Precision Recall with User Modeling (PRUM): Application to Structured Information Retrieval

B. PIWOWARSKI
Yahoo! Research Latin America
P. GALLINARI
Laboratoire d'Informatique de Paris 6
and
G. DUPRET
Yahoo! Research Latin America

Standard Information Retrieval (IR) metrics are not well suited for new paradigms like XML or Web IR in which retrievable information units are document elements and/or sets of related documents. Part of the problem stems from the classical hypotheses on the user models: They do not take into account the structural or logical context of document elements or the possibility of navigation between units. This article proposes an explicit and formal user model that encompasses a large variety of user behaviors. Based on this model, we extend the probabilistic precision-recall metric to deal with the new IR paradigms.

Categories and Subject Descriptors: H.3.3 [**Information Storage and Retrieval**]: Information Search and Retrieval; H.3.4 [**Information Storage and Retrieval**]: Systems and Software—*Performance evaluation*

General Terms: Measurement, Theory

Additional Key Words and Phrases: Evaluation, information retrieval, measure, precision-recall, Web, XML

ACM Reference Format:

Piwowarski, B., Gallinari, P., and Dupret, G. 2007. Precision recall with user modeling (PRUM): Application to structured information retrieval. ACM Trans. Inform. Syst. 25, 1, Article 1 (February 2007), 37 pages. DOI = 10.1145/1198296.1198297 http://doi.acm.org/10.1145/1198296.1198297.

Authors' address: B. Piwowarski, Yahoo! Research Latin America, Blanco Encalada 2120, Santiago, Chile; email: bpiwowar@dcc.uchile.cl; P. Gallinari, Laboratoire d'Informatique de Paris 6, University Paris 6, Paris, France; G. Dupret, Yahoo! Research Latin America, Blanco Encalada 2120, Santiago, Chile.

1. INTRODUCTION

Evaluation has always been a key problem in Information Retrieval (IR). The experimental practice started with the work on the Cranfield collection [Cleverdon 1967] and has been popularized since then by the TREC conferences and other IR challenges. State-of-the-art evaluation metrics are used to compare different systems and to justify theoretical and/or pragmatic developments of IR systems. As a part of the evaluation process for emerging IR fields like XML or Web IR, developing adequate metrics is an essential and an open question.

Among the different measures and criteria that have been proposed (see Van Rijsbergen [1979], Baeza-Yates and Ribeiro-Neto [1999], Meadow et al. [1999] for a description of most of the metrics proposed in IR), standard metrics are most often combinations of recall and precision. *Recall* is defined as the ratio of the number of relevant documents that are retrieved to the total number of relevant documents. *Precision* is the proportion of relevant documents among the retrieved ones. Both measures rely on assumptions concerning the nature of the collection and the user behavior. These basic assumptions are that (1) the units to be retrieved are whole documents, (2) the relevance judgements of different documents are independent, (3) the user reads only one document at a time before proceeding to the next one, and (4) the user is assumed to consult an ordered list ranked by decreasing relevance score (as estimated by the retrieval system) and to stop as soon as his information need is satisfied.

These assumptions need to be reexamined before they can be applied to new IR paradigms. XML, video, and Web documents, for example, are organized according to some logical structure where document elements may share different relations. This evolution of the document structure changes the way information is to be accessed. XML and video[1] IR aim at retrieving elements with the appropriate level of granularity: for example, an element in an XML document or a shot in a video. In Web retrieval, potential answers are Web pages linked together. The granularity of the retrieved information unit may be chosen by the retrieval system. In XML and video IR, the atomic unit to be retrieved is no longer a whole document, contradicting assumption (1), but rather document components which best satisfy the user need. Most of the time these components are not independent and share some relation. This does not comply with assumption (2).

Two other major issues concern *overlapping* and *near miss* elements. The former refers to retrieved components which may overlap in some way, for instance, a section and its paragraphs in XML retrieval. The latter is related to the structural dependence of elements: it seems natural to reward an element containing a relevant element (e.g., a section containing a relevant paragraph or a shot containing a relevant video fragment) or giving access to (e.g., hyperlinks) relevant elements. The context of an element thus plays a crucial role. Assumption (3) implies the absence of user navigation and thus needs to be modified for taking into account context and navigation.

[1]In the context of MPEG-7 retrieval.

Defining an adequate metric for these new IR paradigms is still an open problem. Most metrics proposed so far for structured IR are not satisfying. The first generalization of precision recall [Kazai 2003] did not correctly handle overlapping and near misses. The precision-recall metric proposed by Gövert et al. [2003] handled overlap but introduced instability in the recall base (i.e., the set of relevant elements to which the list returned by the search engine is compared). Eventually, an interesting extension of the cumulated gain metric [Kekäläinen and Järvelin 2002] was proposed by Kazai et al. [2004]. This measure is, however, specific to XML retrieval and has no explicit user model. A thorough discussion of existing metrics and a comparison with PRUM is provided in Section 4.

The contributions of this article are twofold. First, we propose a general navigating user model that can be parametrized and adapted to the new IR paradigms like XML, Web, or video IR. This will provide a formal framework for modeling the possibility of navigating among the different elements of these structured information sources. Second, from this user model, we define a generalization of the classical precision-recall metric and derive the formulas that will be used to compute it. The proposed metric thus integrates a well-defined general user model and takes into account structured IR characteristics. It solves the major issues of structured IR and offers, in particular, a solution for overlapping and near miss elements. This metric is an extension of the probabilistic version of the precision-recall metric described in Raghavan et al. [1989].

The article is organized as follows. The navigating user model is presented in Section 2. In Section 3, precision-recall is adapted to the new user model. This new metric is called PRUM for Precision Recall with User Modeling. In Section 4, we compare PRUM qualitatively with existing metrics. Most of the formal derivations are described in Appendix A. A proof that PRUM generalizes precision-recall is given in Appendix B.

2. NAVIGATING USER AND THE RELEVANCE MODEL

Deriving appropriate user models is an essential component of any IR metric. The classic user model used in most flat document retrieval systems is not adapted to structured IR tasks. A well-defined user model should take into account the nature of document elements and the relations between the different elements. Since the usual flat model cannot be naturally extended to take into account structural information, one needs a new model formulation.

In order to define our user model, we first need to refine and extend the concept of relevance. While in flat document IR the different units can be considered as independent, this is not the case in, for example, XML IR: if a paragraph is exactly what the user would want to see, then its enclosing section also bears some relevance. One needs to introduce a new concept to make the distinction between the element(s) the user wants to retrieve and the elements which may contain or be linked to this desired element and which are in some sense also relevant. Considering the section/paragraph example, we will say that both elements are relevant (as they contain desired material) but that only the paragraph is *ideal*. Note that an ideal element is always relevant but the reverse is true only in classical IR. We will not discuss here how such a unit can be found (see Piwowarski and Gallinari [2003] and Kazai et al. [2004] for more details in the case in XML IR) from human assessments; we only suppose it can be defined.

Current metrics suppose that a user only sees the elements of the list. PRUM, on the other hand, considers these elements as entry points to the collection. The context of an entry element **a** is defined as the set of elements that can be reached through navigation from **a**, including **a** itself. The set of elements belonging to the context depends on the collection and on the relations between elements. PRUM assumes that when a user reaches an entry element **a**, he explores a part of that element context. This is what we call the user navigational behavior. When he has explored the element context enough, he proceeds to the next entry of the list and repeats the process until his information need is satisfied.

For any item in the list, user navigation is restricted to the context of this entry element which, in turn, is determined by the document structure. For modeling the user behavior inside the context, PRUM relies on a probabilistic model. It can be parametrized to accommodate different settings. The model parameters can be estimated from user observations: for XML or Web IR, the probabilities of navigating from an element to its parent, descendants, or siblings or to follow a link can be estimated from the observation of user navigation behavior. Alternatively, the parameters can also be defined a priori: in Web retrieval, the user may follow a hyperlink with a probability inversely proportional to the total number of hyperlinks in the document (Pagerank [Lawrence et al. 1998] uses a similar approximation). In video IR, we could, for example, use the T2I model [Vries et al. 2004]: a returned result is an entry point in the video, and users watch the video from this point until their tolerance to irrelevant material is reached.

In this Section, the user model is described in detail (Section 2.1) and illustrated with some practical examples (Section 2.2). In Section 2.3, we then present three simplifying hypothesis and discuss their implications. These hypotheses are necessary in order to compute PRUM.

2.1 The Generic Navigating User Model

We have defined the context of an element x as the set of elements that are reachable through navigation from x. After the user selects the first element in the list, he can either navigate to an element from its context or continue to the second element of the list. For a given document element ranking, a stopping criterion is necessary for computing precision-recall values. The usual assumption is that the user stops after retrieving a given number of relevant documents [Voorhees 2003]. For our user model, we will suppose that the user stops consulting the list as soon as he sees a predefined number r of ideal elements, where r ranges from 1 to the number of ideal elements of a given query. Given precision at these r points, it is possible to construct a precision-recall curve by interpolation.

Following Raghavan et al. [1989], we divide the list consulted by the user into a ranked and an unranked part. This division of the list into two parts is made because probabilistic metrics like PRUM are defined *conditionally* to a given recall value. If the recall value cannot be reached, the metric is not well defined theoretically as the probability is conditioned to an impossible event. The list must then contain (an access to) all ideal elements needed for a given recall value. It is then necessary to add to the ranked list the remaining elements of the collection which is then considered unranked.

When a user has exhausted the ranked list, he continues consulting the remaining elements that were not ranked by the search engine. In this second part, there is no preferred list order. Said otherwise, each ordering of the unranked list has the same probability of occurrence. This hypothesis is also made by Raghavan and is somehow present in classic recall-precision, albeit implicitly, and justifies a null precision for recall values that cannot be reached in the ranked list. Because the unranked list contains a large number of documents, the probability for a user to find a relevant element is small. Note that for the TREC evaluation procedure, only the ranked part is considered.

For simplification, we will introduce two hypotheses. First, the ranked part of the list is supposed totally ordered, and second, one switches back to the classic user model for the unranked part of the list. Both hypotheses are made in order to simplify the computation of precision for the proposed user model. These two limitations are not so important in practice: the former because the effect of the ties will be smoothed when averaging over different query evaluations, the latter because the precision drops to very small values whenever the user has to consult the unranked list.

Figure 1 provides an example of user behavior in accordance with our model. Note that all the elements in this limited corpus belong to a unique document and the context of each of these elements is the whole document. Let F_k (ideal elements **F**ound at k) denote the number of distinct ideal elements seen while consulting the list until position k. If a user requires 4 ideal elements ($R = 4$), a particular sequence of steps he decides to follow could be the following:

(1) he consults element **a** in the list but does not see its context (except **a** itself). Element **a** is not ideal, hence $F_1 = 0$ and the user continues in the list;
(2) he consults element **b** but again not its context. This time, the element is ideal and $F_2 = 1$;
(3) he consults **c**, and this time decides to explore the context, discovering element **f** which is ideal ($F_3 = 2$);
(4) he consults **d**, explores its context and sees **f** and **i**. Element **i** is not ideal, while **f** is ideal, but it has already been seen by the user (F_4 still equals 2);
(5) he consults **e** and discovers **g** and **h** in its context. Both elements are ideal and seen for the first time (hence $F_5 = 4$). The user has found all the information he needs and stops consulting the list.

This example illustrates different facets of our user model. Steps (1) and (2) are similar to the standard user model. Step (3) illustrates how the user can navigate to an ideal element. In Step (4), we show that the same ideal element

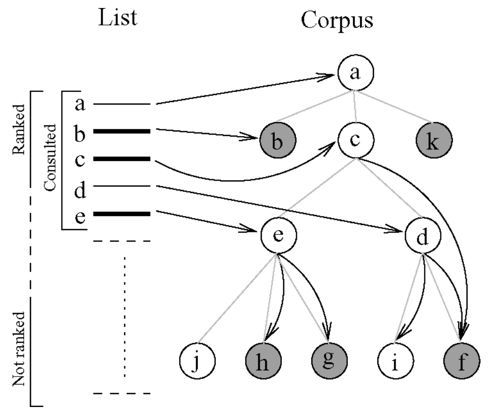

Fig. 1. The user wants to see 4 ideal elements (nodes filled with grey). He has consulted the first 5 elements of the list (**a**, **b**, **c**, **d**, **e**), 3 of them ((**b**, **c**, **e**)—bolder lines in the list) leading to ideal elements. Consulted elements are all within the same document whose structure is shown with grey lines: **a** is the root node and has three children (**b**, **c** and **k**), etc. Arrows show the browsing behavior of the observed user. For example, the user consults the third element of the list and navigates to an ideal one (**f**). The fourth element in the list also leads the user to an ideal element (**f**), but as it has already been seen, the list element is not an unseen element (hence the thin line for the list element). We also assume that elements **a**-**e** are in the ranked list. The list is composed of ranked and unranked elements.

can be retrieved multiple times in different contexts. Because it should not be counted twice, the user can only see an element for the first time once, F remains equal to 2. Eventually, Step (5) shows how an element can act as a so-called Best Entry Point [Lalmas and Moutogianni 2000] to more than one ideal unit: from the fifth consulted item in the list, the user sees two ideal elements for the first time.

Throughout this article, the verb *see* will have the following meaning. When a user sees an element, he figures out rapidly if it is ideal or not. If not, he quits or navigates to another element. In the XML case, the whole content of a seen element is not thoroughly read by the user and hence contained elements cannot be assumed to be seen. In the previous example, after rank 1, the user has seen element **a** but not element **b**, **c**, or **k**. We can imagine that element **a** contains a lot of irrelevant text before the element **b**, so that the user is discouraged before getting to **b**.

Now we introduce the probabilistic events used to model the user behavior. Notations and definitions are given in Tables I and II. At a given rank i, we say that the user has *consulted* an element x only if it belongs to the part of the list that was consulted by the user, and we denote it $x \in \mathcal{L}_i$. For a given list position i, an element x is *seen* (or found) by the user only if it belongs to the explored context of an element consulted between rank 1 and i. This event is denoted $x \in \mathcal{S}_i$. Because a user always navigates from a consulted element to itself, consulted elements are also seen elements: $x \in \mathcal{L}_i$ implies $x \in \mathcal{S}_i$. Let $\mathrm{P}(x \rightarrow y)$ denote the probability that the user reaches an element y from a consulted element x. The set of elements y for which $\mathrm{P}(x \rightarrow y) > 0$ formally defines the context of an element x.

Table I. Random Variables Used in the Model
The second column gives the domain of the random variable: $\mathcal{B}$ for boolean, $\mathbb{N}$ for integer. We assume that boolean variables take values 1 (for true) and 0 (for false). Realizations of variables are in lowercase (r for R, l for L).

Event	$\mathcal{D}$	Description
Events below are defined for an element in the corpus		
Lui	$\mathcal{B}$	***L**eads to a previously **U**nseen **I**deal. The* element under consideration leads to an ideal element *that was not previously seen.*
Cs	$\mathcal{B}$	*Consulted. The* element under consideration is in the part of the list *consulted by the user*. (Note that an element seen by the user during navigation that does not belong to the consulted list does not enter in this category)
Rk	$\mathcal{B}$	*Ranked. The* element under consideration belongs to the ranked part of the list.
AtRank_i	$\mathcal{B}$	*At Rank i. The* element under consideration is the ith of the list.
Other events		
R	$\mathbb{N}$	Number of *distinct* ideal elements the user requires to see. The user stops consulting the list after having seen exactly R elements.
F_i	$\mathbb{N}$	Number of *distinct* ideal elements which have been found by the user if he consults the first i elements in the list.
$x \in \mathcal{S}_i$	$\mathcal{B}$	The element x has been seen by the user if he consults the i first elements in the list.
$x \to y$	$\mathcal{B}$	The user navigated from element x to element y.

Table II. Model Parameters

Notation	Description
X	The set of all elements
$\mathfrak{I}$	The set of ideal elements for a given query need
$\mathcal{L}_i$	The set of elements consulted by the user up to rank i
$\mathbf{o}$	Size of the part of the list ranked by the search engine.
$\mathbf{u}$	Size of the part of the list composed of unranked elements.

Probability $\mathrm{P}(x \to y)$ may depend on the query and takes into account all the physical steps taken by the user in order to go from x to y: For example, if a Web page x has a link to z which in turn has a link to y, this is reflected in the probability $\mathrm{P}(x \to y)$. This situation is illustrated by Figure 3 (right). There are two indirect paths from **a** to **f**, summarized by the probability $P(\mathbf{a} \to \mathbf{f}) = 0.4$. Another example would be the following. Suppose we have measured the navigation of several users between three elements x, y, and z as depicted in Figure 2. Navigation from x would be summarized by the following probabilities: $\mathrm{P}(x \to x) = 1$, $\mathrm{P}(x \to y) = 0.5 + 0.3 \times 0.12 = .536$, and $\mathrm{P}(x \to z) = .3 + .5 \times .8 = .7$.

The set X denotes all the elements x of the database. Although the PRUM metric could be derived for a multivalued relevance scale, for simplification we will consider here a binary scale: the set of ideal elements is denoted $\mathfrak{I}$, its cardinality is $|\mathfrak{I}|$.

The parameters necessary to compute PRUM are the probability distribution $\mathrm{P}(x \to y)$ that the user browses from x to y and the set $\mathfrak{I}$. The latter can be derived from the assessments – from human judges, for example—associated with a query, while the former has to be defined in accordance with the user

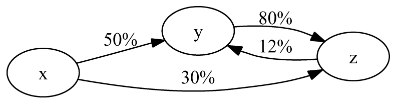

Fig. 2. An observation of user behaviors. The arrows depict the observed navigation patterns: 50% of the users navigated from x to y and 80% of them continued to z. 20% of the users did not navigate to other elements from x.

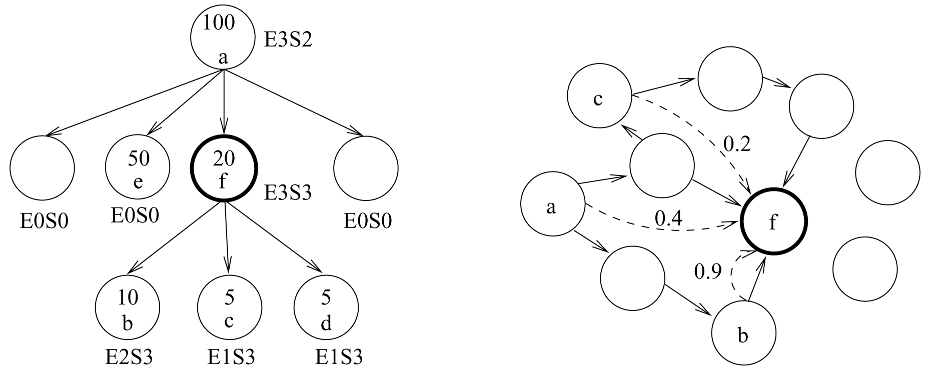

Fig. 3. XML (left) and Web (right) retrieval. The size of XML elements (in number of words) are shown inside the nodes (enclosing nodes thus contain more or as many words as their children) and INEX assessments are shown next to nodes. Web links are shown with plain arrows.

model adopted for the IR paradigm. Ideally, user experiments should be used in order to set P($x \rightarrow y$) probabilities. As it is not possible to measure an estimate for every couple of elements x and y, this probability could depend on some simple characteristics and on some parameters. The characteristics of x and y could be the size, the position inside the element (XML IR), or the average number of traversed links between x and y (Web IR). The parameters could be learned so that P($x \rightarrow y$) provides a good estimate of the ratio of users that, given one x in the list, would eventually reach y through navigation. Such experiments are not yet available for structured retrieval and for now P($x \rightarrow y$) are set using some heuristic function which does not rely on user observations but on basic assumptions about structural IR paradigms.

In the next section, we present some simple user models already proposed in the literature.

2.2 Comparison with Existing User Models

The navigating behavior relies on a set of hypotheses that are common to various previous metrics: (1) the user consults the elements following their order of appearance in the list returned by the search engine and stops when he has seen the information he needs; (2) an element is ideal even if a different element that contains the same (or a part of the) information has been seen previously; (3) two or more nonideal elements cannot become ideal even if they are merged.

The PRUM user model goes further by considering more complex user models. For illustration, we show in the following how it encompasses and generalizes models proposed in the context of XML retrieval.

Classical (No Browsing). The user does not browse at all: $\mathrm{P}(x \to y)$ is 0 if $x \neq y$ and 1 otherwise. This user model corresponds to the classic precision-recall user model. In this case, it can be shown that PRUM leads to the same values as standard IR precision and recall (see Appendix B).

Tolerance to Irrelevance (T2I). For video and XML retrieval, [Vries et al. 2004] proposed the T2I model. The list elements provide entry points in documents from where the user starts reading until he finds an ideal element or his tolerance to irrelevant material has been reached. This can be translated in terms of the PRUM user model by defining a transition probability between two elements of the same document to be 1 only if these two elements are close enough (the distance is usually measured in number of words) and 0 otherwise. Instead of being binary, the transition probability could also be inversely proportional to the distance between elements.

Structural. Gövert et al. [2003] and Kazai and Lalmas [2005] consider an implicit user model where the user is allowed to navigate within the hierarchy of XML elements. For example, he can browse from a subsection to its enclosing section or to one of its paragraphs. In this model, $\mathrm{P}(x \to y)$ is simply the ratio of the number of words in x and y if they are nested and 0 otherwise.

The computation of the navigational probabilities can take into account any available information, that, the relevance assessments, the structure of the documents, and so on. For example, emulating T2I requires considering relevance information for computing probabilities. The PRUM user model can thus model rather complex behaviors.

2.3 Simplifying Hypothesis

Assumptions on user behavior are necessary in order to define a tractable PRUM metric. We describe the 3 main assumptions of our model. We will use the notation $\mathrm{P}_r(\bullet)$ for $\mathrm{P}(\bullet | R = r)$ in the remainder of the article. The first simplification says that user behavior is not affected by the number r of ideal elements he wants to see.

Hypothesis 2.1. The navigating user behavior is independent of the number of ideal elements the user requires to see that

$$\forall x, y \in X \ \mathrm{P}_r(x \to y) = \mathrm{P}(x \to y).$$

The two following simplifying assumptions are necessary in order to derive the metric. The first states that a user navigates to an element independently of its navigation pattern to other elements.

Hypothesis 2.2. A user consults the context of an element independently of its previous navigation. Formally,

$$\forall x, y, \text{events } x \to y \text{ are mutually independent},$$

or equivalently[2], *for all set of distinct couples* (x_i, y_i)*:*

[2]A collection of events is mutually independent if the probability of the collection is the product of the probabilities of all events.

$$\mathrm{P}_r\left(\bigwedge_i x_i \to y_i\right) = \prod_i \mathrm{P}_r(x_i \to y_i)$$

(x_i, x_k):

$$\mathrm{P}_r\left(\bigwedge_{\{i,k\}} x_i \to x_k\right) = \prod_{\{i,k\}} \mathrm{P}_r(x_i \to x_k).$$

This hypothesis has several implications. First, it means that even if the user navigates from, for example, a section to its first paragraph, this does not give an indication on whether the user will navigate from the section to the second paragraph. This hypothesis is not so strong: the fact that a user can first browse from x to y, and then to z, is captured by the probability of the event $x \to z$. Let us consider the example of Figure 2. The probability $\mathrm{P}(x \to y \wedge x \to z)$ that the user navigates from x to both y and z can be computed:

(1) without simplifying hypothesis 2.2. We simply add the probabilities of the two possibilities: the user navigates first to y and then to z ($.5 \times .8 = 0.4$), or first to z and then to y ($.3 \times .12 = .036$). This gives a probability of .436.
(2) using hypothesis 2.2. We multiply the probability of navigating to y (navigating first by z or not) and to z (navigating first by y or not): $\mathrm{P}(x \to y) \times \mathrm{P}(x \to z) = .7 \times .536 = .3752$

The difference is less than what it would be if we considered only the direct navigation from x to y and z. Their probability would be equal to .5 and .3, respectively, yielding a joint probability $\mathrm{P}(x \to y \wedge x \to z)$ of .15 only.

The main reason for hypothesis 2.2 is that it simplifies the computation of the probability of event $x \in \mathcal{S}_i$ that a user saw an element x after he consulted the i first ranks of the returned list. Let us briefly examine how this simplification works. The event $x \in \mathcal{S}_i$ is true if and only if the user consulted a list item, possibly x itself, that led him to see x. Formally,

$$x \in \mathcal{S}_i \equiv \bigvee_{y \in \mathcal{L}_i} y \to x.$$

Hypothesis 2.2 implies that all causes (the user saw an element y in the list and browsed to x) are independent of each other in terms of their abilities to influence the consequence (the user has seen x). This is the *noisy-or* hypothesis [Heckerman and Breese 1994] which is often used in probabilistic models when an event is the consequence of many causes. Formally, we can show that under Hypotheses 2.2 and 2.1, the probability of the event $x \in \mathcal{S}_i$ simplifies to

$$\mathrm{P}_r(x \in \mathcal{S}_i) = 1 - \prod_{y \in L_i} (1 - \mathrm{P}(y \to x)). \tag{1}$$

This leads to a realistic model whose complexity is linear in terms of the number of elements leading to x. Without this hypothesis, computation of $\mathrm{P}_r(x \in \mathcal{S}_i)$ becomes rapidly intractable. We illustrate the computation of Equation (1) using the example of Figure 4. If a search engines returns the list (**a**, **b**, **c**), then after element **a** is consulted, the probability that the user sees the ideal

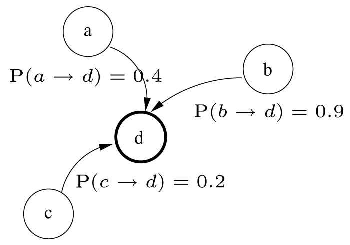

Fig. 4. An ideal element **d** which belongs to the context of three consulted elements (**a**, **b**, **c**). Probabilities of navigation are shown next to the arrows.

element **d** is

$$1 - (1 - 0.4) = 0.4.$$

After element **b**, the probability is

$$1 - (1 - 0.4)(1 - 0.9) = 0.94,$$

and after element **c,** it becomes

$$1 - (1 - 0.4)(1 - 0.9)(1 - 0.2) = 0.952.$$

Consequently, 95.2% of users would see **d** after consulting elements **a** to **c**. Without the noisy-or assumption, this probability would require the knowledge of the interaction of the different navigations between **a**, **b**, **c**, and **d**. In practice, the number of such interactions grows exponentially with the number of elements in the context and becomes intractable.

Our last hypothesis is related to the computation of the probability that the user sees an ideal element for the first time.

Hypothesis 2.3. The collection of events such that the user has seen an ideal element x at rank i for the first time for all $x \in \Im$, are mutually independent knowing the number of ideal elements already seen at the previous rank. This is a noisy-or hypothesis and the following result holds:

$$\mathrm{P}_r\left(\bigvee_{x\in\Im} x \in \mathcal{S}_i \wedge x \notin \mathcal{S}_{i-1} \middle| F_{i-1}=s\right) = 1 - \prod_{x\in\Im}(1 - \mathrm{P}_r(x \in \mathcal{S}_i \wedge x \notin \mathcal{S}_{i-1} | F_{i-1}=s)).$$

This is also a simplification of the reality needed for the calculation of the precision values. Without this hypothesis, we would have to consider all the possible cases where the user has seen for the first time $k \in \{1, \ldots, |\Im|\}$ ideal elements at rank i; this would imply $\mathcal{C}_k^{|\Im|}$ possibilities which is prohibitive in practice. Like Hypothesis (2.2), Hypothesis (2.3) permits capturing a minimal interaction of the different causes (the user has seen for the first time a given ideal element x) on their common consequence (the user has seen for the first time at least one ideal element). With this hypothesis, the computation becomes linear with respect to the number of ideal elements for which the probability of being seen has increased.

For the unranked list, using the navigating user model would be intractable. Therefore, we switch back to the nonbrowsing user model for this nonranked part, and the preceding hypotheses need only hold for the ranked part.

3. PRECISION RECALL WITH USER MODELING

Having described the user model, we are now in position to formally specify the metric. PRUM is an extension of the probabilistic precision-recall proposed by Raghavan et al. [1989]. Precision is defined with reference to a recall value, identified here as a predefined number of ideal elements that the user wants to see. In our case, the number of elements that need to be consulted to achieve this recall value is stochastic because it depends on the user behavior, that is, on his decision to navigate between elements.

In the next section, we define and motivate our metric. We illustrate its principles with a toy example. We then describe synthetically the equations that fully define the PRUM metric. The complete derivation is provided in Appendix A.

3.1 Extension of Raghavan's Definition

The probabilistic definition of recall-precision presented by Raghavan et al. [1989] is more general than the one used in TREC [Voorhees 2003]. According to their definition, the list consulted by the user for a given information need contains all the elements of the database. The user, who searches for exactly r ideal elements, stops consulting the list as soon as he finds them. In this context, the recall level ℓ is the ratio of r to the total number $|\mathfrak{I}|$ of ideal units. Precision is defined as the probability that an element consulted by the user is ideal. Formally, precision at recall level ℓ is written $\mathrm{P}(Ideal|Cs, L = \ell, Q = q)$, where Cs means the element is in the consulted part of the list and Ideal means the element is ideal, while Q expresses the user's query.

We generalise Raghavan's definition of relevance to fit our more complex user model. In the PRUM user model, an element of the list might lead the user to see other elements of the database even if it is not ideal per se. Also, a user might see an (ideal) element more than once. In Figure 4, elements **a**, **b**, and **c** can all lead to the ideal element **d**, and a search engine retrieving them should be rewarded. Besides the notions of relevant and ideal elements, we then introduce the new concept of *leading to a previously unseen ideal element*, denoted as *Lui*. If we ignore user browsing (i.e., a consulted element only leads to itself), this definition degrades to the element is a previously unseen ideal element. If we further suppose that there are no duplicates in the list, then it becomes the element *is* ideal, which is the classica definition (ideal and relevant are equivalent in standard IR).

In order to average the results over queries, precision is usually computed at several recall levels ℓ between 0 and 1. Precision in PRUM metric is defined for a given query q (and its associated assessments) at recall level ℓ as:

$$\mathrm{P}(Lui|Cs, L = \ell, Q = q),$$

which is, in the context of a given query q, the probability that an element consulted by the user leads to an unseen ideal element if the user is searching for ℓ% of the ideal elements present in the database. We also suppose that an element leads to itself. In order to simplify notations, we omit from now on

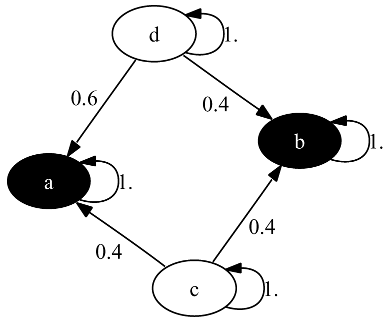

Fig. 5. An example in Web retrieval. Nodes are the different documents of the database: white nodes are irrelevant documents, while black nodes are ideal. Navigational probabilities are shown next to the edges.

the conditioning on $Q = q$ for all the probabilities and events as the metric is computed for each query separately.

Precision is usually computed for integer values, and interpolation techniques are used to derive precision at an arbitrary recall level. We chose to use the ceiling interpolation procedure of TREC [Voorhees 2003] where precision for a given recall level ℓ is the maximum of precision for a recall value superior or equal to ℓ. We thus restrict the computation of PRUM to recall values between 1 and $|\mathfrak{I}|$ in the remainder of the article, and we need to evaluate the precision at a given recall r:

$$\mathrm{P}_r(Lui|Cs). \tag{2}$$

In the rest of this section, we summarize PRUM derivation and show how to compute the measure for any recall value. We also show in Appendix B that the PRUM formula reduces to the Raghavan's one when the user model is the classic IR one.

3.2 PRUM Illustration

Let us illustrate the computation of precision with PRUM using the small database in Figure 5. Note that the example is computed without some of the hypotheses we made; PRUM is an approximation of the results described in the following. It is composed of four Web documents (**a**-**d**) with links between them. Elements **a** and **b** are ideal units to be retrieved for a given query, while elements **c** and **d** are not ideal themselves but can lead the user to both **a** and **b**. We set the navigational probabilities so that

—60% of users navigate from **d** to **a**;

—40% of users navigate from **c** to **b**, from **c** to **a**, or from **d** to **b**.

How do we evaluate the performance of an engine that returns the list (**c**, **d**, **a**, **b**)? Table III gives a summary of the different navigational probabilities (Table of scenarios). The first column of the upper table corresponds to the consulted list. The other columns indicate at which rank ideal elements **a** and **b** are first seen according to the different scenarios. For example, the second column evaluates to .16 the probability that a user navigates to elements **a**

Table III.
This table summarizes the different situations that might happen if the database is the one depicted in Figure 5 and if the list returned by the system is (**c**, **d**, **a**, **b**). The upper part of the table shows the list position where either **a** or **b** is first seen. The row labeled **PS** (probability of the scenario) gives the probability that such a situation is observed. Row **CL** (consulted and leading to a previously unseen ideal) gives the number of ranks where a consulted element leads to an ideal one, while rows **C** (consulted) give the number of ranks the user has to consult in order to reach a given recall value. The **Exp** column gives the expected values of **CL** and **C** over the different configurations

	Table of Scenarios									
Rank	Ideal Elements (seen for the first time)									
1 (c)	**a, b**	**a**	**a**	**b**			**b**			
2 (d)		**b**		**a**	**a, b**	**a**		**b**		
3 (a)							**a**	**a**	**a**	
4 (b)			**b**			**b**			**b**	***Total***
PS	.16	.096	.144	.144	.0864	.1296	.096	0.0576	.0864	*1*
	Search for one Ideal Element									**Exp**
CL(1)	1	1	1	1	1	1	1	1	1	*1*
C(1)	1	1	1	1	2	2	1	2	3	*1.45*
	Search for two Ideal Elements									**Exp**
CL(2)	1	2	2	2	1	2	2	2	2	*1.75*
C(2)	1	2	4	2	2	4	3	3	4	*2.71*

and **b** from the first element returned by the engine (**c**). Similarly, the third column represents the case where a user discovers element **a** navigating from **c** and element **b** when navigating from the second element of the engine list. The probability of such a scenario is .096 since:

$$\mathrm{P}(\mathrm{usernavigatesfrom}\ \mathbf{c}\mathrm{to}\ \mathbf{a}) \times \mathrm{P}(\mathrm{userdoesnotnavigatefrom}\ \mathbf{c}\mathrm{to}\ \mathbf{b}) \\ \times \mathrm{P}(\mathrm{usernavigatesfrom}\ \mathbf{d}\mathrm{to}\ \mathbf{b}) = 0.4 \times 0.6 \times 0.4 = 0.096.$$

Note that in this example we don't have to multiply by the probability that the user navigates from **d** to **a** or not since both events are compatible with the scenario, and thus would sum to 1 by marginalization. It also means that PRUM does not make the difference between seeing an ideal element once or more than once. The other configuration probabilities are computed likewise. The probabilities over all scenarios naturally sum to one.

Table III shows the number **C**(r) of list elements that a user consulted and the number **CL**(r) of consulted elements that actually led the user to discover an ideal element, where r is the number of ideal elements wanted by the user. The column ***Exp*** reports the expected values for each corresponding row. For a given line, they are computed by multiplying each cell by the corresponding probability of scenario (PS), and then by summing all these values, that is. The precision in PRUM is defined as the ratio of two quantities, that is, the expected value of **CL**(r) over the expected value of **C**(r). The precision at recall 1 is $1/1.45 \simeq 0.691$ and, at recall 2, equals $1.75/2.71 \simeq 0.646$. These values are superior to the standard precision-recall estimates (0.333 and 0.5, respectively).

This is the expected behavior since the first two elements returned by the system (**c** and **d**) can lead the user to see both ideal units.

We have now seen how to compute precision. In practice, the number of scenarios is combinatorial and this naive enumeration approach cannot be used. Computations in PRUM are then performed using the assumptions introduced in Section 2.3. The values computed by PRUM equations in the previous example are 0.691 and 0.636, respectively, which are close to the exact values given. The difference is not important since (1) all the systems will be evaluated with the same bias, and (2) the user model will always be an approximation of the reality.

3.3 PRUM Metric

We now show how PRUM can be computed for a query and for an arbitrary recall value r. Precision is then interpolated for any recall value r. This allows one to average precision over queries. To clarify the presentation, we chose to postpone the formal derivation to Appendix A and only present the formulas that permit the effective computing of PRUM along with intuitive motivations.

Starting from Equation (2), we decompose the probability that a consulted list item leads the user to discover an ideal element $\mathrm{P}_r(Lui|Cs)$ using the Bayes rule:

$$\mathrm{P}_r(Lui|Cs) = \frac{\mathrm{P}_r(Lui, Cs)}{\mathrm{P}_r(Cs)}. \tag{3}$$

As introduced in Section 2, the list is divided into a ranked and an unranked part. Equation (3) is decomposed accordingly:

$$\frac{\overbrace{\mathrm{P}_r(Lui, Cs, Rk)}^{(a)} + \overbrace{\mathrm{P}_r(Lui, Cs, \neg Rk)}^{(b)}}{\underbrace{\mathrm{P}_r(Cs, Rk)}_{(c)} + \underbrace{\mathrm{P}_r(Cs, \neg Rk)}_{(d)}}, \tag{4}$$

where Rk means that the element is in the ranked part of the list.

Parts (a) and (c) in Equation (4) correspond to the ranked part of the list and are computed with the PRUM user model. Parts (b) and (d) correspond to the unranked part and are computed with the classic user model. Note that the unranked list is usually composed of many more elements than the ranked one: the probability of finding an element leading to an unseen ideal will then be low, and, in such cases, the difference between the two user models will be small.

We derive each term of the previous equation in Appendices A.4 (for a and b) and A.3 (c and d). In Appendix A, we show that (a) and (c) can be expressed in terms of the probabilities $\mathrm{P}_r(F_i = s)$ and $\mathrm{P}_r(F_i > s|F_{i-1} = s)$.

Equation (4) can be written in terms of these probabilities:

$$\mathrm{P}_r(Lui|Cs) = \frac{\overbrace{\sum_{s=0}^{r-1}\sum_{i=1}^{\mathbf{o}} \mathrm{P}_r(F_{i-1}=s)\mathrm{P}_r(F_i > s|F_{i-1}=s)}^{(a)} + \overbrace{\sum_{s=0}^{r-1}\mathrm{P}_r(F_{\mathbf{o}}=s)(r-s)}^{(b)}}{\underbrace{\sum_{s=0}^{r-1}\sum_{i=1}^{\mathbf{o}} \mathrm{P}_r(F_{i-1}=s)}_{(c)} + \underbrace{\sum_{s=0}^{r-1}\mathrm{P}_r(F_{\mathbf{o}}=s)(r-s)\times\left(1+\frac{\mathbf{u}-(|\mathfrak{I}|-s)}{|\mathfrak{I}|-s+1}\right)}_{(d)}}, \tag{5}$$

where $\mathbf{o}$ is the size of the ranked list, and $F_{\mathbf{o}}$ is the number of ideal elements found at rank $\mathbf{o}$. $\mathbf{u}$ is the size of the unranked list, and $|\mathfrak{I}|$ is the number of ideal elements. Let us now briefly explain the role of each term in the this equation.

(c) is equivalent to$\sum_{i=1}^{\mathbf{o}} \mathrm{P}_r(F_{i-1} < r)$. First consider the deterministic case where there is no navigation. $\mathrm{P}_r(F_{i-1} < r)$ is equal to 1 for all the ranks between 1 and the first rank where the user sees r ideal units. Therefore,$\sum_{i=1}^{\mathbf{o}} \mathrm{P}_r(F_{i-1} < r)$ is simply equal to the number of consulted elements in the ranked list when the user searches for r ideal elements. In the general case, its value will be the expected number of consulted elements in the ranked list when searching for r ideal elements. Note that by definition the user has not found any ideal element before consulting the list so that $\mathrm{P}_r(F_{\mathbf{o}} = 0) = 1$.

(a) is equivalent to $\sum_{i=1}^{\mathbf{o}} \mathrm{P}_r(F_{i-1} < r,\ F_i > F_{i-1})$. This is the expected number of consulted elements in the ranked list that lead a user searching for r ideal units to see for the first time one or more ideal elements.

(b) is the expected number of ideal and consulted elements after position $\mathbf{o}$.

(d) is the expected number of consulted elements after position $\mathbf{o}$.

The ratio of (a + b) to (c + d) thus corresponds to the (probabilistic) precision at r. This is the expected number of consulted elements which lead the user to discover an ideal element divided by the expected number of elements he consulted. We now describe how the different probabilities in the formula are computed.

We will need to calculate $\mathrm{P}(x \in \mathcal{S}_i)$. According to Equation (1), this can be computed as

$$\mathrm{P}_r(x \in \mathcal{S}_i) = 1 - \prod_{y \in \mathcal{L}_i}(1 - \mathrm{P}(y \to x)). \tag{6}$$

The probability $\mathrm{P}_r(F_i = s)$ that the user has found s ideal elements while consulting the list up to rank i is derived in Appendix A.1. The final expression used for computing the precision is

$$\mathrm{P}_r(F_i = s) = \sum_{\substack{A\subseteq\mathfrak{I}\\|A|=s}} \overbrace{\prod_{x\in A}\mathrm{P}(x \in \mathcal{S}_i)\prod_{x\in\mathfrak{I}\backslash A}\mathrm{P}(x \notin \mathcal{S}_i)}^{(*)}, \tag{7}$$

where A is a subset of cardinality s of the ideal set $\mathfrak{I}$. For a given A, $(*)$ is the probability that the user has seen exactly the s ideal elements of the subset A of $\mathfrak{I}$ and no other ideal element. Then all the (exclusive) combinations for which the user has seen exactly s ideal elements are enumerated when summing over all the possible subsets A of cardinality s. This can be computed in a time quadratic with respect to the number of elements $\mathcal{N}$ for which $1 > \mathrm{P}(x \in \mathcal{S}_i) > 0$.[3] In practice, it can also be approximated by a normal law when $\mathcal{N}$ is large (see Appendix A.1). In our implementation when $\mathcal{N} > 10$, we used the normal approximation.

In Appendix A.2, Equation (12) shows how to compute the probability $\mathrm{P}(F_i > s|F_{i-1} = s)$ that the user discovers an ideal element at position i, knowing that he has seen s ideal elements before:

$$\mathrm{P}_r(F_i > s|F_{i-1} = s) = 1 - \prod_{x \in \mathfrak{I}} \left(1 - \frac{\overbrace{(\mathrm{P}(x \in \mathcal{S}_i) - \mathrm{P}(x \in \mathcal{S}_{i-1}))\mathrm{P}\,(F_{i-1} = s\,|x \notin \mathcal{S}_{i-1})}^{(**)}}{\mathrm{P}(F_{i-1} = s)} \right), \quad (8)$$

where the probability $\mathrm{P}\,(F_{i-1} = s\,|x \notin \mathcal{S}_{i-1})$ is computed using Equation (7), where $\mathfrak{I} \setminus \{x\}$ is substituted for $\mathfrak{I}$. Said otherwise, we identify $\mathrm{P}(F_{i-1} = s|x \notin \mathcal{S}_{i-1})$ to the probability that the user sees s ideal elements from a corpus where an ideal element x has been removed. The ratio $(**)$ is equal to $\mathrm{P}(x \in \mathcal{S}_i \wedge x \notin \mathcal{S}_{i-1}\,|F_{i-1} = s\,)$, the probability that the user, who has seen s ideal elements at rank i, has discovered the ideal element x at rank i but not before. The product $\prod_{x \in \mathfrak{I}}$ over all ideal elements is a noisy-or, it combines all the possible causes that lead the user to a newly-ideal element. If one ideal element x has been seen only after the user consulted rank i, that is, $\mathrm{P}(x \in \mathcal{S}_i \wedge x \notin \mathcal{S}_{i-1}\,|F_{i-1} = s\,) = 1$, then the product $\prod_{x \in \mathfrak{I}}$ is equal to 0, and the probability $\mathrm{P}_r(F_i > s|F_{i-1} = s)$ that the user saw for the first time an ideal element at rank i is 1. If for each list element x at rank i the user's probability to see for the first time an ideal element is not increased, then $\mathrm{P}(x \in \mathcal{S}_i \wedge x \notin \mathcal{S}_{i-1}\,|F_{i-1} = s\,)$ is equal to 0 for all x . The probability $\mathrm{P}_r(F_i > s|F_{i-1} = s)$ thus equals 0. It is between 0 and 1 in other cases.

Equation (5) can now be fully computed using Equations (7), (8), and (6). This equation provides us the precision at recall r. Interpolation allows us to compute precision at any arbitrary recall level ℓ. The precision at ℓ for multiple queries can be then averaged.

4. COMPARISON WITH STATE-OF-THE-ART METRICS IN THE XML DOMAIN

In this section, we compare PRUM with metrics recently proposed for structured document retrieval. Most new metrics have been proposed for XML, so we will mainly focus our comparison on this domain. We first describe in

[3]Generally, most elements have not even been partially seen ($\mathrm{P}(x \in \mathcal{S}_i) = 0$) at low ranks, and the number of ideal elements seen with a probability 1 tends to augment the number of consulted ranks.

Section 4.1 the current metrics and their limitations using a simple example for illustration. In Section 4.2, we compute precision on two simple examples for illustrating the computation of PRUM and for comparing the behavior of the three metrics.

Most XML metrics were developed within the INEX[4] initiative. They make use of a two-dimensional four-graded scale [Kazai et al. 2003] assessment. Its dimensions are:

Specificity (spe). The extent to which a document component is focused on the information need;

Exhaustivity (exh). The extent to which the information contained in a document component satisfies the information need.

An assessment in the INEX scale is denoted ExSy, where x and y are, respectively, the exhaustivity and specificity levels which may take four values between 0 (not exhaustive/specific), 1, 2, or 3 (maximal exhaustivity or specificity). Exhaustivity always increases when going up in the XML hierarchy as bigger elements may contain more ideal information. Starting from an S3 node and going up in the tree, specificity usually decreases as bigger elements tend to contain more irrelevant information.

4.1 Current Metrics

Metrics should be able to handle the specific problems encountered with semistructured IR. Important issues, as mentioned in the introduction, are (1) the need to reward near misses, that is, elements in the context of an ideal element, (2) the ability to deal with overlapping retrieved elements, and (3) the ability to accommodate *best entry points*, that is, elements that lead to more than one ideal element. We analyze different metrics based on their respective ability to deal with these problems in the following.

First metrics proposed for XML evaluation were variations of recall-precision. They are denoted GRP for generalized RP and presented first. Metrics using more sophisticated user models are presented next.

4.1.1 *Variations of Recall-Precision Metrics (GRP and GRP_{ng}).*

Near misses can act as entry points leading to one or more ideal elements as illustrated in Figure 3. The left part represents an XML document with one ideal element **f**. Possible answers to a query for which **f** is ideal are a sibling (**e**), a descendant (**b**), or the parent (**a**) of the ideal element, from where the ideal element can be accessed. In the case of Web retrieval (right side of the Figure), returning element **b** or **a**, for example, can lead the user to **f** through hyperlinks. It is generally admitted [Kazai et al. 2004] that near misses should be rewarded but to a lesser extent than exact answers.

The GRP model uses a very simple way to take context into account: it gives a positive score, lower than for the ideal element, to near misses [Kazai 2003]. The amount of relevance for the near miss is determined by a specific

[4]Initiative for the Evaluation of XML Retrieval `http://inex.is.informatik.uni-duisburg.de/index.html`.

quantization function. Unfortunately, this way of rewarding near misses leads to an *overpopulation* of the recall base [Piwowarski and Gallinari 2003; Kazai et al. 2004]. The problem is illustrated in Figure 3. Suppose **a**, **b**, **c**, and **d** are rewarded, respectively, by 0.75, 0.75, 0.5, and 0.5, because elements **a** to **d** now contribute to the relevance. A system must return all 5 elements (**a**, **b**, **c**, **d**, and **f**) to achieve 100% recall level. On the other hand, a perfect system which returns the element **f** achieves only $1/(1+0.75+0.75+0.5+0.5)\times 100 = 28.6\%$ recall level.

To restrict overpopulation, we might attempt to prevent the same ideal information from being counted twice. Gövert et al. [2003] proposed the $\mathrm{GRP}_{\mathrm{ng}}$ recall-precision metric where the score of the consulted element is proportional to the ratio of unseen material in this consulted element, counted in number of words. Let us use again the XML example in Figure 3 to illustrate this idea. Suppose **a**, **b**, **c**, and **d** are rewarded respectively by 2/3, 2/3, 1/3, and 1/3. If element **a** is consulted before **b**, its contribution to recall is $\frac{2}{3}$ since no part of **a** has been seen previously. On the other hand, if **a** is consulted after **b**, the contribution of **a** to recall is computed as $\frac{2}{3}\times(1-\frac{10}{100}) = 0.6$ because 10/100 of **a** has been seen previously with element **b**. One drawback of this approach is that recall can be superior to the number of truly ideal elements. If **b** and then **a** are seen, recall amounts to

$$\underbrace{\frac{2}{3}}_{\mathbf{b}\text{ reward}} + \underbrace{\left(1-\frac{10}{100}\right)}_{10/100\text{ of }\mathbf{a}\text{ has been seen}} \times \underbrace{\frac{2}{3}}_{\mathbf{a}\text{ reward}} = 1.27.$$

Another limitation is the dependence of the recall base on the list order. If **a** is consulted before **b**, the contributions of **a** and **b** are $\frac{2}{3}$ and 0, respectively, leading to a total recall of .66, significantly inferior to 1.27. Two systems retrieving the same elements are subject to achieving dramatically different scores depending on the element order: recall of **a** followed by **b** is different from the recall of **b** followed by **a**. We shall refer to this problem as the *recall instability*.

GRP and $\mathrm{GRP}_{\mathrm{ng}}$ are simple extensions of recall-precision measures. They cannot simultaneously accommodate near misses and offer a consistent model. They also fail to take nonoverlapping near misses into account. In Figure 3, for example, element **e** is not rewarded, although it is a near miss.

They cannot handle best entry points [Lalmas and Moutogianni 2000] either. A BEP is an element that leads to several ideal elements like a Web page with hyperlinks to several ideal pages or like a section in an XML document containing several ideal paragraphs. Assume that a section containing the only two ideal paragraphs for a given query is returned by the search engine. If only the section is consulted, GRP and $\mathrm{GRP}_{\mathrm{ng}}$ parameters cannot be set to give a 100% precision and recall which is not consistent. Note that the classic definition of precision also leads to inconsistencies in this case: retrieving the section leads to a score superior to 1 because one element is consulted while two ideal elements are seen.

PRUM metrics handle these problems gracefully with the concept of *Lui* elements. Only ideal elements are *directly* rewarded, but IR systems returning a near miss are rewarded indirectly because doing so increases the probability of finding an ideal element. They will, however, be ranked below those directly returning the ideal elements. There is neither recall base overpopulation nor recall instability. For BEP, PRUM handles consistently the cases where more than one ideal element can be reached from a single consulted one.

4.1.2 *Metrics with a Navigating User Model.* Generalized recall-precision metrics proposed so far rely on a basic user model in which, for flat document collections, the user inspects the ranked elements sequentially in decreasing order of relevance without navigating to proximal elements. This contradicts the nature of structured corpora in which the context of an element is an important source of information.

Other measures specific to XML retrieval have been developed. Kazai et al. [2004] proposed a modification of the cumulated gain criterion [Kekäläinen and Järvelin 2002] called xCG. We will only discuss the basic xCG measure, but our comments remain valid for different variations of this metric which were proposed later. For each rank of the list, xCG computes a gain that depends on the judgement associated with the element and on the previous element consulted by the user (see Appendix C.3 for more details).

The xCG metrics manage to overcome GRP limitations while maintaining a simple definition. Near misses can be taken into account without expanding the recall base and overlap is also considered. This is clearly an improvement over the basic PR measures discussed before.

Compared to PRUM, there are two important differences. First, xCG has been developed specifically for a special case of XML retrieval. For example, navigation is allowed down and up the XML tree, and near misses, which are not direct ancestor or descendants, are not considered. Second, there is no well defined and explicit user model. The implicit user model of xCG is related in some way to the metric parameters, but this relation is neither formally defined nor explicit. For example, overlap is not rewarded when fixing the weighting factor α to the specific value of 1. Doing so also limits the flexibility of the user model. For other values of the weighting factor, the metric may present small incoherences. As a consequence, xCG extension to other structured information retrieval paradigms or to more realistic user models would be problematic.

More sophisticated user models have been defined previously by Quintana et al. [1993] for Web retrieval and by Dunlop [1997] in the more general context of IR interfaces used to present results. Both models take into account the user effort (measured in elapsed time) for accessing relevant information. An interesting extension of the latter work [Vries et al. 2004] proposed the Tolerance To Irrelevance (T2I) metric, which is based on Cooper's Expected Search Length Measure [Cooper 1973]. The core of this proposal is a user model considering overlap and near misses. The user consults the returned elements until he finds relevant information or until his tolerance (counted in number of words, e.g.,) to irrelevant material is reached. In both cases, he proceeds to the next element in the list. As with PRUM, the T2I model only rewards an ideal element reached through navigation once. The direct use of Raghavan's formula for T2I implies

Table IV. Summary of Metric Properties for XML Retrieval
The first row (precision+recall) indicates which metric supports both dimensions. The second row states whether the metric relies on an explicit user model. The third and fourth rows indicate whether the metric handles near misses and overlap. RP is defined in Raghavan et al. [1989], GRP in Kazai [2003], GRP_{ng} in Gövert et al. [2003], XCG in Kazai et al. [2004], T2I in Vries et al. [2004], and GR in Piwowarski and Gallinari [2003]

PropertyMetric	RP	GRP	**GRP** $_{\text{ng}}$	T2I	XCG	GR	PRUM
Precision and Recall	yes	yes	yes	yes (a)	yes	no	yes
Formal derivation from an explicit user model	yes	no	no	yes	no	yes	yes
Near misses	no	partial	partial	yes	yes (b)	yes	yes
Overlap	no	no	partial	yes	yes (c)	yes	yes
Graded assessments	no	yes	yes	no	yes	yes	no

(a) but not formally exact
(b) only for nodes assessed relevant by judges
(c) with parameter α set to 1

that precision is computed under the assumption that an element leads to at most one ideal element: this can lead to a precision strictly superior to 1. In PRUM, one or more ideal elements can be reached from a single rank without problem. PRUM estimates the number of list elements leading to at least one previously unseen ideal element. Also, the relevance score of an element drops from one to zero depending on a preset number of words that precede it (the user behavior is deterministic while it is stochastic in PRUM), which seems too strict for our purpose. The T2I user model can be used in our metric by appropriately setting the navigational probabilities of the PRUM user model; said otherwise, PRUM includes T2I as a special case.

The PRUM user model was first introduced in Piwowarski and Gallinari [2003] in the context of the generalized recall (GR) measure[5]. This metric, however, lacked one dimension, that is, the precision, and is therefore less expressive than PRUM.

A summary of the different metrics and their properties is given in Table IV. Metrics like GRP, GRP_{ng}, and xCG need graded relevance judgements in order to express the fact that an element is not ideal but a near miss. In PRUM, this is handled by the navigation behavior of the user and the specific definition of seeing an element used in our model. For instance, suppose that an element has a quantization 1 and its parent 0.75 for GRP; for PRUM, the former would be ideal (i.e., a system returning this element would be fully rewarded), while the latter would not be ideal but have a probability of navigation to the ideal one of 0.75.

4.2 Examples

In this section, we illustrate some of the properties of PRUM through two simple examples. The first one shows how PRUM solves overlap and near miss problems, while the second concerns best entry points. In the discussion, we

[5]Also known as ERR (Expected Ratio of Relevant units).

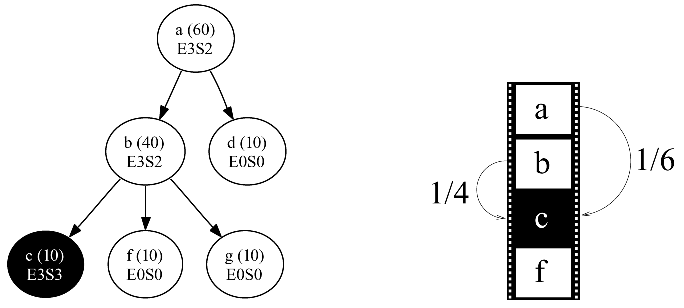

Fig. 6. (Left) An XML document with an ideal element (**c**). The size of an element is shown between brackets after the element name. All leafs contain 10 words. Other nodes contain node 10-specific words plus the words of their children. For example, **a** (respectively **b**, **c**) contains 60 words (respectively 40, 10). The assessment (INEX scale) is shown in the lower part of the nodes. On the right, a video with four scenes (**a, b, c**, **f**) is depicted. The scene **c** is an ideal unit for the user's query need; one sixth (respectively one fourth) of the users would see **c** if **a** (respectively **b**) were returned.

also compare PRUM to GRP, $\mathrm{GRP}_{\mathrm{ng}}$, and xCG, the three metrics discussed previously.

4.2.1 *Example* 1. *Hierarchical Navigation Within XML Documents.* Consider the toy example of Figure 6. This example concerns a single document with six elements where only **c** is ideal. The left part of the figure is a tree-like XML document while the right part is a portion of a video composed of four scenes. We will compute the scores for the three metrics of two search engines returning the same elements in different order: **c**, **b**, **a** and **a**, **b**, **c**, respectively. We expect the first search engine to be scored higher since it returns the unique ideal element in first position.

Formal definitions of GRP, $\mathrm{GRP}_{\mathrm{ng}}$, and xCG can be found in Appendix C. The computation of PRUM requires the specification of the probability of transition between elements $\mathrm{P}(x \to y)$. Consider first the tree-like document. In order to keep the example simple, we restrict the user to moving only up and down in the XML hierarchy and define heuristically the transition probability as follows:

$$\mathrm{P}(x \to y) = \begin{cases} \mathrm{length}(x)/\,\mathrm{length}(y) & \text{if } y \text{ contains } x \\ \mathrm{length}(y)/\,\mathrm{length}(x) & \text{if } x \text{ contains } y \\ 0 & \text{otherwise.} \end{cases} \quad (9)$$

We set the length of each element to 10 words more than the total length of included elements. For example, element **b** contains 40 words, **c** 10 words, and the probability of transition from **b** to **c** is $10/40 = 0.25$.

For the video example, we set the navigation probabilities between **a/b** and **c** so that they match their counterpart in the XML example: one sixth (respectively one fourth) of the users would browse from **a** (respectively **b**) to **c**. All results that follows also hold for this example.

The good search engine. The first engine returns the three elements **c**, **b**, **a**, in this order, presenting the unique ideal element first.

The GRP metric attributes a recall of 2.5 (2×0.75 for **a** and **b**, and 1 for **c**) to the engine when the whole recall base has been discovered

$$\text{precision} = \frac{2.5}{2.5 + 0.25 + (0.25 \times 0.75)/(0.75 + 1)} \simeq 0.88.$$

The precision is thus inferior to 1 even though the ideal element has been seen first.

Since **c** is entirely new (10 words), $1 - \frac{10}{40} = \frac{3}{4}$ of **b** are new (30 words) and $1 - \frac{40}{60} = \frac{1}{3}$ of **a** are new (20 words), The score calculated by GRP_{ng} is

$$\begin{aligned} \text{precision} &= \frac{10 \times 1 + 30 \times \frac{3}{4} + 20 \times \frac{1}{3}}{10 + 30 + 20} \simeq 0.65 \\ \text{at recall} &= \frac{1 \times 1 + 1 \times \frac{3}{4} + 1 \times \frac{1}{3}}{1 + 1 + 1} \simeq 0.69 \end{aligned}$$

Precision is still less than 1, and recall level is inferior to 100%. This is the effect of the recall instability. For GRP_{ng}, a 100% recall level cannot be achieved, and, by definition, this metric sets a 0 precision at recall level 1.

When computing the gain of elements **b** (or **a**), xCG takes into account the fact that **c** was already seen by the user; the gain of **b** (or **a**) is 0. The xCG score is simply the vector (1, 1, 1) where each component corresponds to the cumulated gain at a given rank. The xCG metric rightly rewards the unique ideal element.

The PRUM score is evaluated in different steps. The probabilities $P(c \in \mathcal{S}_1)$, $P(c \in \mathcal{S}_2)$, $P(c \in \mathcal{S}_3)$ that element **c** was seen after the user has consulted ranks 1, 2, and 3, respectively, are all 1. The next step consists of evaluating the probabilities $P(F_i)$ that F_i ideal elements have been seen when the user reaches position i. The user has not seen any ideal element before he starts consulting the list, and $P(F_0 = 0) = 1$. The first element in the list is the unique ideal element in the database, and we have $P(F_i = 1) = 1$ for $i > 0$. The probability that the user sees the first ideal element is computed using formula (8):

$$P(F_1 = 1 | F_0 = 0) = \frac{1 \times (1 - 0) \times 1}{1 \times 1} = 1.$$

For recall $r = 1$, formula (5) gives:

$$\text{precision} = \frac{\sum_{i=1}^{3} P(F_{i-1} = 0) P(F_i > 0 | F_{i-1} = 0)}{\sum_{i=1}^{3} P(F_{i-1} = 0)} = 1.$$

This is the expected result as the first component consulted is the only ideal element in the corpus. The user has only to consult one element of the list which leads to an ideal one.

The bad search engine. This engine returns the same elements in a different order: the list is now **a**, **b**, **c**. We expect the metrics to score this engine significantly lower than the first given that the unique ideal element is in position 3 instead of 1.

The GRP metric attributes a precision of 0.83 at recall level 1:

$$\text{precision} = \frac{2.5}{2.5 + 0.5 + (0 \times 1)/(1+1)} \approx 0.83.$$

As expected, we observe a lower score compared with the 88% achieved by the first engine, but the difference is rather small. For the GRP$_{\text{ng}}$ metric, the maximum recall is reached after the first element of the list has been consulted by the user since the unseen part of **b** or **c** is nonexistent (they were seen with **a**):

$$\begin{aligned}\text{precision} &= \frac{\frac{2}{3} \times 60 + \frac{2}{3} \times 0 + 1 \times 0}{60} \approx 0.67\\ \text{at recall} &= \frac{1 \times 1 + 1 \times 0 + 1 \times 0}{1+1+1} \approx 0.33\end{aligned}$$

Precision is the same as in the first scenario. The recall level is significantly inferior to 1, but more serious is the following inconsistency. The recall level is significantly different from the 65% score achieved by the good search engine although exactly the same elements have been seen.

The xCG score is simply the vector $(.75, .75+(1-\alpha)\times .75, .75+(1-\alpha)\times 1.75)$, where α is a value between 0 and 1 that represents how much frustration a user is willing to tolerate when accessing redundant elements; a value of 1 reflects a user who does not tolerate already viewed elements. A value $\alpha < \frac{2}{3}$ can lead to a gain at rank 3 superior to 1; when $\alpha < \frac{3}{4}$, the same can happen at rank 2. This could be fixed simply by normalizing the scores as is currently done with descendants of an ideal element. However, the procedure is not straightforward since an ancestor of an ideal element can contain other ideal elements. Another possibility is to set α to 1 so that descendants of an already consulted element are not rewarded at all. This solution was adopted in INEX 2005. This shows one problem that stems from the fact that the metric is not defined with respect to a formal and explicit user model.

We now show how PRUM solves these problems. As before, we first compute the probabilities $\mathrm{P}(c \in \mathcal{S}_1)$, $\mathrm{P}(c \in \mathcal{S}_2)$, and $\mathrm{P}(c \in \mathcal{S}_3)$ that element **c** was seen after the user consulted rank 1, 2, and 3, respectively. Following Equation (9), the probability that the user browses from **a** (respectively, **b** and **c**) to **c** is $\frac{10}{60}$ (respectively, $\frac{10}{40}$ and 1). Consequently $\mathrm{P}(c \in \mathcal{S}_1) = \frac{10}{60}$, $\mathrm{P}(c \in \mathcal{S}_2) = 1 - (1 - \frac{10}{60})(1-\frac{10}{40}) = \frac{3}{8}$ and $P(c \in \mathcal{S}_3) = 1-(1-\frac{10}{60})(1-\frac{10}{40})(1-1) = 1$. The probability that the user sees no ideal element before he starts and until position 2 is computed with formula (7). We have $\mathrm{P}(F_0 = 0) = 1$, $\mathrm{P}(F_1 = 0) = \frac{5}{6}$ and $\mathrm{P}(F_2 = 0) = \frac{5}{8}$. The probability that the user sees an ideal element knowing he has not seen one before is formula (8):

$$\mathrm{P}(F_1 = 1 | F_0 = 0) = \frac{1 \times \left(\frac{1}{6} - 0\right) \times 1}{1 \times 1} = \frac{1}{6}$$

$$\mathrm{P}(F_2 = 1 | F_1 = 0) = \frac{1 \times \left(\frac{3}{8} - \frac{1}{6}\right) \times \frac{5}{6}}{\frac{5}{6} \times \frac{5}{6}} = \frac{1}{4}$$

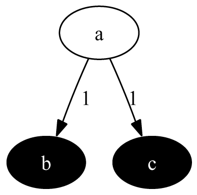

Fig. 7. A collection composed of 100 elements; only three of them are shown in the graph. There are two ideal elements (**b** and **c**). The navigation probabilities are shown next to the graph edges. In the case of XML IR, we suppose that the three elements are judged E3S3.

$$\mathrm{P}(F_3 = 1 | F_2 = 0) = \frac{1 \times \left(1 - \frac{3}{8}\right) \times \frac{5}{8}}{\frac{5}{8} \times \frac{5}{8}} = 1$$

This leads to a precision of 41% at $r = 1$:

$$\begin{aligned} \text{precision} &= \frac{\sum_{i=1}^{3} \mathrm{P}(F_{i-1} = 0) P(F_i > 0 | F_{i-1} = 0)}{\sum_{i=1}^{3} \mathrm{P}(F_{i-1} = 0)} \\ &= \frac{1 \times \frac{1}{6} + \frac{5}{6} \times \frac{1}{4} + \frac{5}{8} \times 1}{1 + \frac{5}{6} + \frac{5}{8}} = 0.41 \end{aligned}$$

As expected, the bad engine score is significantly lower than the 100% achieved by the good one. It is also slightly superior to a strict version of precision-recall (33%) because it takes into account the possibility that the user navigates to the ideal element from **a** or **b**.

4.2.2 *Example* 2. *Best Entry Point.* In this Section, we illustrate the ability of the PRUM metric to properly reward best entry points. Figure 7 represents a collection where two elements (**b** and **c**) are the only ideal units. A third element, **a**, acts as an entry point; all the users would eventually see **b** and/or **c** if the system retrieves **a**. We will compare PRUM with the other metrics when the retrieved lists is **a** (the BEP is ranked before the ideal units). We restrict **b** and **c** to be the only children of element **a**. We also suppose that **b** and **c** are fully exhaustive and specific answers to the query, that is, they score E3S3 on the INEX assessment scale. Consequently, element **a** is also E3S3 as it does not contain any irrelevant material (S3) and as it contains an exhaustive answer to the query (E3). The quantizations values of **a**, **b**, and **c** for GRP, GRP$_{\text{ng}}$, and xCG are thus 1 (see Appendix C).

The retrieved list does not contain all the elements of the database that lead the user to see the two ideal elements. For GRP, it is necessary to know the number of elements of the database. It is arbitrarily set to 100 here. As in the previous section, we are interested in the precision at a recall level of 100% which is reached here for a recall value of 2.

When the search engine returns only the best entry point (**a**), the unranked part of the list is then composed of 99 elements containing two ideal units.

For the GRP metric, the maximum achievable recall value is 3. This recall value is achieved only when the user consults the unranked list which is composed of 2 ideal and 97 irrelevant units.

$$\text{precision} = \frac{3}{3 + 0 + (97 \times 2)/(2+1)} \approx 0.044.$$

Precision is low because the user has to consult the whole unranked list (none of the ideal elements have been retrieved in the ranked part).

A recall level of 1 cannot be achieved by GRP_{ng}. The score calculated by GRP_{ng} is

$$\text{precision} = 1 \ \text{ at recall} = \frac{1}{3}.$$

Since the gain of **a** is 1, the xCG score is a vector reduced to a single component, 1. However, this is not the best achievable result since an ideal run for xCG would be composed of both elements **b** and **c**. The system achieves only a normalized gain of 0.5. From a gain perspective, the system managed to retrieve only one ideal unit.

The PRUM score is evaluated as follows. First, the probability that the user sees element **b** or **c** is 1 at rank 1. Consequently, at rank 1, the probability that the user sees 0, 1, or 2 ideal units is respectively 0, 0, and 1: As in the previous example, the user did not see any ideal element before he start consulting the list, and $\text{P}(F_0 = 0) = 1$. The probability that the user sees for the first time an ideal unit at rank 1 is computed using formula (8):

$$\begin{aligned}\text{P}(F_1 > 0|F_0 = 0) &= 1 - \prod_{x \in \{a,b\}} \left(1 - \frac{\left(\text{P}(S_x^i) - \text{P}(S_x^{i-1})\right) \text{P}\left(F_{i-1} = s \,|\neg S_x^{i-1}\right)}{\text{P}(F_{i-1} = s)}\right) \\ &= 1 - \left(1 - \frac{(1-0) \times 1}{1}\right)\left(1 - \frac{(1-0) \times 1}{1}\right) = 1.\end{aligned}$$

We can now compute PRUM (for recall $r = 2$) with formula (5):

$$\begin{aligned}\text{precision} &= \frac{\text{P}(F_0 = 0)P(F_1 > 0|F_0 = 0) + \sum_{s=0}^{1} \text{P}(F_1 = s) \times (2 - s)}{\text{P}(F_0 = 0) + \sum_{s=0}^{1} \text{P}(F_1 = s) \times (2 - s) \times \left(1 + \frac{99-(2-s)}{2-s+1}\right)}. \\ &= \frac{1+0}{1+0} = 1\end{aligned}$$

This is the expected result as **a** is a BEP for both **b** and **c**.

In the first example, PRUM is the only metric that is consistent. The two metrics GRP and GRP_{ng} suffer from the varying recall base problems mentioned in the discussion section. For xCG, the parameter α may lead to inconsistent results. The second example shows that PRUM is able to properly handle the evaluation of a run containing BEP.

5. DISCUSSION

Most metrics used to compare the performance of structured document search engines rely—sometimes implicitly—on a simplistic model of user behavior.

The user is restricted to consulting exclusively with the elements of the list returned by the engine. This user model is not adapted to recent structured IR tasks like XML, Web or video retrieval. In particular, it does not consider this user ability to navigate between elements, using the list as entry points to the information. Beyond being inadequate, this may also lead to metrics that score search engines inconsistently.

We presented in this work an extension of the probabilistic measure of precision-recall which explicitly models the navigational behavior of a user. The precision-recall with user modeling (PRUM) metric is designed for new IR paradigms like XML, Web, or video. This metric has sound theoretical foundations and is consistent with the assumptions of these new structured IR tasks.

PRUM is a generalization of commonly accepted metrics. It reduces to standard recall-precision if browsing between elements is not allowed. It is able to formally describe a whole family of user models and makes explicit the different user behavior assumptions. It allows sophisticated user behaviors even though we restricted ourselves to simple cases in the examples presented here. For instance, the browsing probability $P(x \to y)$ can be made dependent on the position of x in the search engine list, and hence on previous list elements consulted by the user. In the context of XML retrieval, it can accommodate a user who would be unwilling to browse again in a document he has already consulted in the context of a previous list element. Other models can be defined to match the findings of user experimentation.

Future work on the PRUM metric is experimental. We intend to collect data on XML IR user behavior and derive the corresponding PRUM parameter values. We will also quantify the relation between the measures given by the PRUM metric and the satisfaction of the user measured through subjective (e.g., questionnaires) and objective (ideal material found) methods.

The metric is implemented in the EvalJ project which aims at providing a set of proposed metrics for XML IR. More information can be found at this URL: `http://evalj.sourceforge.net.`

APPENDIXES

A. DETAILS OF PRUM COMPUTATION

In this section, we show how PRUM can be derived, based on the user model defined in Section 2. Appendices A.3 and A.4 show how $\mathrm{P}_r(Lui|Cs)$ can be expressed in the form

$$\frac{\overbrace{\sum_{s=0}^{r-1}\sum_{i=1}^{\mathbf{o}} \mathrm{P}_r(F_{i-1}=s)\mathrm{P}_r(F_i > s|F_{i-1}=s)}^{(a)} + \overbrace{\sum_{s=0}^{r-1}\mathrm{P}_r(F_{\mathbf{o}}=s)(r-s)}^{(b)}}{\underbrace{\sum_{s=0}^{r-1}\sum_{i=1}^{\mathbf{o}} \mathrm{P}_r(F_{i-1}=s)}_{(c)} + \underbrace{\sum_{s=0}^{r-1}\mathrm{P}_r(F_{\mathbf{o}}=s)(r-s)\times\left(1+\frac{\mathbf{u}-(|\mathfrak{I}|-s)}{|\mathfrak{I}|-s+1}\right)}_{(d)}}$$

Appendices A.1 and A.2 detail how the quantities $\mathrm{P}_r(F_i = s)$ and $\mathrm{P}_r(F_i > s|F_{i-1} = s)$ appearing in the different terms of the preceding equation can be computed.

A.1 Probability of Seeing a Given Number of Ideal Units

For different terms in Equation (5), one needs to compute the probability of the event $F_i = s$ that a user sees s ideal elements after he has consulted the i first ranks of a totally ordered list. In PRUM, this is equivalent to stating that the set of ideal elements seen by the user contains s elements:

$$|\{x|x \in \mathcal{S}_i \wedge x \in \mathfrak{I}\}| = s \equiv F_i = s.$$

Formally, the event $F_i = s$ is true if and only if there exists a subset $A \subseteq \mathfrak{I}$ of cardinality s such that all elements of A have been seen by the user and no other ideal element has been seen at rank i. $F_i = s$ is true if and only if

$$\forall x \in \mathfrak{I} \text{ we have } x \in A \Leftrightarrow x \in \mathcal{S}_i.$$

The latter event can be written:

$$\left(\bigwedge_{x \in A} x \in \mathcal{S}_i\right) \wedge \left(\bigwedge_{x \in \mathfrak{I} \setminus A} x \notin \mathcal{S}_i\right). \tag{10}$$

The different possibilities for a user to see exactly s ideal elements correspond to a disjunction over all the possible subsets A of $\mathfrak{I}$ of cardinality s:

$$F_i = s \equiv \bigvee_{\substack{A \subseteq \mathfrak{I} \\ |A|=s}} \left(\left(\bigwedge_{x \in A} x \in \mathcal{S}_i\right) \wedge \left(\bigwedge_{x \in \mathfrak{I} \setminus A} x \notin \mathcal{S}_i\right)\right).$$

The composite events (10) inside the disjunction being mutually exclusive, the probability of the disjunction can be rewritten as the sum over all the subsets $A \subseteq \mathfrak{I}$ of cardinality s.

Using hypotheses (2.1) and (2.2) that imply that events $x \in \mathcal{S}_i$ are mutually independent, we easily obtain the following expression:

$$\mathrm{P}_r(F_i = s) = \sum_{\substack{A \subseteq \mathfrak{I} \\ |A|=s}} \prod_{x \in A} \mathrm{P}_r(x \in \mathcal{S}_i) \prod_{x \in \mathfrak{I} \setminus A} \mathrm{P}_r(x \notin \mathcal{S}_i), \tag{11}$$

where the sum is over all the subsets A of X which contain s elements and where $\mathrm{P}_r(x \in \mathcal{S}_i)$ is given by (1). Equation (11) can be computed in $O(m^2)$, where m is the number of elements x for which $1 > \mathrm{P}_r(x \in \mathcal{S}_i) > 0$. Note that a normal approximation—theorem of Lindenberg [Saporta 1990]—can be used for large values of m:

$$\mathrm{P}_r(F_i = s) \approx \mathrm{P}_r(s - 0.5 < \widetilde{F}_i \leq s + 0.5),$$

where $\widetilde{F}_i$ follows a normal distribution of mean $\frac{1}{|X|}\sum_{x \in X} \mathrm{P}_r(x \in \mathcal{S}_i)$ and variance $\frac{1}{|X|}\sum_{x \in X} \mathrm{P}_r(x \in \mathcal{S}_i)(1 - \mathrm{P}_r(x \in \mathcal{S}_i))$. Using a sample of 10,000 random trials, we found experimentally that the maximum absolute difference between the normal approximation and the exact distribution was on average 0.01 for $m = 10$.

A.2 Probability of Seeing an Ideal Element for the First Time

For term (a) in Equation (5), one needs to compute the probability $P_r(F_i > s|F_{i-1} = s)$ that the i th element leads to a previously unseen ideal element, knowing that s ideal elements have already been seen by the user after he consulted the list up to rank $i-1$. This is equivalent to finding at least one ideal element at position i that was not seen at position $i-1$. Using hypothesis (2.3), we have

$$P_r(F_i > s|F_{i-1} = s) = 1 - \prod_{x \in \mathfrak{I}} (1 - P_r(x \in \mathcal{S}_i \wedge x \notin \mathcal{S}_{i-1}|F_{i-1} = s)),$$

where the right term can be decomposed using the Bayes rule and the fact that $P_r(F_{i-1} = s|x \notin \mathcal{S}_{i-1} \wedge x \in \mathcal{S}_i) = P_r(F_{i-1} = s|x \notin \mathcal{S}_{i-1})$:

$$P_r(x \in \mathcal{S}_i \wedge x \notin \mathcal{S}_{i-1}|F_{i-1} = s) = \frac{P_r(x \in \mathcal{S}_i \wedge x \notin \mathcal{S}_{i-1})P_r(F_{i-1} = s|x \notin \mathcal{S}_{i-1})}{P_r(F_{i-1} = s)}.$$

Once an element has been seen at a given rank, it remains seen for all the following ranks: formally, $x \in \mathcal{S}_{i-1} \implies x \in \mathcal{S}_i$. This implies that $x \in \mathcal{S}_i \wedge x \in \mathcal{S}_{i-1} \equiv x \in \mathcal{S}_{i-1}$. As

$$P_r(x \in \mathcal{S}_i) = P_r(x \in \mathcal{S}_i \wedge x \notin \mathcal{S}_{i-1}) + P_r(x \in \mathcal{S}_i \wedge x \in \mathcal{S}_{i-1}),$$

we have $P_r(x \in \mathcal{S}_i \wedge x \notin \mathcal{S}_{i-1}) = P_r(x \in \mathcal{S}_i) - P_r(x \in \mathcal{S}_i \wedge x \in \mathcal{S}_{i-1}) = P_r(x \in \mathcal{S}_i) - P_r(x \in \mathcal{S}_{i-1})$. Then,

$$\begin{aligned} &P_r(F_i > s|F_{i-1} = s) \\ &\quad = 1 - \prod_{x \in \mathfrak{I}} \left(1 - \frac{(P_r(x \in \mathcal{S}_i) - P_r(x \in \mathcal{S}_{i-1}))P_r\left(F_{i-1} = s\,|x \notin \mathcal{S}_{i-1}\right)}{P_r(F_{i-1} = s)}\right), \end{aligned} \quad (12)$$

where $P_r\left(F_{i-1} = s\,|x \notin \mathcal{S}_{i-1}\right)$ can be evaluated starting with Equation (11) and using the Bayes rule:

$$\begin{aligned} P_r\left(F_{i-1} = s\,|x \notin \mathcal{S}_{i-1}\right) &= \frac{1}{P_r(x \notin \mathcal{S}_{i-1})} \sum_{\substack{A \subseteq \mathfrak{I}\setminus\{x\} \\ |A|=s}} \prod_{y \in A} P_r(y \in \mathcal{S}_{i-1}) \prod_{y \in \mathfrak{I}\setminus A} P_r(y \notin \mathcal{S}_{i-1}) \\ &= \sum_{\substack{A \subseteq \mathfrak{I}\setminus\{x\} \\ |A|=s}} \prod_{y \in A} P_r(y \in \mathcal{S}_{i-1}) \prod_{\substack{y \in \mathfrak{I}\setminus A \\ y \neq x}} P_r(y \notin \mathcal{S}_{i-1}) \end{aligned}$$

This is formally identical to Equation (11) in Appendix A.1 and is equivalent to computing the probability $P_r(F_i = s)$ of finding s ideal elements until position i, considering that element x does not belong to the database anymore.

Using this result we are now able to compute term (a) in Equation (5).

A.3 Probability of Being Consulted

We will now show how the denominator of Equation (4) can be expanded into the one of Equation (5). Consider an element at random in the list: what is the probability $P_r(Cs)$ that this element was consulted by the user while searching for r ideal elements?

We separate the cases where the element is in the ranked and in the unranked parts of the list:

$$\mathrm{P}_r(Cs) = \mathrm{P}_r(Cs \wedge Rk) + \mathrm{P}_r(Cs \wedge \neg Rk).$$

$\mathrm{P}_r(Cs \wedge Rk)$ will be computed with our user model. For $\mathrm{P}_r(Cs \wedge \neg Rk)$, we have to switch back to the classic (nonnavigating) user model as explained in Section 2.

Let us define the event AtRank_i that corresponds to "The element under consideration is at position i". The set $\{ \text{AtRank}_i \}_{i=1..\mathbf{o}+\mathbf{u}}$ forms a partition of the probabilistic universe. As there is no reason to prefer an element to another, it follows a uniform distribution with probability $\mathrm{P}_r(\text{AtRank}_i) = \frac{1}{\mathbf{o}+\mathbf{u}}$. We then have.

$$\mathrm{P}_r(Cs \wedge Rk) \;=\; \sum_{i=1}^{\mathbf{o}} \mathrm{P}_r(\text{AtRank}_i)\mathrm{P}_r(Cs|\ \text{AtRank}_i). \tag{13}$$

An element at position i is consulted if and only if the user did not find previously (at rank $i-1$) the r elements he requires. This implies the equivalence $\mathrm{P}_r(Cs|\ \text{AtRank}_i) = \mathrm{P}_r(F_{i-1} < r)$. Since $\mathrm{P}(\text{AtRank}_i) = \frac{1}{\mathbf{o}+\mathbf{u}}$, we can express $\mathrm{P}_r(Cs, Rk)$ in terms of the probability $\mathrm{P}_r(F_{i-1} = s)$ described in Appendix A.1:

$$\mathrm{P}_r(Cs \wedge Rk) \;=\; \frac{1}{\mathbf{o}+\mathbf{u}} \sum_{i=1}^{\mathbf{o}} \sum_{s<r} \mathrm{P}_r(F_{i-1} = s). \tag{14}$$

We now deal with the nonranked part. Cooper [1968] shows how to compute the expected number of elements a user has to consult if he wants to find a given number of ideal and unseen elements in an unranked set where the ratio of ideal to nonideal elements is known. As the ratio depends on the value of $F_{\mathbf{o}}$, which is the number of ideal elements the user has found in the ranked list, we have to consider all the possible values of $F_{\mathbf{o}}$. We decompose $\mathrm{P}_r(Cs, \neg Rk)$ as follows:

$$\mathrm{P}_r(Cs \wedge \neg Rk) \;=\; \sum_{s=0}^{r-1} \mathrm{P}_r(F_{\mathbf{o}} = s)\underbrace{\mathrm{P}_r(Cs \wedge \neg Rk | F_{\mathbf{o}} = s)}_{(*)}. \tag{15}$$

The term $(*)$ can be decomposed into

$$\sum_{i=1}^{\mathbf{u}} \underbrace{\mathrm{P}_r(Cs \wedge \neg Rk | F_{\mathbf{o}} = s \wedge N = i + \mathbf{o})}_{(**)} \mathrm{P}_r(N = i + \mathbf{o} | F_{\mathbf{o}} = s),$$

where N is the number of elements consulted by the user in the list. The term (**) in this equation corresponds to the probability that an element is in the nonranked part and consulted by the user, knowing that he has consulted $i + \mathbf{o}$ elements. Among these $i + \mathbf{o}$ elements, only i are in the nonranked part.

Consequently, term (**) equals $\frac{i}{\mathbf{o}+\mathbf{u}}$ and Equation (15) can be rewritten as:

$$\begin{aligned}
\mathrm{P}_r(Cs \wedge \neg Rk) &= \sum_{s=0}^{r-1} \mathrm{P}_r(F_{\mathbf{o}} = s) \sum_{i=1}^{\mathbf{u}} \frac{i}{\mathbf{o}+\mathbf{u}} \mathrm{P}_r(N - \mathbf{o} = i | F_{\mathbf{o}} = s) \\
&= \sum_{s=0}^{r-1} \mathrm{P}_r(F_{\mathbf{o}} = s) \frac{1}{\mathbf{o}+\mathbf{u}} \mathbb{E}[N - \mathbf{o} | F_{\mathbf{o}} = s, R = r] \\
&= \sum_{s=0}^{r-1} \mathrm{P}_r(F_{\mathbf{o}} = s) \frac{1}{\mathbf{o}+\mathbf{u}} (r-s) \times \left(1 + \underbrace{\frac{\mathbf{u} - (t-s)}{t-s+1}}_{a} \right), \qquad (16)
\end{aligned}$$

where $\mathbb{E}$ is the expectation. The proof of the last line can be found in Cooper [1968] and is intuitive: (a) is the expected number of nonideal elements which have to be consulted before finding an ideal one. A user wanting $r - s$ ideal elements will thus have to consult $(r - s) \times (a)$ irrelevant elements and $r - s$ ideal ones.

Equations (14) and (16) correspond, respectively, to the terms (c) and (d) in Equation (5). Both terms can be computed using the expression for $\mathrm{P}_r(F_i = s)$ obtained in Appendix A.1.

A.4 Probability of an Element Being Ideal and Consulted

To transform the numerator $\mathrm{P}_r(Lui \wedge Cs)$ of Equation (4) into the numerator of Equation (5), we again separate between ranked and unranked lists:

$$\mathrm{P}_r(Cs \wedge Lui) = \mathrm{P}_r(Cs \wedge Lui \wedge Rk) + \mathrm{P}_r(Cs \wedge Lui \wedge \neg Rk).$$

As before, we will decompose the event $Cs \wedge Lui \wedge Rk$ using the set of disjoint events AtRank_i. We use the following observations: 1) the user consults the ith element if and only if he has found previously strictly less than r ideal elements, 2) the ith list element leads to a newly-ideal element (Lui) if and only if the number of ideal elements seen by the user increases after he has consulted the ith element, that is, if $F_i > F_{i-1}$. This leads to the result for the ranked part:

$$\begin{aligned}
\mathrm{P}_r(Lui \wedge Cs \wedge Rk) &= \sum_{i=1}^{\mathbf{o}} \mathrm{P}_r(F_{i-1} < r \wedge F_i > F_{i-1} | \mathrm{AtRank}_i) \mathrm{P}_r(\mathrm{AtRank}_i) \\
&= \frac{1}{\mathbf{o}+\mathbf{u}} \sum_{i=1}^{\mathbf{o}} \mathrm{P}_r(F_{i-1} < r \wedge F_i > F_{i-1} | \mathrm{AtRank}_i) \\
&= \frac{1}{\mathbf{o}+\mathbf{u}} \sum_{i=1}^{\mathbf{o}} \sum_{s=0}^{r-1} \mathrm{P}_r(F_{i-1} = s) \mathrm{P}_r(F_i > s | F_{i-1} = s) \qquad (17)
\end{aligned}$$

For the nonranked part, we consider all the possible numbers of ideal elements seen by the user after he has consulted the ranked part ($F_{\mathbf{o}}$).

$$\mathrm{P}_r(Lui \wedge Cs \wedge \neg Rk) = \sum_{s=0}^{r-1} \mathrm{P}_r(F_{\mathbf{o}} = s) \mathrm{P}_r(Rel \wedge Cs \wedge \neg Rk | F_{\mathbf{o}} = s).$$

We use again the classic nonbrowsing user model. Since there is no navigation, and since s ideal elements were found in the ranked list, the user needs to find $r - s$ ideal elements in the unranked list. The probability that he randomly picks an element which is ideal, consulted, and not in the ranked part is then:

$$\mathrm{P}_r(Lui \wedge Cs \wedge \neg Rk | F_{\mathbf{o}} = s) = \frac{r-s}{\mathbf{o}+\mathbf{u}}. \tag{18}$$

The final result is the written:

$$\mathrm{P}_r(Lui \wedge Cs \wedge \neg Rk) = \sum_{s=0}^{r-1} \mathrm{P}_r(F_{\mathbf{o}} = s) \frac{r-s}{\mathbf{o}+\mathbf{u}}.$$

Equations (17) and (18) show the equivalence between the numerators of Equations (4) and (5). As before, the different terms can be computed using directly the results from Appendices A.1 and A.2.

B. PRUM AND STANDARD PRECISION-RECALL

Precision-recall is a particular case of the PRUM metric with no navigation, that is, when the standard user model is used. The absence of navigation implies that $P(x \to y)$ is 1 if and only if $x = y$ and 0 otherwise. The user sees the element if and only if it is in the part of the list he consults, that is, $x \in \mathcal{S}_i$ if and only if $\times$ is within the i first elements consulted by the user, that, is $x \in \mathcal{L}_i$. We can then write that the user has found s ideal elements if and only if he has examined exactly s ideal elements:

$$\mathrm{P}_r(F_i = s) = 1 \Leftrightarrow |\mathcal{L}_i \cap \mathfrak{I}| = s,$$

where $\mathcal{L}_i \cap \mathfrak{I}$ is the set of ideal elements the user has seen directly in the returned list. In the following, we denote e_i the number of ideal elements seen when the user has examined all elements from position 1 to i; e_i verifies

$$\mathrm{P}_r(F_i = e_i) = 1$$

We can easily state that $e_i \geq e_j$ for $i > j$. We also denote $\mathbf{l_r}$ the minimum number of ranks the user has to consult before seeing r ideal elements:

$$\mathbf{l_r} = \min\{i | P(F_i = r) = 1\}.$$

.

We specialize for the classic user model the four Equations (14), (16), (17), and (18) composing the PRUM formula in Equation (5):

Equation (14). The equation can easily be rewritten

$$\sum_{i=0}^{\mathbf{o}-1} \mathrm{P}_r(F_i < r).$$

If the user wants to see s ideal elements, then we have to consider two cases:

(1) The ranked part contains at least r ideal units ($e_{\mathbf{o}} \geq r$). The first rank for which $\mathrm{P}(F_i < r) = 0$ is by definition $\mathbf{l_r}$, the rank at which the user finds its

rth ideal element. For all the ranks between 0 and $\mathbf{l_r} - 1$, the probability $\mathrm{P}(F_i < r)$ equals to 1. The sum is thus equal to $\mathbf{l_r}$.

(2) The ranked part contains strictly less than r ideal elements, and the user has to consult all the elements ($\mathbf{o}$) in the ranked part to be satisfied: $\mathrm{P}(F_i < r) = 1$ for $0 \leq i \leq \mathbf{o} - 1$ and the sum equals $\mathbf{o}$.

Formally, the two cases are summarized as:

$$\sum_{i=1}^{\mathbf{o}} \sum_{s=0}^{r-1} \mathrm{P}_r(F_{i-1} = s) = \begin{cases} \mathbf{l_r} & \text{if } e_{\mathbf{o}} \geq r \\ \mathbf{o} & \text{otherwise.} \end{cases}$$

Equation (16). As the user has found $e_{\mathbf{o}}$ ideal elements at the end of the ranked list, he doesn't have to consult the unranked list if his information need is satisfied: We have $\mathrm{P}_r(Cs, \neg Rk) = 0$ if $e_{\mathbf{o}} \geq r$. Otherwise, using Equation (16), we equal $\mathrm{P}_r(Cs, \neg Rk)$ to

$$\sum_{s=0}^{r-1} \mathrm{P}_r(F_{\mathbf{o}} = s)(r - s) \left(1 + \frac{\mathbf{u} - (t - s)}{t - s + 1}\right) = (r - e_{\mathbf{o}}) \left(1 + \frac{\mathbf{u} - (t - e_{\mathbf{o}})}{t - e_{\mathbf{o}} + 1}\right),$$

where we used the fact that $\mathrm{P}_r(F_{\mathbf{o}} = s) = 1 \Leftrightarrow s = e_{\mathbf{o}}$.

Equation (17). We need to evaluate $\sum_{i=1}^{\mathbf{o}} \sum_{s=0}^{r-1} \mathrm{P}_r(F_{i-1} = s) \mathrm{P}_r(F_i > s | F_{i-1} = s)$ or equivalently

$$\sum_{i=1}^{\mathbf{o}} \sum_{s=0}^{r-1} \mathrm{P}_r(F_i > s \wedge F_{i-1} = s).$$

We know that $F_{i-1} = s$ is true if and only if $s = e_{i-1}$, hence we can restrict the second summation—the number of ideal elements within the $\mathbf{o} - 1$ first elements of the ranked part cannot exceed $e_{\mathbf{o}-1}$:

$$\sum_{i=1}^{\mathbf{o}} \sum_{s=0}^{\min(r-1, e_{\mathbf{o}-1})} \mathrm{P}_r(F_i > s \wedge F_{i-1} = s)$$

as $\mathrm{P}_r(F_{i-1} = s) = 0$ if $s > e_{\mathbf{o}-1}$ and $i \leq \mathbf{o}$ (this is a consequence of the fact that $e_{i-1} \leq e_{o-1}$). Let us consider the event $F_{i-1} = s$ first. It is true only if $s = e_{i-1}$ which, in turn, can happen only if $e_{i-1} \leq r - 1$ in the summation (the other requirement is that $e_{i-1} \leq e_{o-1}$ which is always true). The last condition is equivalent to $i \leq \mathbf{l}_r$ as $\mathbf{l_r}$ is the first rank for which $e_i = r$, and hence the previous equation can be rewritten:

$$\sum_{i=1}^{\min(\mathbf{l}_r, \mathbf{o})} \mathrm{P}_r(F_i > e_{i-1}).$$

If the user consults the list up to the rank $\mathbf{l}_r$, then he consults r ideal elements, that is, $\mathrm{P}(F_i > e_{i-1})$ equals 1 exactly r times and 0 in the other cases. Otherwise, he sees only $e_{\mathbf{o}}$ ideal elements. Summarizing, we have:

$$\sum_{i=1}^{\mathbf{o}} \sum_{s=0}^{r-1} \mathrm{P}_r(F_{i-1} = s) \mathrm{P}_r(F_i > s | F_{i-1} = s) = \begin{cases} r & \text{if } e_{\mathbf{o}} \geq r \\ e_{\mathbf{o}} & \text{otherwise.} \end{cases}$$

Equation (18). If the ranked list contains less than r ideal elements, the user will require $r - e_{\mathbf{o}}$ ideal elements from the nonranked part:

$$\sum_{s=0}^{r-1} \mathrm{P}_r(F_{\mathbf{o}} = s)(r - s) = \begin{cases} 0 & \text{if } e_{\mathbf{o}} \geq r \\ r - e_{\mathbf{o}} & \text{otherwise} \end{cases}$$

as $\mathrm{P}(F_{\mathbf{o}} = s) = 1 \Leftrightarrow s = e_{\mathbf{o}}$.

Conclusion. Putting all together, we have

$$\mathrm{P}_r(Lui|Cs) = \begin{cases} \frac{r}{\mathbf{l_r}} & \text{if } e_{\mathbf{o}} \geq r \\ \frac{r}{r+(o-e_{\mathbf{o}})+(r-e_{\mathbf{o}})\left(\frac{\mathbf{u}-(t-e_{\mathbf{o}})}{t-e_{\mathbf{o}}+1}\right)} & \text{otherwise.} \end{cases}$$

When the list is composed of a totally ordered set of elements followed by a totally unordered set of elements and for $r \in \mathbb{N}$, $\mathrm{P}_r(Lui|Cs)$ is identical to Raghavan's formula (19) of Section C.1. To see why, we consider the same two cases:

—if $e_{\mathbf{o}} \geq r$, $r + j$ (number of elements consulted by the user searching for r elements) corresponds to the definition of $\mathbf{l_r}$;

—if $e_{\mathbf{o}} < r$, then j (number of non ideal elements consulted by the user the position before he stops) can be identified to $\mathbf{o} - e_{\mathbf{o}}$ (number of elements in the ranked part); the number i (resp. k) of nonideal (respectively ideal) elements consulted by the user in the last rank can be identified to $\mathbf{u} - (t - e_{\mathbf{o}})$ (respectively $t - e_{\mathbf{o}}$).

C. FORMAL DEFINITIONS OF OTHER XML IR METRICS

C.1 GRP

The formal definition of GRP is

$$\mathrm{GRP} = \frac{r}{r + j + (i \times s)/(k + 1)}, \tag{19}$$

where r is the number of ideal elements the user requires; j the number of nonideal elements consulted by the user one rank[6] before he stops; i (respectively, k) is the number of nonideal (respectively, ideal) elements at the last rank consulted by the user; s is the number of ideal elements the user wants to see at the last rank. To compute GRP in the context of INEX experiments, we need to map the ExSy assessment scale defined in Section 4.1 to a measure of element relevance. The generalized quantization method of INEX, for example, rewards some near misses that are either ancestors or descendants of an ideal

[6]For Raghavan, a rank can contain more than one document/element.

element.

$$q_{\text{generalized}}(x) = \begin{cases} 1 & \text{if } a_x = \text{E3S3} \\ 0.75 & \text{if } a_x \in \{\text{E2S3, E3S2, E3S1}\} \\ 0.5 & \text{if } a_x \in \{\text{E1S3, E2S2, E2S1}\}, \\ 0.25 & \text{if } a_x \in \{\text{E1S1, E1S2}\} \\ 0 & \text{if } a_x = \text{E0S0} \end{cases}$$

where a_x is the assessment of element x. As elements relevance is potentially between 0 and 1, we need to precisely define how the different values defined by Raghavan are computed with GRP. Let f_i be the number of ideal elements in the list up to rank i (included). It is defined as

$$f_k = \sum_{x \text{ before or at rank } k} q(x),$$

where q is a given quantization. Let $\mathbf{l_r}$ be the rank where the user finds its rth ideal element ($f_{\mathbf{l_r}-1} < r$ and $f_{\mathbf{l_r}} \geq r$); Then i, j and k are defined as:

$$\begin{aligned} i &= \sum_{x \text{ at rank } \mathbf{l_r}} 1 - q(x) \\ j &= \sum_{x \text{ before rank } \mathbf{l_r}} 1 - q(x) \\ k &= \sum_{x \text{ at rank } \mathbf{l_r}} q(x) \end{aligned}$$

C.2 GRP$_{\text{ng}}$

GRP$_{\text{ng}}$ takes into account overlap and is defined as:

$$\text{recall} = \frac{\sum_{i=1}^{k} q(exh(x_i)) \times \frac{\text{unseen length}(x_i)}{\text{length}(x_i)}}{\sum q(exh(x_i))}$$

$$\text{precision} = \frac{\sum_i q(spec(x_i)) \times \text{ unseen length}(x_i)}{\sum_i \text{ unseen length}(x_i)},$$

where *exh* and *spec*, respectively, returns the exhaustivity and the specificity of an element and where x_i is the ith element of the list returned by the IR system. The function q is used to map the 0–3 assessment scale to the unit interval:

$$q(x) = \begin{cases} 1 & \text{if } x = 3 \\ 2/3 & \text{if } x = 2 \\ 1/3 & \text{if } x = 1 \\ 0 & \text{if } x = 0, \end{cases}$$

where x is either an exhaustivity or a specificity value.

C.3 xCG

The definition of xCG at rank n is as follows:

$$\mathrm{xCG}(n) = \sum_{k=1}^{n} rv(x_k),$$

where x_k is the kth element of the list consulted by the user. The relevance value, rv, is defined by different formulas, depending on whether ancestors and/or descendants of the element are ranked higher in the retrieved list:

$$rv(x) = \begin{cases} (1-\alpha)q(x) & \text{if an ancestor of } x \text{ has been seen} \\ \alpha \sum\limits_{y \text{ child of } x} \dfrac{rv(y) \times \mathrm{size}(y)}{\mathrm{size}(x)} + (1-\alpha)q(x) & \text{if only a descendant of has been seen} \\ q(x) & \text{otherwise (no overlap)} \end{cases}$$

where $q(x)$ is a quantization of the assessment of element x, and α is a coefficient between 0 and 1 which is set manually. α represents how much frustration a user is willing to tolerate when accessing redundant components or component-parts. In this article, we used the generalized quantization defined in Appendix C.1. In order to prevent rewarding a set of descendant elements more than their ideal ancestor, the following constraint is used for any ideal element y

$$\sum_{x \in S} rv(x) \le rv(y),$$

where S is the set of elements both retrieved and descendant of y.

When α is set to 1, a descendant of an element ranked higher in the list is not rewarded at all. When set to any other value, overlap is rewarded with an amount controlled by α.

ACKNOWLEDGMENT

Many thanks to M. Lehtonen for his useful comments.

REFERENCES

Baeza-Yates, R. and Ribeiro-Neto, B. 1999. *Modern Information Retrieval*. Addison Wesley, New York, NY.

Cleverdon, C. 1967. The cranfield tests on index language devices. In *Proceedings Aslib*. vol. 19, 173–192.

Cooper, W. S. 1968. Expected search length: A single measure of retrieval effectiveness based on the weak ordering action of retrieval systems. *Amer. Documentat. 19*, 30–41.

Cooper, W. S. 1973. On selecting a measure of retrieval effectiveness. part 1. *J. Amer. Soci. Inform. Sci. 24*, 87–100.

Dunlop, M. D. 1997. Time, relevance and interaction modeling for information retrieval. In *Proceedings of the 20th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval*. ACM Press, 206–213.

Fuhr, N., Lalmas, M., and Malik, S., eds. 2003. *INEX Proceedings*.

Gövert, N., Kazai, G., Fuhr, N., and Lalmas, M. 2003. Evaluating the effectiveness of content-oriented XML retrieval. Tech. rep., Computer Science 6, University of Dortmund.

Heckerman, D. and Breese, J. S. 1994. A new look at causal independence. In *Proceedings of the 10th Annual Conference on Uncertainty in Artificial Intelligence INEX Proceedings. (UAI'94)*. Morgan Kaufmann Publishers, San Francisco, CA, 286–292.

KAZAI, G. 2003. Report on the INEX 2003 metrics group. *INEX Proceedings*. 184–190.

KAZAI, G. AND LALMAS, M. 2005. Notes on what to measure in inex. In *Proceedings of the INEX Workshop on Element Retrieval Methodology*, A. Trotman, M. Lalmas, and N. Fuhr, eds. *INEX Proceedings*. University of Otago, Univerisity of Glasgow. *INEX Proceedings*.

KAZAI, G., LALMAS, M., AND PIWOWARSKI, B. 2003. Inex guidelines for topic development.

KAZAI, G., LALMAS, M., AND VRIES, A. P. 2004. The overlap problem in content-oriented XML retrieval evaluation. In *Proceedings of the 27th Annual International Conference on Research and Development in Information Retrieval*. Sheffield, UK. ACM Press, 72–79.

KEKÄLÄINEN, J. AND JÄRVELIN, K. 2002. Using graded relevance assessments in IR evaluation. *J. Ameri. Soc. Inform. Sci. 53*, 13, 1120–1129.

LALMAS, M. AND MOUTOGIANNI, E. 2000. A dempster-shafer indexing for the focussed retrieval of a hierarchically structured document space: Implementation and experiments on a web museum collection. In *6th RIAO Conference, Content-Based Multimedia Information Access*. Paris, France.

LAWRENCE, P., BRIN, S., MOTWANI, R., AND WINOGRAD, T. 1998. The pagerank citation ranking: Bringing order to the Web. Tech. rep., Stanford Digital Library Technologies Project.

MEADOW, C. T., KRAFT, D. H., AND BOYCE, B. R. 1999. *Text Information Retrieval Systems*. Academic Press, Orlando, FL.

PIWOWARSKI, B. AND GALLINARI, P. 2003. Expected ratio of relevant units: A measure for structured information retrieval. *INEX Proceedings.*

QUINTANA, Y., KAMEL, M., AND MCGEACHY, R. 1993. Formal methods for evaluating information retrieval in hypertext systems. In *Proceedings of the 11th Annual International Conference on Systems Documentation*. Kitchener-Waterloo, Ontario, Canada. ACM Press, 259–272.

RAGHAVAN, V. V., JUNG, G. S., AND BOLLMANN, P. 1989. A critical investigation of recall and precision as measures of retrieval system performance. *ACM Trans. Inform. Syst. 7*, 3, 205–229.

SAPORTA, G. 1990. *Probabilités, analyse des données et statistique*. Editions Technip, Paris, France.

VAN RIJSBERGEN, C. J. 1979. *Information Retrieval*. Butterworths.

VOORHEES, E. M. 2003. Common evaluation measures. *The 12th Text Retrieval Conference (TREC'03)*. Number SP 500-255. NIST, 1–13.

VRIES, A., KAZAI, G., AND LALMAS, M. 2004. Tolerance to irrelevance: A user-effort oriented evaluation of retrieval systems without predefined retrieval unit. In *Proceedings of RIAO (Recherche d'Information Assistée par Ordinateur (Computer Assisted Information Retrieval))*. Avignon, France.

Received July 2005; revised January 2006 and May 2006; accepted July 2006