrm{P}(\mathrm{X})$. Each component of the query (either axis or label L_i or filter F_i) can be processed separately:

- An axis is transformed into the structural operator $\Psi_A(//) = \overline{desc/}$ except for the first component of the XPath which is transformed into $\Psi_A^{(0)}(//) = \overline{desc/}(d)$.
- A label (or a set of labels) L_i is transformed into a function Ψ_L that selects a subset of doxels which have a label L_i in the set:

$$\begin{array}{lll} \Psi_L(L_i): & \mathrm{P}(\mathrm{X}) & \mapsto \mathrm{P}(\mathrm{X}) \\ & X & \mapsto X \cap label^{-1}(L_i) \end{array}$$

where we handle the special case of * by defining $label^{-1}(*) = \mathrm{X}$

- As for the filter F_i, the transformation is more complex and is denoted Ψ_F:

$$\begin{array}{lll} \Psi_F(F_i): & \mathrm{P}(\mathrm{X}) & \mapsto \mathrm{P}(\mathrm{X}) \\ & X & \mapsto X \cap \Psi'_F(F_i) \end{array}$$

where Ψ'_F is the function which transforms a filter into the set of doxels that fulfill the conditions expressed in the filter.

With these notations, the query $p = // L_0[F_0] // L_1[F_1] \ldots // L_n[F_n]$ is the result of the evaluation of the algebraic expression:

$$\begin{array}{rcl} \Psi(d,p) & = & \Psi_F(F_n) \circ \Psi_L(L_n) \circ \Psi_A(//) \\ & & \ldots \\ & & \circ\, \Psi_F(F_1) \circ \Psi_L(L_1) \circ \Psi_A(//) \\ & & \circ\, \Psi_F(F_0) \circ \Psi_L(L_0) \circ \Psi_A^{(0)}(//) \\ & = & \Psi'_F(F_n) \cap label^{-1}(L_n) \\ & & \cap\, \overline{desc/} \left(\begin{array}{l} \ldots \\ \cap\, \overline{desc/} \left(\begin{array}{l} \Psi'_F(F_1) \cap label^{-1}(L_1) \\ \cap\, \overline{desc/} \left(\begin{array}{l} \Psi'_F(F_0) \cap label^{-1}(L_0) \\ \cap\, \overline{desc/}(d) \end{array} \right) \end{array} \right) \end{array} \right) \end{array} \tag{10}$$

We do not detail here how Ψ'_F is evaluated.

3.2 Experiments

To compute the union or the intersection of two vague sets, we used the probabilistic and/or operators defined below:

- $P(a \in A \wedge b \in B) = P(a \in A) P(b \in B)$
- $P(a \in A \vee b \in B) = P(a \in A) + P(b \in B) - P(a \in A) P(b \in B)$

Table 2. VCAS official runs

	Generalized recall	Average of all RP measures
Okapi	33.14	0.05
BN1	27.97	0.05
BN2	31.67	0.04

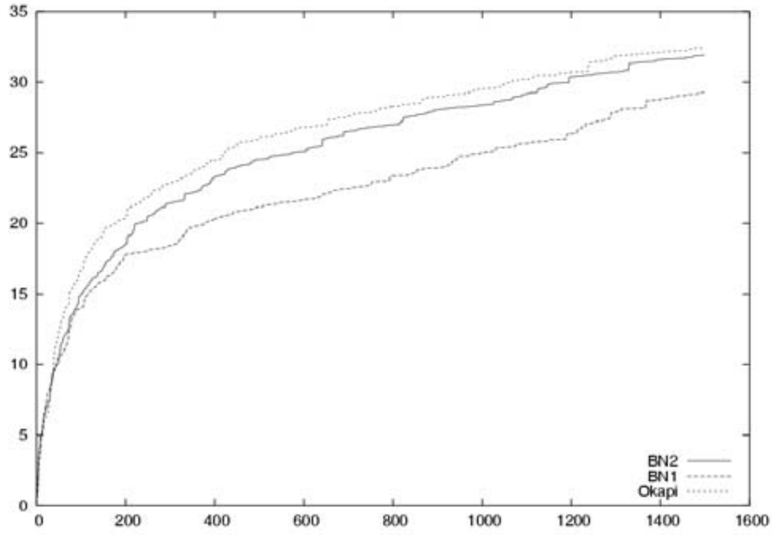

Fig. 3. VCAS official runs with generalized recall metric

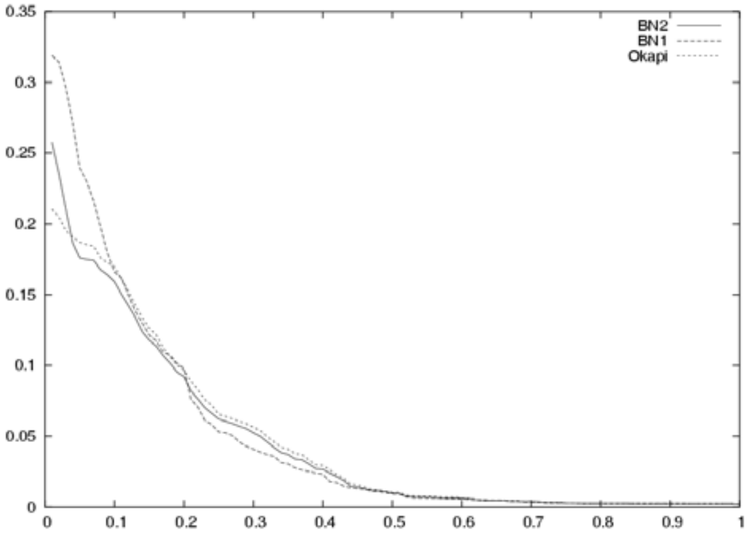

Fig. 4. VCAS official runs with the average of all RP measures

Complement remains $P(a \notin A) = 1 - P(a \in A)$. We have also tested min/max and Lukasievicz operators, but they were outperformed by the probabilistic operator.

In order to introduce vagueness into the query structure, we used the following labelling function:

$$invlabel(l) = \bigcup_{x \in \mathrm{X}, label(x)=l} \{pa(x)\} \cup \{x\} \cup \{y \in \mathrm{X},\, pa(y) = x\} \tag{11}$$

where we supposed that all the doxels from this set have the same probability of being labelled *l*. We have also tested the labelling function $label^{-1}(l)$, and other simple strategies to introduce vagueness into structure (not considering tag names), but they were outperformed by $invlabel(l)$.

Three official runs were submitted to INEX'04; the models we used to compute the probability of relevance are the same as in section 2.3.

The results are summarized in figures 3, 4 and in table 2. For generalized recall, Okapi model outperforms BN1 and BN2 models but not significantly. For CO, Okapi was below the BNs. For RP measures, results are similar for all models.

Our algebra can answer all INEX VCAS and also more complex structured queries. Nevertheless, the connection between CO queries and VCAS queries is not clear because the best model for CO queries is not the best one for VCAS queries. The Okapi model gives better results for structured queries than for content only queries. Moreover, the choice of union and intersection functions for aggregation has to be further investigated.

4 Conclusion

We introduced a BN model whose conditional probability functions are learnt from the data via a gradient descent algorithm. The BN framework has some advantages. Firstly, it can be used in distributed IR, as we only need the score of the parent element in order to compute the score of any its descendants. Secondly, it can use simultaneously different baseline models: we can therefore use specific models for non textual media (image, sound, etc.) as another source of evidence.

We have described the new algebra we have used in order to process content-and-structure queries. This algebra is a generic way to represent structured queries and can be easily used with the IR system based on Bayesian Networks we have developed.

Our system can answer CO and VCAS queries. The model has still to be improved, tuned and developed. In particular, we should improve the baseline models and fit them to the specificities of CO or VCAS queries. We showed that this algebra allows answering VCAS queries but we still have to investigate new ways of including vagueness in the structure of queries.

References

1. Abiteboul, S., Quass, D., McHugh, J., Widom, J., Wiener, J.L.: The lorel query language for semistructured data. International Journal on Digital Libraries, 1 (1997) 68–88
2. Callan, J., Croft, W.B., Harding, S.M.: The INQUERY Retrieval System. In A. Min Tjoa and Isidro Ramos, editors, Database and Expert Systems Applications, Proceedings of the International Conference, pages 78-83, Valencia, Spain, Springer-Verlag (1992)
3. Crestani, F., de Campos, L.M., Fernandez-Luna, J.M., Huete, J.F.: A multi-layered Bayesian network model for structured document retrieval, ECSQARU, LNAI 2711, Springer-Verlag (2003) 74-86
4. Crestani, F., de Campos, L.M., Fernandez-Luna, J.M., Huete, J.F.: Ranking Structured Documents Using Utility Theory in the Bayesian Network Retrieval Model, In SPIRE (String Processing and Information Retrieval), Brazil (2003) 168-182
5. Dempster, A.P., Laird, N.M., Rubin, D.B.: Maximum Likelihood from incomplete data via de EM algorithm. The Journal of Royal Statistical Society, 39 (1977) 1-37
6. Fuhr, N., Grossjohann, K.: XIRQL: A query language for information retrieval in XML documents. In W. B. Croft, D. J. Harper, D. H. Kraft, and J. Zobel, editors, The 24th International Conference on Research and Developmenent in Information Retrieval, New Orleans, Louisiana, USA, ACM (2001)

7. List, J., Mihajlovic, V., de Vries, A.P., Ramirez, G.: The TIJAX XML-IR system at INEX 2003. In N. Fuhr, M. Lalmas, and S. Malik, editors, INitiative for the Evaluation of XML Retrieval (INEX). Proceedings of the Second INEX Workshop, Dagstuhl, Germany (2003).
8. Myaeng, S.H., Jang, D.H., Kim, M.H. , Zhoo, Z.C.: A Flexible Model for Retrieval of SGML documents. In Proceedings of the 21st Annual International ACM SIGIR Conference on Research and Development in Information Retrieval, Melbourne, Australia, ACM Press, New York (1998) 138-140
9. Navarro, G., Baeza-Yates, R.: Proximal nodes: A model to query document databases by content and structure. ACM TOIS, 15 (1997) 401-435
10. Piwowarski, B., Faure, G.E., Gallinari, P.: Bayesian networks and INEX. In Proceedings of the First Annual Workshop of the Initiative for the Evaluation of XML retrieval (INEX), DELOS workshop, Dagstuhl, Germany ERCIM (2002)
11. Piwowarski, B., Gallinari, P.: A Bayesian Network for XML Information Retrieval: Searching and Learning with the INEX Collection, In Information Retrieval (2004)
12. Piwowarski, B., Vu, H.T., Gallinari, P.: Bayesian Networks and INEX'03. In INitiative for the Evaluation of XML Retrieval, Proceedings of the Second INEX Workshop (2003)
13. Piwowarski, B., Gallinari, P.: An algebra for probabilistic XML Retrieval. In The First Twente Data Management Workshop (2004)
14. Robertson, S.: Threshold setting and performance optimization in adaptive Filtering. Information Retrieval, 5 (2002) 239-256
15. Walker, S., Robertson, S.E.: Okapi/Keenbow at TREC-8. In E. M. Voorhees and D. K. Harman, editors, NIST Special Publication 500-246: The Eighth Text REtrieval Conference (TREC-8), Gaithersburg, Maryland, USA (1999)
16. Zadeh, L.A.: Fuzzy sets (1965)