Notes on what to measure in INEX

Gabriella Kazai
Queen Mary University of London
Mile End Road
London, UK
gabs@dcs.qmul.ac.uk

Mounia Lalmas
Queen Mary University of London
Mile End Road
London, UK
mounia@dcs.qmul.ac.uk

ABSTRACT

This paper looks at a number of issues regarding the evaluation of XML retrieval. It aims to identify what the requirements on a measure of XML retrieval effectiveness are and how the actual evaluation methodology and aspects such as the relevance dimensions and the assessment procedure affect the evaluation. We examine various current and proposed metrics, how they fit the requirements and aim to give an explanation of what exactly they measure. A question we are attempting to address is: "Is there a single good measure of retrieval effectiveness for XML retrieval?".

Categories and Subject Descriptors

H.3.3 [**Information Storage and Retrieval**]: Information Search and Retrieval

General Terms

Performance, Measurement

Keywords

XML retrieval, INEX, evaluation metrics

1. INTRODUCTION

Since its launch in 2002, INEX (INitiative for the Evaluation of XML Retrieval) has been challenged by the issue of how to measure an XML information retrieval (IR) system's effectiveness. Due to the fact that most underlying assumptions that traditional IR metrics are based upon no longer hold in an XML IR setting [6], INEX has been investigating various adaptations of established measures as well as newly proposed metrics.

Currently there are five metrics under consideration to be used as the official metric of INEX 2005. One issue with having a range of available metrics is that unless we are clear about what exactly they measure, their incorrect use can lead to confusion regarding the result of the evaluation. Although pair-wise comparisons of some of the metrics now exist in the literature [10, 17], we are still largely in the dark as to how these different measures relate to each other or how they differ from each other, or, in fact, how well they suit the evaluation task.

Copyright is held by the author.
INEX Workshop on Element Retrieval Methodology, Glasgow, UK, July 30, 2005

In this paper, we look at various issues regarding the evaluation: what should we expect from a measure, how do the relevance dimensions and the assessment procedure affect the evaluation and what the current metrics measure.

2. WHAT TO MEASURE

The main criterion of any evaluation measure is that it should be able to rank systems according to how well they satisfy a user's information need, given a retrieval task and a model of user behaviour.

2.1 Retrieval task

In INEX, the retrieval task is given as the ad-hoc retrieval of XML documents. As in traditional IR, the INEX task of ad-hoc retrieval is considered as a simulation of how a library might be used: a static set of documents being searched using a new set of topics. However, the similarity ends there.

In traditional IR, the library consists of documents, representing well-defined units of retrieval, where the relevance of one document to the query is (considered) independent from the relevance of other documents to the query. The user's information need is typically expressed in the form of a natural language statement or simply as a set of keywords. Given this, the task of an IR search engine is to return to the user, in response to his/her query, as many relevant documents and as few irrelevant documents as possible. The output is usually presented to the user as a ranked list of documents, ordered by presumed relevance to the query.

Established measures, such as recall and precision graphs, provide suitable and intuitive mechanisms for evaluating the effectiveness of IR systems based on the above retrieval task and model of user interaction. The atomic retrieval unit of a document and the binary relevance assumption allows for the simple counting of the number of relevant and the number of retrieved documents, which forms the basis of recall/precision measures. The ranking is considered by taking counts at various recall levels.

In an XML IR setting, the library consists of XML documents composed of different granularity nested XML elements, each of which represents a valid unit of retrieval, where the relevance of one component may be dependent on the relevance of other structurally related components. Furthermore, the user's query may contain structural constraints in addition to the typical content conditions. These structural constraints may then be interpreted by an XML

IR system as strict conditions that must be met by relevant elements or as vague conditions that can be considered only as hints or clues as to where relevant information may be found. The decision really comes down to the question of how much we trust users' understanding of the searched collection's structure as well as their abilities in expressing complex queries.

The general task of an XML IR engine has been defined in INEX as the task of returning, instead of whole documents, those document components (XML elements) that are most specific and exhaustive to the user's information need [15, 14]. This general task definition is then extended for the various CAS sub-tasks to take into account that user satisfaction is also dependent on the structural conditions being met (strictly or vaguely). To simplify our study, for the rest of the paper we concentrate on the CO tasks.

An issue with the above general task definition is that it leaves the concept of "most specific and exhaustive" somewhat unspecified (or rather specified only within the quantisation functions, which currently do not form part of the task description). In addition, a common misinterpretation of "most specific and exhaustive" is to equate it to "highly specific and highly exhaustive" (i.e. $(e, s) = (3, 3)$). However, the term "most" is understood to refer to the highest available combined exhaustivity and specificity score of nodes in a given XML tree. For example, it may be that amongst all possible retrievable components in an XML document (article), even the most exhaustive node is only marginally exhaustive ($e = 1$) if the topic of request is only mentioned. Similarly, it may well be that even the most specific node contains some irrelevant information (e.g. $s = 2$). When the combination of the two dimensions is considered, additional criteria is required to decide if the most specific and exhaustive elements of a tree should be, for example, two $(e, s) = (2, 3)$ paragraph elements or their container $(e, s) = (3, 2)$ section element.

The output of XML IR systems, up until the time of writing this, has been assumed to be a ranked list of XML elements, ordered by their presumed relevance to the query. Other forms of non-linear result presentations, e.g. where related results are clustered based on structural relationships, have so far been ignored due to the added complexity in their evaluation. The INEX 2005 task guidelines [14], developed in the meantime, provide a welcomed step in this direction, although - in the opinion of the first author - they require further development (more on this in Section 2.3).

2.2 User behaviour

The definition of the general ad-hoc retrieval task in the previous section is still rather vague, and one that requires further clarification. For example, what exactly is meant by returning XML elements to users? Will users have access to the full text of a returned element and its sub-nodes? What about access to the element's context? Will users need to browse in order to access related components or will the system show result elements, for example, as highlighted text fragments within their larger context element? What can be assumed of users' interaction with the system? All these factors affect the assumed user model which then impacts on the evaluation.

A user of a traditional IR system is typically associated with a simple model for interacting with the system. He/she is assumed to examine the returned ranked list in a linear fashion, moving from the top of the list down, either until the end of the list is reached, or until the point where his/her information need has been satisfied or where the user gives up. Each examined document is assumed to require approximately the same amount of effort from the user.

The user of an XML IR system is currently assumed to follow the same routine and work through the returned ranked list from top to bottom, with similar stopping options. The required effort to consult a result element, however, can no longer be assumed to be equal, but should rather be given as some function of element size or required reading time.

When users of an XML IR system access a result element, they may then have access, in one form or another (e.g. browsing or scrolling), to the element's structurally related nodes and/or context (where, depending on the user interface, the cost of this access may differ in different situations). This motivates the need to consider so-called near-misses, elements from where users can access desired relevant content, within the evaluation. For example, a section containing the sought-after relevant paragraph, a list item within the paragraph, or a neighbouring paragraph or section may all be considered as near-misses. A near-miss may itself be relevant or irrelevant to the user's query. Assuming that such near-misses may be useful for a user, as it gives him/her access to otherwise lost relevant information, the idea is then to allow systems to pick up partial scores for finding such elements.

In addition, due to the possible overlap of result elements (e.g. returned nested elements), it is argued that a further assumption is needed in INEX, according to which redundant relevant fragments are to be considered of no further value to the user. For example, once seen, a relevant paragraph may be of no interest to the user if it is again returned as part of its container section. The need for making this behaviour an explicit assumption has only been highlighted recently in [3, 10]. In [10], it was shown that unless an evaluation metric that explicitly addresses this issue is employed, unfair advantage can be gained by systems that exploit this phenomena of the INEX recall-base over systems that actually put effort into not to inundate users with such redundancy. This process of deliberately returning overlapping elements to increase effectiveness results has since been popularly named as "milking" and has been the centre of some debate in INEX.

Most of us agree that returning overlapping results contradicts the intuition about the retrieval task, which aims to decrease the user effort required in finding relevant information, and that it can lead to user disorientation when such related redundant components are dotted around at different ranks in the output list. This was also indicated in the experiments conducted by the INEX 2004 interactive track [22]. However, we may equally argue that overlap - from a system evaluation point of view - should not be seen as an issue since systems can be assumed to be able to deal with it when presenting their results to users. For example, systems may remove overlapping nodes via some filtering strategy or cluster them together, and so on. Therefore, overlap should be allowed in result lists when a system-oriented evaluation is applied. This said, a crucial (implicit) assumption of this argument is that overlap, while allowed, should not represent a potential gain factor to be exploited. This means that systems should not be penalised for not retrieving overlapping nodes! For example, a system that retrieves all relevant nodes on a path (e.g. `article[1]`, `bdy[1]`, `sec[6]` and `p[1]` in Figure 1), ranking the highest scoring first (e.g. `sec[6]`), should not be ranked better by the evaluation than another system that only returns the highest scoring node from the same path (e.g. `sec[6]`). In conclusion, *given a suitable metric*, systems would be free to follow a retrieval strategy based on "milking", but such a strategy would not present an advantage, but may in fact prove unbeneficial as it may result in the output list being filled with elements of no further value while pushing other non-overlapping relevant nodes down the list.

As a result, if returning overlapping results does not lead to a sensible retrieval strategy, then systems will be forced to make decisions as to which element(s) to retrieve from an arbitrary tree of XML elements, which is arguably the presumed aim of XML IR. As mentioned before, the retrieval task implicitly relies on a set of user preferences, modeled within the quantisation functions. These preferences dictate which elements systems should return to the user from a given XML tree. For example, the generalised quantisation function describes a user who would prefer more exhaustive components despite the additional effort needed to be spent on locating the relevant information within. Based on these set of preferences, a system would need to locate those elements in an XML tree that are more exhaustive than any of their structurally related nodes, where from two nodes with the same exhaustivity the more specific one is preferred.

2.3 Matching user types to tasks

A problem with the current setup of the general INEX retrieval task and the various user models represented by the quantisation functions is that in the first instance the task is not explicitly motivated by a given user model and secondly that different systems may have been tuned to different user models, but were all evaluated under the general CO task umbrella and using all quantisations. While this may provide an indication of how well systems do in general (in trying to satisfy all types of users), an appropriate matching of evaluation criteria and tasks is still needed. To this end, what is required is to define specific retrieval tasks that motivate certain user behaviours. For example, the task of highlighting highly specific relevant text fragments may reflect a user who prefers more specific elements and who may have access to the context of the highlighted text fragments. In line with this, during the INEX 2004 workshop (see http://inex.is.informatik.uni-duisburg.de:2004/presentations/metrics-wg.ppt), the following system task has been put forward for INEX 2005:

- Find the most specific elements (in each path), i.e., those elements with the highest ratio of relevant to irrelevant information. These elements are considered independent (i.e., non-overlapping), of equal quality, and it does not matter if they are from the same or different documents.

An argument that supports the selection of this task as "the" main task in INEX is that it requires search engines to pinpoint the exact location of relevant texts (and hence it is not enough to just go for a 'safe' option and return large container units). We see this as one of the main driving forces for XML IR in the first place: XML IR systems should aim to present users with more focused material, and thus reduce users' efforts in locating sought-after information. In other words, systems should return components that contain as much relevant information and as little irrelevant information as possible.

Given this retrieval task, a suitable evaluation measure should be able to rank systems according to how well they are able to locate XML elements that contain as much relevant information and as little irrelevant information as possible.

In addition to the above system task, a number of user tasks have been outlined at the workshop:

- Find the most specific elements in a path
- Find as much relevant content as possible
- Find as many relevant elements as possible

Here, the first task may be considered as an extension of the system task, where additional aspects of the user's interaction with the retrieval system may be included, e.g. browsing to structurally related elements. The second and third tasks are a bit harder to interpret and seem to be more motivated from a system-oriented point of view, whereby systems are required to return all reference elements that form the full recall-base (including all overlapping nodes).

Based (loosely) on the above task proposals, INEX 2005 defined a number of specific retrieval strategies to be investigated: "focused", "thorough" and "fetch and browse" strategies. These strategies build on assumed user behaviours that take into account how the results may actually be presented and provide explicit guidelines for search engines on how to deal with issues such as overlap. For example, the focused strategy aims to remove overlap and can be associated with a user interface where most specific elements may be highlighted for the user.

Although these sub-tasks go some way to clarify what actual output is expected of an XML IR system, they still leave a lot of questions open mainly due to the problem that we are unsure about what real users of an XML digital library would want returned to them. As a result, in the opinion of the first author, the sub-task definitions still remain open to individual interpretation, which is bound to lead to confusion and later on to questions regarding the appropriateness of the adopted evaluation metrics.

In an effort to correct this, the following modifications are suggested with respect to the focused task: "This strategy should return to the user those largest XML elements that contain *only relevant* (or minimal irrelevant[1]) information. A reason to specify 'largest' in the definition is that in case of

[1]E.g. if the most specific node on a path is $s = 2$ or $s = 1$.

a completely relevant section (e.g. $s = 3$), the section should be returned instead of its individual paragraphs (which will also have $s = 3$ since no irrelevant information is contained in the section element and consequently in any of its subnodes). More formally, the task is to return, given an arbitrary tree of relevant XML elements, the most specific non-overlapping relevant elements, where relevant simply means having any level of exhaustivity ($e > 0$). From two nodes with the same specificity the one with higher exhaustivity should be retrieved. In the case where two nodes on the same path are equally specific and exhaustive, the ascendant element should be returned. The output should be presented to the user as a ranked list of XML elements, ranked by specificity first and then by exhaustivity."

The thorough strategy, which may be motivated by the idea of using it as a catch-all for possible different retrieval strategies, may be defined as a task to "find all relevant elements, where a relevant element is one with $e > 0$. The output is assumed to be a ranked list of XML elements, ranked by combined exhaustivity and specificity according to a chosen quantisation function."[2]

The fetch and browse strategy is also felt to be rather vague. While the basic idea of the fetch phase is clear, the browse phase will need further clarification. Although the authors do not actually agree on this point, we would like to suggest as discussion point the following redefinition of this task into two separate tasks: a fetch and highlight strategy and a fetch and browse strategy.

The aim of the former strategy would be to first identify relevant articles (the fetching phase), and then to identify the most specific relevant elements within the fetched articles (the highlighting phase). In the fetching phase, articles should be ranked according to how exhaustive and specific they are, where the relative value of the combined exhaustivity and specificity would be given by a chosen quantisation function. For the highlighting phase, the ranking of XML elements within an article should be done according to the focused retrieval strategy. The assumed output is a ranked list of articles, which are then viewed by the user as flat text files, where the most specific relevant elements are highlighted.

Within the fetch and browse strategy, as with the fetch and highlight strategy, the aim of the fetching phase is to retrieve relevant articles, ranked by exhaustivity and specificity (based on a chosen quantisation function). For the browsing phase, the ranking of XML elements within an article should be done according to the thorough retrieval strategy. The assumed output is again a ranked list of articles, but on viewing the user is assumed to interact with a ranked list of XML elements from the article.

Alternative tasks may also consider the retrieval of "best elements", which involves finding the preferable units of retrieval (given a specific user interface). We, however, believe that such a task requires, as its precondition, knowledge of the locations of the most specific elements. Strategies for deciding which elements would be best to return to the user will then further depend on assumptions about the user's preferences and browsing behaviour as well as assumption about how the results are presented to the user. For example, best elements may be best hub nodes for a user of a hyperlinked environment who is happy to browse in search for relevant content. However, if the results are presented to the user as highlighted text fragments within a document unit (e.g. article), then best elements may well be the same nodes as the most specific elements.

[2]The current definition is already along these lines, but the phrasing of the task may be slightly misleading.

A further issue with the "finding the best elements" task is that it may require additional assessments, whereby given a set of relevant nodes in an XML tree and a specific user interface, users need to identify which elements they would want to be returned by a search system [12]. Note that we would not recommended to try to obtain assessments - directly within the relevance assessment procedure - with the "best elements" task in mind as it is ultimately a much more complex notion than relevance. Different people will have widely varied ideas as to what should be a best element to return (even if the user interface is fixed), which is likely to have an impact on the quality of the assessments.

Nevertheless, the best element task is one that is of particular interest to us and we would be keen to support its integration into INEX. We envision the interactive track as probably the best venue for starting experiments to investigate this task and how best to derive assessments for it. Some initial results on a different test data can be found in [19, 20].

3. RELEVANCE

3.1 Multiple dimensions and degrees

As mentioned before, the ordering of the results in the output list is according to presumed relevance. In traditional IR experiments, this output is then compared against the set of relevant documents identified by human assessors (or its subsets at different recalls). Since relevance assessments are typically given in the form of binary decisions, e.g. relevant or not, simple counting mechanisms can be employed by the evaluation measures (i.e. precision and recall).

In INEX, relevance represents a more complex notion with two separate identified aspects: exhaustivity and specificity. Both these aspects influence the overall relevance of an XML element: the more exhaustive and more specific an element, the more it is desired by the user. Exhaustivity reflects how exhaustively a document component discusses the topic of request (and hence relates to the amount of relevant information contained within the element), while specificity reflects how focused the component is on the topic of request, i.e. discusses no other, irrelevant topics (and hence relates to the amount of irrelevant information contained within the element). These two aspects have been separated into two relevance dimensions for better control. Although there have been arguments against this separation, it was decided that this solution would provide a more stable measure of relevance than if assessors were asked to rate elements on a single scale. This is because on a single scale an element may be judged, for example, marginally relevant if it contained only relevant information, but this information was not very exhaustive; and also if it was exhaustive, but the element also contained a lot of irrelevant information. Judges are also likely to place varying emphasis on these two aspects when assign a single relevance value.

This argument is supported by our findings from building a small test collection from Shakespeare plays marked up in XML. There, we employed binary relevance assessments, which were derived using a highlighting procedure (assessors marked relevant text fragments with a yellow marker), where each topic was assessed by multiple assessors. We found that different people highlighted widely different sized text fragments as being relevant to the same query. Some of the assessors highlighted very specific relevant sentences only, while others highlighted complete sections [12]. This suggests that, when considering relevance, different judges placed varying degrees of importance on the exhaustivity or specificity aspects and highlighted text segments according to a relative rating that they felt was appropriate in a given situation and at a given time. In addition, text fragments that were not strictly relevant, but provided contextual information may have also been highlighted by some of the judges.

One advantage of a single scale relevance, however, is that it implicitly combines exhaustivity and specificity (and probably other aspects too), which closer reflects the user's true preferences, rather than being modeled afterwards using quantisation functions.

In INEX, in addition to the two dimensions, it was felt that multiple grades were necessary in order to be able to reflect the relative relevance of a component with respect to its sub-components. For example, a document component may be *more* exhaustive than any of its sub-components alone given that it covers *all* (i.e. the union of) the aspects discussed in each of the sub-components. Similarly, sub-components may be *more* specific than their parent components, given that the parent components may cover multiple topics, including irrelevant ones.

The relevance degree of an assessed component, given by the combined values of exhaustivity and specificity, is denoted as $(e, s) \in ES$, where $ES = \{(0,0), (1,1), (1,2), (1,3), (2,1), (2,2), (2,3), (3,1), (3,2), (3,3)\}$.

A consequence of separating the two dimensions is that evaluation measures need to be able to either handle the dimensions separately or be able to combine them in a way that reflects appropriate user expectations. As mentioned before, the quantisation functions aim to do just that. They provide a relative ordering of the various combinations of (e, s) values and a mapping of these to a single relevance scale: $\mathbf{f}_{quant}(e, s)\colon ES \to [0, 1]$[3].

A number of relevance value functions have been in use throughout the years reflecting various user preferences. Some of these functions, e.g. strict quantisations, result in binary relevance values, while others, e.g. generalised or SOG (see Equation 2), result in multiple degree relevance scales having a range of values in $[0, 1]$. While strict quantisations lend their results suitable for an evaluation measure based on counting mechanisms, others that produce non-binary relevance values require alternative measures like generalised precision and recall or cumulated gain [13, 8, 1].

[3]Note that the quantisation functions used within the inex-2003 metric provide a separate mapping for exhaustivity, $\mathbf{f'}_{quant}(e)\colon E \to [0, 1]$ and specificity, $\mathbf{f'}_{quant}(s)\colon S \to [0, 1]$, where $E = \{0, 1, 2, 3\}$ and $S = \{0, 1, 2, 3\}$

One question that remains open, even with the defined "most specific" task, is the question of how to decide about the exact mapping to be employed (within a given quantisation function). How much would a $(2, 3)$ element be worth to the user, or how much more could it be worth than a $(1, 1)$ element?

3.2 Binary relevance

With all the additional effort involved in producing relevance assessments according to the two dimensions and along the multiple grades, arguments have been raised time and time again for the use of a simple binary relevance measure. The report of the INEX 2003 workshop [9] reports on a similar discussion, where the benefits of graded relevance assessments have again been pointed out (see [13, 7, 21]). An additional problem with binary relevance assessments is that it becomes no longer possible to reason about relative preferences among related relevant elements (i.e. component vs. its sub-components).

3.3 Continuous scale

In order to decrease assessment effort, a highlighting procedure is being considered for INEX 2005 (INEX organizers mailing list), and may even have been put in place by the time of this workshop. A proposed process for assessment is as follows:

- In the first pass, assessors highlight text fragments that contain only relevant information
- In the second pass, assessors judge the exhaustivity level of any elements that have highlighted parts.

As a result of this process, any elements that have been fully highlighted will be automatically labeled as fully specific. For example, if the last paragraph of a section (say `sec1`) and the first two paragraphs of the next section (`sec2`) have been highlighted, then the three paragraphs and any of their descendants will be marked as fully specific (e.g. $s = 1 = 100\%$). The specificity of any other (partially highlighted) elements will be calculated automatically as some function of the contained relevant and irrelevant content (e.g. in the simplest case as the ratio of relevant content to all content, measured in number of words or characters). The two sections in our example may then get a specificity score of $s = 10/100 = 10\%$ and $s = 20/100 = 20\%$, respectively, assuming that each paragraph consists of 10 words and each section has 10 paragraphs. The same procedures can be applied when highlighting is done at the sentence or word level.

The main advantage of this highlighting approach is that assessors will now only have to judge the exhaustivity level of the elements that have highlighted parts (in the second phase). A vital consideration, however, is that the highlighting must be based solely on the specificity dimension (e.g. ignoring exhaustivity in the first phase). Assessors should be made aware not to highlight larger contexts because these are more exhaustive, if at the same time they are less specific (i.e. contain irrelevant fragments). It is important that only purely relevant information fragments get highlighted.

Although, with this semi-automated method, specificity will be measured on a continuous scale, with a simple quantisation method, it can be mapped onto the already established 4 point specificity scale, if desired. However, the use of a continuous scale for specificity may also simplify the evaluation as it will no longer require a relative ordering of (e, s) pairs, but allows for a more natural combination of the two dimensions.

Although there have been suggestions for also employing a continuous scale for the exhaustivity dimension, this option has not yet been explored. It is not yet clear to us what benefits this may have and if it could lead to a reliable measure.

4. WHY DO WE NEED AN IDEAL RECALL-BASE?

In INEX, the recall-base consists of sets of overlapping elements (which will remain the case even with the proposed new assessment procedure). For example, from the XML article of co/2001/r7022.xml, all elements shown in Figure 1 form part of the recall-base for INEX 2004. As detailed in [10], this so-called *overpopulated recall-base* can lead to skewed and misleading effectiveness results if it is ignored by the employed evaluation metric. The root of this problem lies in the fact that the recall-base contains more reference elements than an ideal system should in fact retrieve. In fact, if the problem is ignored by the metric then perfect recall can only be reached by systems that return all the relevant reference components of the recall-base, including all the overlapping elements [16, 10, 3, 17]. Such retrieval behaviour, however, contradicts the definition of an effective XML retrieval system.

Following on from the focused task definition in section 2.3, systems should return only the most specific non-overlapping elements from an XML tree of relevant nodes. Based on the thorough task, ideal elements are those that score highest along a path of the XML tree according to a chosen quantisation function. Elements that correspond to such ideal nodes must also be selected from the recall-base. Given a suitable procedure, we can define an 'ideal recall-base' as a collection of ideal nodes, where overlap between reference elements is completely removed. All remaining components of the original recall-base may then be considered as near-misses.

The constructed ideal recall-base could be used (by itself) for evaluating XML retrieval systems using traditional metrics (i.e. recall and precision). In such an evaluation setting, however, systems would be measured against a rather strict ideal scenario, where only exact matches between retrieved elements and ideal reference elements are considered a hit. However, given the possibly fine graded structure of an XML document, the judgement to only credit systems that are able to return exactly the ideal components may seem too harsh, especially since the retrieval of near-misses may still be considered useful for a user when the ideal component is

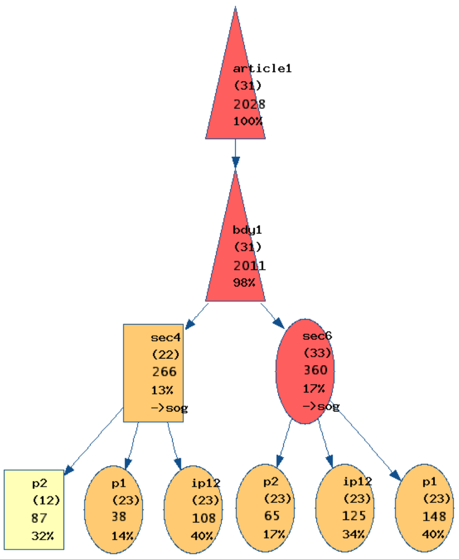


Figure 1: Sample assessments showing only relevant nodes (i.e. $e > 0$ and $s > 0$) for topic 163 in the article file co/2001/r7022.xml. For each node, the node name, the assessment value pair (e, s), the size in number of words and the size ratio to its parent node is shown. Nodes marked as "→ sog" are the selected ideal nodes based on the SOG quantisation function.

not found.

A better solution can be reached by the combined usage of the full recall-base and the derived ideal recall-base: elements in the ideal recall-base represent the desired target components that should be retrieved, while all other elements in the full recall-base (or even in the full collection) may be rewarded partial score. The main significance of the definition of an ideal recall-base is that it supports the evaluation viewpoint whereby components in the ideal recall-base *should* be retrieved, while the retrieval of near-misses *could* be rewarded as partial successes, but other systems *need not* be penalised for not retrieving such near-misses.

4.1 How to build an ideal recall-base

An ideal recall-base is a set of ideal result nodes selected from the full recall-base, where the selection process must follow assumptions regarding the given retrieval task and user behaviour. In [10], a proposed selection process was based on a chosen quantisation function, representing a user model, and the following methodology. Given any two components on a relevant path[4], the component with the higher quantised score is selected. In case two components' scores are equal, the one deeper in the tree is chosen. The procedure is applied recursively to all overlapping pairs of components along a relevant path until one element remains.

[4]A relevant path is defined as a path in an article file's XML tree, whose root node is the `article` element and whose leaf node is a relevant component (i.e. $(e>0, s>0)$) that has no or only irrelevant descendants. E.g. in Figure 1 there are 2 relevant paths.

After all relevant paths have been processed, a final filtering is applied to eliminate any possible overlap among ideal components, keeping from two overlapping ideal paths the shortest one. The resulting ideal recall-base contains the best elements to return to a user based on the assumptions that overlap between result nodes should be avoided and that the user's preferences are reflected within the employed quantisation function.

For example, using the SOG quantisation function (Equation 2), the ideal nodes selected from the XML tree shown in Figure 1 are `sec[6]` and `sec[4]`.

Based on the proposed new highlighting procedure and the focused task, the ideal elements will be the largest fully specific (or most specific[5]) elements that directly contain the highlighted relevant information.

An alternative method is proposed by Benjamin Piwowarski (in PRUM's implementation in EvalJ), where a node x is selected as ideal if:
1.) $f_{quant}(x) > 0$ AND
2.) for any descendant z of x $f_{quant}(z) < f_{quant}(x)$ AND
3.) for any ancestor y of x $f_{quant}(x) \geq f_{quant}(y)$ OR there exists a descendant z of y for which $f_{quant}(z) \geq f_{quant}(y)$.

For example, using the SOG quantisation function, the ideal nodes selected from the XML tree shown in Figure 1 are `sec[6]`, `sec[4]/ip1[2]`, `sec[4]/p[1]` and `sec[4]/p[2]`.

The difference between the two methods is that the latter places additional emphasis on selecting nodes deeper in the tree and can also cater for some assessment error, while the former relies only on the assessors' judgements. This is illustrated in Table 1, which shows the obtained ideal recall-bases for each of the sample XML trees of Figure 2, using the SOG quantisation function. For tree a) both methods select the same ideal node. For tree b) Kazai's method selects nodes 2 and 3 initially then keeps only node 2, while Piwowarski's method selects nodes 3 and 5. For tree c) Kazai's method selects node 3, while Piwowarski's method selects all relevant leaf nodes: 4,5,6,7,8 and 9.

It could be debated as to which method is better than the other. For example, in tree b) one might argue that nodes 3 and 5 provide a better representation of our user's ideal results based on the SOG user model and if the assessor had judged node 2 (`bdy[1]`) as $(e, s) = (3, 1)$ (instead of $(3, 2)$) then Kazai's method would also select these nodes as ideal. Without looking at element size, we cannot be sure if the assessor's decision was a correct one or a possible mistake[6]. An advantage of the continuous specificity scale and the highlighting assessment procedure would be that such problems would be eliminated.

Tree c) represents an interesting situation, whereby the two sets of ideal nodes cover the same content[7], but - depending on how they are presented to the user (e.g. highlighted text in an XML document or XML elements in a ranked list) and what measure is employed - could obtain different effectiveness results. This motivates the need for more elaborate methods for constructing ideal recall-bases, taking into account result presentation.

Table 1: Ideal nodes for Kazai's and Piwowarski's methods for the XML trees in Figure 2

Method	Tree a)	Tree b)	Tree c)
Kazai [10]	3.	2.	3.
Piwowarski [17]	3.	3. and 5.	4. - 9.

A metric should then take the chosen ideal recall-base as its parameter. The total score for retrieving any number of elements in a given sub-tree, having an ideal node as its root, should be limited by the quantised score of the ideal node as its maximum. For example, if in tree c) node 3 (`sec[1]`) is the ideal node (score of 1), then a run consisting of `p[4]` (0.75), `p[5]` (1) and `p[2]` (0.75) would score $min(1 - 0, 0.75) = 0.75$, $min(1 - 0.75, 1) = 0.25$ and $min(1 - (0.75 + 0.25, 0.75) = 0$, respectively. The score of retrieving an ascendant of a set of ideal nodes should, in our opinion, be based on the result's quantised score. For example, if in tree c) the ideal nodes are all the relevant leaf nodes, then `sec[1]`'s score is simply 1. It may be argued that if the assumed result presentation is highlighted text, then `sec[1]` should get the same score as the total score of the ideal nodes as it would highlight the same content. However, for this to be true we need to assume that the user has access to the context of a highlighted paragraph, in which case the ideal node should anyhow be the largest most specific element (focused task).

4.2 Don't call me ideal, I am only E3S3

There seems to be a widely popular misunderstanding of E3S3 (i.e. $(e, s) = (3, 3)$) elements being referred to as ideal results, and arguments are being raised as to how it should not be possible to have multiple nodes on a given path assessed as E3S3.

We would like to reiterate here that $(e, s) = (3, 3)$ or even $s = 3$ are NOT sufficient conditions of ideal elements! Specificity is simply a measure of the amount of relevant content vs. irrelevant content within a node. An element is highly specific ($s = 3$) *iif* it contains only relevant information (or only minimal irrelevant information), and so there is absolutely no reason why there could not be more than one highly specific nodes on a path.

This may be easier to see when considering the new highlighting assessment process. Take for example a highlighted section. The fact that it has been highlighted means that it must be fully specific, i.e. contains only relevant information. Therefore, each of its paragraph child nodes must also be fully specific (since the section contains no irrelevant information). Now, take a highlighted article that is also highly exhaustive. There is no reason why it could not have descendant elements that are also highly exhaustive

[5]For example, if only a sentence of a paragraph has been highlighted then the paragraph is selected as the ideal element.

[6]If `bdy[1]` consists only of the two sections judged relevant, then $s = 2$ is reasonable. However, if it has other irrelevant sections, then $s = 1$ would seem more appropriate.

[7]Assuming `sec[1]` does not have a text child node.

1. /a[1] $(3,1) \rightarrow 0.25$
2. /a[1]/bdy[1] $(3,1) \rightarrow 0.25$
3. /a[1]/bdy[1]/sec[1] $(3,3) \rightarrow 1$

XML tree a)

1. /a[1] $(3,1) \rightarrow 0.25$
2. /a[1]/bdy[1] $(3,2) \rightarrow 0.75$
3. /a[1]/bdy[1]/sec[1] $(2,3) \rightarrow 0.9$
4. /a[1]/bdy[1]/sec[2] $(1,1) \rightarrow 0.1$
5. /a[1]/bdy[1]/sec[2]/p[1] $(1,2) \rightarrow 0.25$

XML tree b)

1. /a[1] $(3,1) \rightarrow 0.25$
2. /a[1]/bdy[1] $(3,1) \rightarrow 0.25$
3. /a[1]/bdy[1]/sec[1] $(3,3) \rightarrow 1$
4. /a[1]/bdy[1]/sec[1]/p[1] $(1,3) \rightarrow 0.75$
5. /a[1]/bdy[1]/sec[1]/p[2] $(1,3) \rightarrow 0.75$
6. /a[1]/bdy[1]/sec[1]/p[3] $(1,3) \rightarrow 0.75$
7. /a[1]/bdy[1]/sec[1]/p[4] $(1,3) \rightarrow 0.75$
8. /a[1]/bdy[1]/sec[1]/p[5] $(3,3) \rightarrow 1$
9. /a[1]/bdy[1]/sec[1]/p[6] $(1,3) \rightarrow 0.75$

XML tree c)

Figure 2: Relevance assessments for sample XML trees. For each node, its path (with `article` shortened to `a`), exhaustivity and specificity values (e, s), and derived SOG quantised values are shown. Note that only relevant nodes are included.

(e.g. `bdy`, `app` or `sec` nodes), therefore producing a number of highly specific and highly exhaustive nodes on a path.

5. REQUIREMENTS FOR METRICS

In the previous sections we have detailed a number of requirements that a suitable measure for XML IR should take into account. We summarise these factors here.

In section 2 we stated that the main criterion of any evaluation measure is that it should be able to rank systems according to how well they satisfy a user's information need given a retrieval task and a model of user behaviour. Then during our examination of the retrieval task and user model, we noted that - due to the varying granularity of retrieval units - element size or required reading time should be taken into account when measuring users' effort to view result elements. Because of the structural relationships that exist among result elements, users' browsing behaviour should be considered. An aspect of this is that near-miss components may be considered as partial successes. Another aspect is that overlap should also need to be handled by a suitable metric.

For the focused and thorough tasks, metrics need only to consider the output as a ranked list of XML elements, with most relevant elements at the top of the ranking. Users are assumed to view the ranked list in a linear fashion, moving from the top of the list down, stopping either when the end of the list is reached, or at a point where their information need has been satisfied or where they give up. For the fetch and browse strategies, the evaluation may need to consider additional factors due to the clustering of related results.

Given that INEX employs two relevance dimensions, a measure of effectiveness should be able to either handle these dimensions separately or be able to combine them in a way that reflects a set task and user model. The metric must also be able to handle multiple degree scales of relevance (where counting mechanisms are no longer suitable). Following the proposal for a continuous scale for specificity, the ideal metric should be flexible enough to cater for both discrete and continuous scales.

As a result of the overlap of reference elements within the INEX recall-base, a suitable metric should also incorporate appropriate mechanisms to derive ideal recall-bases from the full set of assessments based on a given user model. The metric should also employ appropriate score normalisation mechanisms to ensure that the total achievable score for retrieving any combinations of relevant nodes (including the ideal node) from the sub-tree of an ideal node does not exceed the score obtainable by retrieving the ideal node itself.

In Section 6.1, we will look at the various current and proposed INEX metrics and attempt to answer whether they meet these requirements:

- Element size: Consider user effort as a function of varying granularity result elements
- Near-misses: Consider near-miss components as partial successes
- Overlap: Do not penalise systems that do not return overlapping nodes
- Output: Take into account ranking and other non-linear presentation
- Exhaustivity and specificity: Handle dimensions separately or able to combine them
- Multiple degrees: Handle multiple degree scales (and continuous scales)
- Ideal recall-base: Incorporate mechanisms to select ideal nodes from the full recall-base
- Normalisation: Incorporate mechanisms to normalise the scoring of elements in the sub-trees of ideal nodes.

6. AN ABUNDANCE OF METRICS

Up to date the following metrics have been used and/or proposed (a more detailed summary of each can be found in the Appendix)[8]:

i2: The inex-2002 (aka. inex_eval) metric [5] applies an intuitive extension of the measure of *precall* [18] to document components and computes the probability $P(rel|retr)$ that a component viewed by the user is relevant.

[8]A further metric, Expected Ratio of Relevant (ERR) [16] is not discussed here.

Table 2: Metrics and requirements matrix (y: yes, n: no, i: indirectly)

Requirements:	i2	i3	XCG	PRUM
Element size	n	y	i	n
Ideal recall-base	n	i	y	y
Near-misses	n	i	y	y
Overlap	n	y	y	y
Output: linear	y	y	y	y
Output: non-linear	n	n	n	n
Exh/Spec	y	y	y	y
Multiple degrees	n	n	y	n
Normalisation	n	n	y	n

i3: The inex-2003 (aka. inex_eval_ng) metric [6, 4] is based on an interpretation of the relevance dimensions within an ideal concept space [23]. Instead of measuring recall or precision after a certain number of document components retrieved, the total size of the retrieved document components is used as the basic parameter. For our experiments we use the version of inex-2003 detailed in [4].

XCG: The XCG (cumulated gain for XML) metrics [10, 11] are an extension of the set of cumulated gain based metrics proposed in [8] for measuring effectiveness in a traditional IR setting but considering multiple degrees of relevance.

PRUM: The PRUM (Precision Recall with User Modelling) [17] metric is an extension of the traditional recall precision metrics that considers users' browsing behaviour.

T_2I: The T_2I (Tolerance to Irrelevance) metric [3] measures success or failure based on whether the user finds relevant text starting from a returned entry point before his/her tolerance to irrelevance is reached.

The inex-2002 metric has been criticised for not considering overlap and leading to misleading effectiveness scores [10]. The inex-2003 metric's disadvantage is that it is hard to interpret and assumes that relevant information is distributed uniformly throughout a component. A shortcoming of the XCG metrics is that effectiveness is only measured at rank positions and not at recall values. PRUM is based on counting mechanisms, where the interpretation of results based on non-strict quantisations is not clear. In addition, its numerous parameters and their exact estimations may appear more of an obstacle than an advantage. T_2I has so far remained a theoretical model without concrete integration into a specific measure, and as such is not further discussed.

6.1 Metrics and requirements

In this section, we take a look at all current and proposed metrics and how they satisfy the requirements identified in the previous sections.

Table 2 lists the collected metric requirements and whether these are catered for by the various metrics.

Element size has only been considered explicitly within the definitions of recall and precision of the inex-2003 metric. XCG uses element size indirectly when calculating the relevance score of a partially seen element (see Equation 8). Element size could, however, be incorporated into PRUM, inex-2002 and directly into XCG (i.e. to measure cumulated gain against the size of the consulted text instead of its rank) by adding a quantisation function that uses element size. It is arguable, however, whether larger relevant texts should warrant higher effectiveness scores (as is the case for inex-2003. It may be more intuitive to consider element size only for irrelevant information (T_2I) or when irrelevant and relevant information is combined (as in T_2I and XCG) in a component as the amount of irrelevant information a user needs to wade through directly influences his/her satisfaction with the system.

Both XCG and PRUM make use of ideal recall-bases. The mechanisms for deriving an ideal recall-base, based on a given user model (quantisation function and assumptions about overlap), are currently implemented as an integral part of the metrics. However, there are plans to allow for a more flexible setup, where arbitrary ideal recall-bases can be applied as a parameter of the metrics. The version of the inex-2003 metric detailed in [4] defines an entity Rel^U, which represents the maximum number of relevant concepts in the full recall-base (counting a relevant concept only once)[9]. This could hence be interpreted as the total relevance score of an ideal recall-base, whose elements are chosen to maximise the total relevance score for the collection's XML tree. In general, this leads to the ideal recall-base consisting of relevant leaf nodes (i.e. the deepest relevant nodes). Since the definition of Rel^U is fixed (due to concept space), it can only be associated with a single given user model and result presentation (a bit like recall and precision).

Both PRUM and XCG are able to give partial reward for near-misses (due to the fact that they both make use of an ideal recall-base). Unlike XCG, however, PRUM is also able to consider irrelevant sibling nodes as near-misses. PRUM does this by increasing the score of a result element (even if irrelevant) if it has structural links to relevant content (based on assumptions about the user's browsing behaviour: no, hierarchical or T_2I browsing). XCG relies only on the ideal and full recall-bases for determining a near-miss. The latest version of inex-2003 [4] also (indirectly) supports the evaluation of near-misses due to scoring elements based on the full recall-base while the collection's total relevance score is based on Rel^U.

Overlap is handled by all metrics except the inex-2002 measure. In XCG overlap is handled within the relevance value functions, which return a node's unmodified quantised value if it has not yet been seen, and otherwise calculate a modified relevance score if it has been seen in full or in part. The relevance value of partially seen elements is derived recursively based on the size and relevance score of the node's not-yet-seen descendants. In inex-2003, overlap is handled

[9]The earlier version of the inex-2003 metric [6] calculated total relevance as $\sum_{i=1}^{N} q_e(e)$ over all N elements of the full recall-base, which resulted in the same problems as with the inex-2002 metric that 100% recall could only be reached by systems returning the full recall-base.

in a similar way, by only considering the not-yet-seen parts, but the relevance score is estimated by assuming that relevant information is distributed uniformly within the component. This means that a section will still obtain a score even if its only relevant paragraph has already been seen. PRUM employs probability estimations for a user's browsing behaviour, and updates the probability of a node being seen by the user depending on its structural relationship to the currently visited node and assumptions about the user's interaction (i.e. no, hierarchical or T_2I browsing). The more structurally related elements have been returned to the user and hence the more chances the user had to access the current result element, the more its score is reduced.

There is no difference between the four metrics as far as the output presentation is concerned: they are all able to evaluate linear ranked result lists. Further investigation of how the metrics can be adapted to deal with clustered representations is required.

All the metrics are able to cope with the two relevance dimensions via the use of quantisation functions.

Since all metrics, except XCG, are extensions of recall and precision, they are all based on counting mechanisms that result in non-perfect effectiveness for ideal runs (see Section 6.2). For example, although PRUM does work with multiple degree relevance scales, it only produces perfect score for an ideal run, if a strict quantisation (or the recently added "binary" option) is applied.

All metrics can adopt a continuous specificity scale via the definition of a suitable quantisation function. For example, a simple quantisation function may be given as: $f_{quant} = q_e(e) \cdot q_s(s)$, where $q_e(e) = e/3$ and $q_s(s) = s$ if $s \in [0,1]$ (where $s = 1$ would mean fully specific). The function f_{quant} would be used by the metrics inex-2002, XCG and PRUM, while the functions $q_e(e)$ and $q_s(s)$ could be used directly in inex-2003.

Normalisation mechanisms to ensure that the total achievable score for retrieving any combinations of relevant nodes (including the ideal node) from the sub-tree of an ideal node does not exceed the score obtainable by retrieving the ideal node itself are implemented in XCG [11].

From the above, it seems that no single metric ticks all the requirements, although the inex-2002 metric seems to be the one lagging behind all others. A reason for this is that most of the problems associated with the evaluation of XML retrieval have not actually came to light until after the first effectiveness results were in. For example, implicit assumptions about overlap (i.e. that systems would avoid returning overlapping nodes) meant that overlap was not explicitly considered by the metric. An obvious question is whether the inex-2002 metric could be extended upon to cater for the additional requirements. We will examine this question in future work.

6.2 What do they measure

In this section we detail the results of some very simple experiments, where we investigated the behaviour of four of the INEX metrics with the use of a single relevant XML tree (taken from the INEX 2004 recall-base for the topic 163). We used four simulated runs for the experiments: see Table 3. The result elements of all runs have been sorted according to our chosen quantization function: SOG (Equation 2).

Table 3: Simulated runs

`frb_SOG_163_r7022:  #All relevant nodes in topic 163's assessments for co/2001/r7022.xml, sorted by SOG quantised value (see Figure 1).`

1. `/article[1]/bdy[1]/sec[6]` $(3,3) \rightarrow 1$
2. `/article[1]/bdy[1]/sec[4]/ip1[2]` $(2,3) \rightarrow 0.9$
3. `/article[1]/bdy[1]/sec[4]/p[1]` $(2,3) \rightarrow 0.9$
4. `/article[1]/bdy[1]/sec[6]/ip1[2]` $(2,3) \rightarrow 0.9$
5. `/article[1]/bdy[1]/sec[6]/p[1]` $(2,3) \rightarrow 0.9$
6. `/article[1]/bdy[1]/sec[6]/p[2]` $(2,3) \rightarrow 0.9$
7. `/article[1]/bdy[1]/sec[4]` $(2,2) \rightarrow 0.5$
8. `/article[1]` $(3,1) \rightarrow 0.25$
9. `/article[1]/bdy[1]` $(3,1) \rightarrow 0.25$
10. `/article[1]/bdy[1]/sec[4]/p[2]` $(1,2) \rightarrow 0.25$

`irb_SOG_163_r7022:  #Ideal nodes from the full recall-base run above, based on Kazai's method and sorted by SOG quantised value.`

1. `/article[1]/bdy[1]/sec[6]` $(3,3) \rightarrow 1$
2. `/article[1]/bdy[1]/sec[4]` $(2,2) \rightarrow 0.5$

`reverse_irb_SOG_163_r7022:  #Nodes from the ideal run above, but in reverse order.`

1. `/article[1]/bdy[1]/sec[4]` $(2,2) \rightarrow 0.5$
2. `/article[1]/bdy[1]/sec[6]` $(3,3) \rightarrow 1$

`lo_SOG_163_r7022:  #All relevant leaf nodes from the full recall-base run, sorted by SOG quantised value.`

1. `/article[1]/bdy[1]/sec[6]/ip1[2]` $(2,3) \rightarrow 0.9$
2. `/article[1]/bdy[1]/sec[6]/p[1]` $(2,3) \rightarrow 0.9$
3. `/article[1]/bdy[1]/sec[6]/p[2]` $(2,3) \rightarrow 0.9$
4. `/article[1]/bdy[1]/sec[4]/ip1[2]` $(2,3) \rightarrow 0.9$
5. `/article[1]/bdy[1]/sec[4]/p[1]` $(2,3) \rightarrow 0.9$
6. `/article[1]/bdy[1]/sec[4]/p[2]` $(1,2) \rightarrow 0.25$

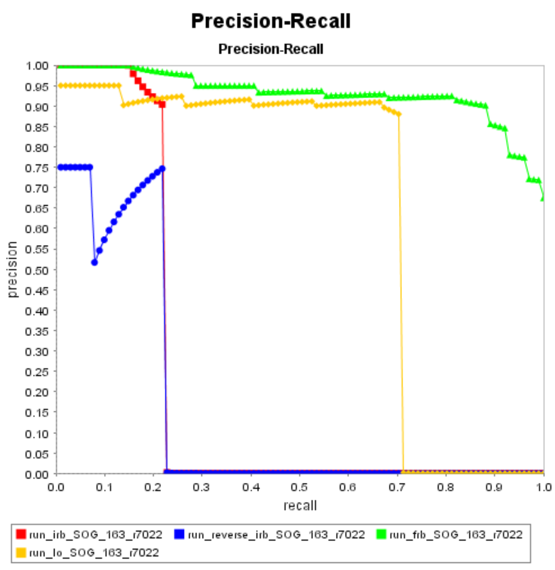
We used the EvalJ source code for the evaluation[10], which implements all four metrics within a single java project.

The runs were evaluated against a full recall-base consisting only of the relevant nodes from the article file `co/2001/r7022` from the assessments of topic 163 (Figure 1). For PRUM and XCG, the ideal recall-bases were automatically generated using the SOG quantisation function during the evaluation (using Kazai's algorithm, detailed in section 4.1)[11].

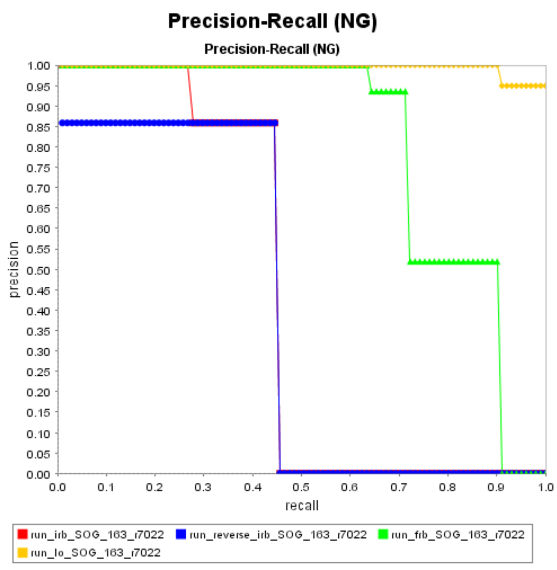
As it can be seen in Figure 3, the inex-2002 metric ranks the reverse ideal run worst followed by the ideal run, which performs slightly better than its reversed version at low recalls. This is intuitive and reflects that highly relevant elements are expected to be ranked before less relevant elements. Ta-

[10]https://sourceforge.net/projects/evalj/

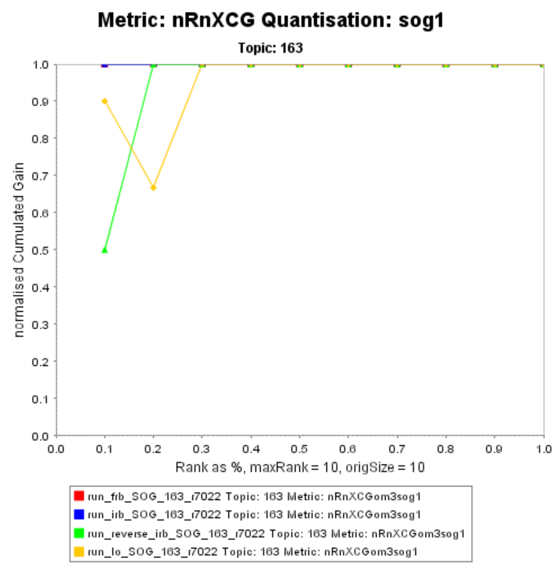
[11]Note that for this, we modified PRUM's code in EvalJ so that the same ideal recall-base is created as with XCG: evalj.corpus.AssessDoxel.addIdealDoxels method.



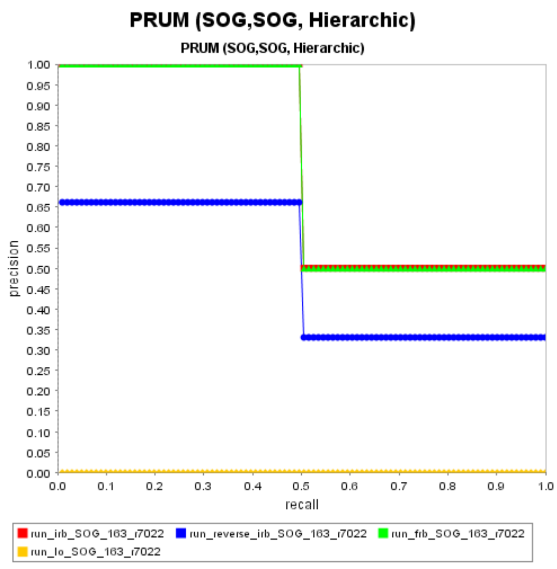
a) inex-2002 metric



b) inex-2003 metric

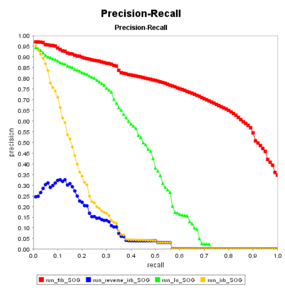


c) nXCG metric

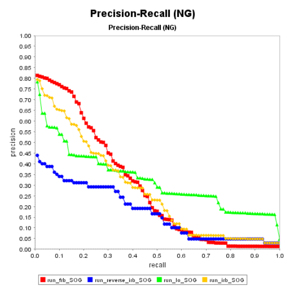


d) PRUM metric

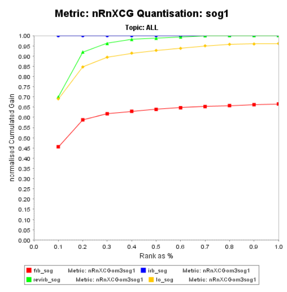
Figure 3: Effectiveness scores for a single XML tree in the article file co/2001/r7022 in topic 163, using the SOG quantisation



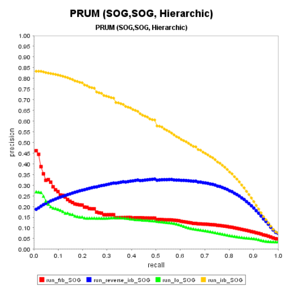
a) inex-2002 metric



b) inex-2003 metric



c) nXCG metric



d) PRUM metric

Figure 4: Effectiveness scores for all INEX 2004 CO topics, using the SOG quantisation

Table 4: Effectiveness scores for the irb_SOG_163_r7022 simulated run (see Appendix for formulas and Figure 1 for element size information)

run	i2	i3	XCG
	$n = 6.75$	$Rel^U = 3.67$	$max_{XCG_{ideal}} = 1.5$
1.	$x = \frac{1}{6.75} = 0.14$	$r = \frac{1 \cdot \frac{360}{360}}{3.67} = 0.27$	$rank = 1$
	$p = \frac{1}{1+0+\frac{1 \cdot 0}{1+1}} = 1$	$p = \frac{1 \cdot 360}{360} = 1$	$nXCG = \frac{1}{1} = 1$
2.	$x = \frac{1+0.5}{6.75} = 0.22$	$r = \frac{1 \cdot \frac{360}{360} + 0.67 \cdot \frac{266}{266}}{3.67} = 0.45$	$rank = 2$
	$p = \frac{1.5}{1.5+0+\frac{0.5 \cdot 0.5}{1+0.5}} = 0.9$	$p = \frac{1 \cdot 360 + 0.67 \cdot 266}{360+266} = 0.86$	$nXCG = \frac{1+0.5}{1.5} = 1$

Table 5: Effectiveness scores for the reverse_irb_SOG_163_r7022 simulated run (see Appendix for formulas and Figure 1 for element size information)

run	i2	i3	XCG
	$n = 6.75$	$Rel^U = 3.67$	$max_{XCG_{ideal}} = 1.5$
1.	$x = \frac{0.5}{6.75} = 0.07$	$r = \frac{0.67 \cdot \frac{266}{266}}{3.67} = 0.18$	$rank = 1$
	$p = \frac{0.5}{0.5+0+\frac{0.5 \cdot 0.5}{1+0.5}} = 0.75$	$p = \frac{0.67 \cdot 266}{266} = 0.67$	$nXCG = \frac{0.5}{1} = 0.5$
2.	$x = \frac{1+0.5}{6.75} = 0.22$	$r = \frac{1 \cdot \frac{360}{360} + 0.67 \cdot \frac{266}{266}}{3.67} = 0.45$	$rank = 2$
	$p = \frac{1.5}{1.5+0.5+\frac{1 \cdot 0}{1+1}} = 0.75$	$p = \frac{0.67 \cdot 266 + 1 \cdot 360}{360+266} = 0.86$	$nXCG = \frac{0.5+1}{1.5} = 1$

Table 6: Effectiveness scores for the frb_163_r7022_sog simulated run (see Appendix for formulas and Figure 1 for element size information)

run	i2	i3	XCG
	$n = 6.75$	$Rel^U = 3.67$	$max_{XCG_{ideal}} = 1.5$
1.	$x = \frac{1}{6.75} = 0.14$	$r = \frac{1 \cdot \frac{360}{360}}{3.67} = 0.27$	$rank = 1$
	$p = \frac{1}{1+0+\frac{1 \cdot 0}{1+1}} = 1$	$p = \frac{1 \cdot 360}{360} = 1$	$nXCG = \frac{min(1-0,1)}{1} = 1$
2.	$x = \frac{1+0.9}{6.75} = 0.28$	$r = \frac{1 \cdot \frac{360}{360} + 0.67 \cdot \frac{108}{108}}{3.67} = 0.45$	$rank = 2$
	$p = \frac{1.9}{1.9+0+\frac{0.9 \cdot 0.1}{1+0.9}} = 0.975$	$p = \frac{1 \cdot 360 + 1 \cdot 108}{360+108} = 1$	$nXCG = \frac{1+min(0.5-0,0.9)}{1.5} = 1$
3.	$x = \frac{1.9+0.9}{6.75} = 0.41$	$r = \frac{(1.67+0.67 \cdot \frac{38}{38}}{3.67} = 0.637$	$rank = 3$
	$p = \frac{2.8}{2.8+0.1+\frac{0.9 \cdot 0.1}{1+0.9}} = 0.95$	$p = \frac{468+1 \cdot 38}{468+38} = 1$	$nXCG = \frac{1.5+min(0.5-0.5,0.9)}{1.5} = 1$
4.	$x = \frac{2.8+0.9}{6.75} = 0.54$	$r = \frac{(2.34+0.67 \cdot \frac{0}{125}}{3.67} = 0.637$	$rank = 4$
	$p = \frac{3.7}{3.7+0.2+\frac{0.9 \cdot 0.1}{1+0.9}} = 0.937$	$p = \frac{506+1 \cdot 0}{506+0} = 1$	$nXCG = \frac{1.5+0}{1.5} = 1$
5.	$x = \frac{3.7+0.9}{6.75} = 0.68$	$r = \frac{(2.34+0.67 \cdot \frac{0}{148}}{3.67} = 0.637$	$rank = 5$
	$p = \frac{4.6}{4.6+0.3+\frac{0.9 \cdot 0.1}{1+0.9}} = 0.929$	$p = \frac{506+1 \cdot 0}{506+0} = 1$	$nXCG = \frac{1.5+0}{1.5} = 1$
6.	$x = \frac{4.6+0.9}{6.75} = 0.81$	$r = \frac{(2.34+0.67 \cdot \frac{0}{65}}{3.67} = 0.637$	$rank = 6$
	$p = \frac{5.5}{5.5+0.4+\frac{0.9 \cdot 0.1}{1+0.9}} = 0.924$	$p = \frac{506+1 \cdot 0}{506+0} = 1$	$nXCG = \frac{1.5+0}{1.5} = 1$
7.	$x = \frac{5.5+0.5}{6.75} = 0.88$	$r = \frac{(2.34+0.67 \cdot \frac{266-108-38}{266}}{3.67} = 0.72$	$rank = 7$
	$p = \frac{6}{6+0.5+\frac{0.5 \cdot 0.5}{1+0.5}} = 0.90$	$p = \frac{506+0.67 \cdot (266-108-38)}{506+120} = 0.936$	$nXCG = \frac{1.5+min(0.5-0.5,0.5)}{1.5} = 1$
8.	$x = \frac{6+0.25}{6.75} = 0.925$	$r = \frac{(2.64+1 \cdot \frac{2028-266-360}{2028}}{3.67} = 0.9$	$rank = 8$
	$p = \frac{6.25}{6.25+1+\frac{0.25 \cdot 0.75}{1+0.25}} = 0.84$	$p = \frac{586.4+0.34 \cdot (2028-266-360)}{626+1402} = 0.524$	$nXCG = \frac{1.5+0}{1.5} = 1$
9.	$x = \frac{6.25+0.25}{6.75} = 0.96$	$r = \frac{(3.33+1 \cdot \frac{0}{2011}}{3.67} = 0.9$	$rank = 9$
	$p = \frac{6.5}{6.5+1.75+\frac{0.25 \cdot 0.75}{1+0.25}} = 0.77$	$p = \frac{1063.08+0.34 \cdot (0)}{2028} = 0.524$	$nXCG = \frac{1.5+0}{1.5} = 1$
10.	$x = \frac{6.5+0.25}{6.75} = 1$	$r = \frac{(3.33+0.34 \cdot \frac{0}{87}}{3.67} = 0.9$	$rank = 10$
	$p = \frac{6.75}{6.75+2.5+\frac{0.25 \cdot 0.75}{1+0.25}} = 0.71$	$p = \frac{1063.08+0.67 \cdot (0)}{2028} = 0.524$	$nXCG = \frac{1.5+0}{1.5} = 1$

bles 4 and 5 show that the reduced effectiveness of the reversed ideal run is due to the irrelevant score obtained for `sec[4]` $(1 - 0.5)$ contributing to Cooper's variable i (irrelevant score at current rank) at rank 1 and then to variable j (irrelevant score up to current rank) at rank 2.

According to the inex-2002 metric, the full recall-base run performs best, followed by the leaf-only run. This is expected as inex-2002 calculates the 100% recall value as the sum of the quantised values of all elements in the full recall-base. Therefore, 100% recall is only reached by the full recall-base run. However, even returning the whole recall-base still does not result in perfect precision. This slope of the precision curve is due to the use of the non-binary relevance scale. Since the quantised exhaustivity and specificity values directly influence the effectiveness score, any quantised values < 1 will result in non-perfect precision scores. For example, at rank 2 of the full recall-base run Cooper's r and s is 0.9, which then results in $i = 1 - 0.9 = 0.1$ and at rank 3 this 0.1 irrelevant score is addedd to Cooper's j variable. While the estimation of these variables in Cooper's formula were based on counting mechanisms (i.e. the number of irrelevant documents), their interpretation in INEX is that of relevance or irrelevence value, where the underlying assumption is that $r = 1 - i$. A problem here is that $r < 1$ does not necessarily mean that a retrieved element contains $1 - r$ irrelevant information, e.g. $f_{SOG}(2, 3) = 0.9$. Employing a quantisation function where $(e, 3) \to 1$ provides only a partial solution, due to possible XML trees where no $s = 3$ nodes exist, while also resulting in a metric that is insensitive to the level of exhaustivity. One solution to the problem would be to calculate Copper's parameters at a given rank in relation to a maximum ideal relevance score achievable at that rank (instead of using 1), e.g. hence resulting in $i = 0$ in the above example as 0.9 is the highest achievable relevance score at rank 2.

Similarly to the inex-2002 metric, the inex-2003 metric ranks the reverse ideal run worst followed by the ideal run, where the ideal run performs slightly better than its reversed version at low recalls. The reason that these runs don't achieve perfect recall is because Rel^U is calculated from the relevant leaf nodes' quantised scores: $Rel^U = 0.\dot{6} \cdot 5 + 0.\dot{3}$. I.e. our ideal test run does not actually match an ideal run for inex-2003. However, the precision values are not affected by this problem, but are nevertheless inperfect. This is again due to the non-binary relevance grades, where normalising the actual relevance score by a maximum score could provide a solution. Unlike inex-2002, the inex-2003 metric ranks the leaf-only run best followed by the full recall-base run. The reason for this is twofold. On the one hand, the leaf-only run is actually the ideal run for inex-2003, and so it achieves 100% recall. On the other hand, the overlap present in the full-recall-base run leads to reduced performance at various recall levels, depending on the ordering of the elements within the run. The reason that even the perfect run for inex-2003, i.e. the leaf-only run, does not achieve perfect precision is simply because `sec[4]/p[2]` has $q_s(s) < 1$, which directly reduces precision.

XCG is the only metric that shows the ideal run having a perfect score of 1. It also shows that in this special case the run derived from the full recall-base achieves the same result as the ideal run. This is because the first two nodes in frb's ranking match exactly the ideal run (due to results being sorted by SOG value): `sec[6]` and `sec[4]`. Therefore, for the first two ranks, the full recall-base run matches the ideal run and hence achieves maximum score. Due to the fact that all remaining nodes in the full recall-base run overlap with an already retrieved node, no further scores are accumulated. The reverse ideal and leaf only runs perform very similar to the ideal, only dipping slightly at the beginning of the curve. The reverse ideal run's non-perfect score is due to the non-ideal ordering of its elements. The leaf-only run starts off at 0.9 normalised cumulated gain, but then drops due to the fact that the cumulated relevance score of further elements in the sub-tree of `sec[6]` ideal node is not allowed to exceed the ideal node's score (i.e. `sec[6]/ip1[2]` scores 0.9, then `sec[6]/p[1]` scores only 0.1, etc.).

PRUM ranks the ideal and full recall-base runs as best (identical performance). The reverse ideal run comes in at third place and the leaf-only run scores the worst. For PRUM there are two possible relevant units ($P(TR = 1) = 0.5$ and $P(TR = 2) = 0.5$ where TR is the total number of relevant elements). Let's consider both cases: Case 1) $TR = 1$ (i.e. only `sec[4]` is relevant for the user). Then both the full recall-base and the ideal runs achieve precision 1 for all recall levels. The leaf-only run's precision is close to 0 due to the fact that for each result the user will potentially have to inspect all the elements of the database to find the relevant nodes. The reverse ideal run's precision is 1/2. Case 2) $TR = 2$ (i.e. both `sec[4]` and `sec[6]` are relevant to the user). Then for $R = 1$ (i.e. recall level = 0.5) we can arrive at the same observations as in Case 1). This is a problem with the non-binary assessments which is only visible when the number of relevant elements is low. In this case, a human should infer that `sec[6]` is relevant. But for PRUM, $Pr(sec[6]/TR = 2)$ is still 0.5, which more or less implies that the second relevant element will have to be searched again in the whole database $\to$ precision is near to 0 for this case. After that, curves are obtained taking the average of Case 1) and Case 2).

For reference, we also include the results for a further four simulated runs, which are based on all INEX 2004 CO topics, see Figure 4.

7. CONCLUSIONS

In this paper we focused on issues regarding the evaluation of XML retrieval. We identified a number of requirements that a suitable measure of XML retrieval effectiveness should meet. We commented on the current task definitions and provided suggestions for their future development. We also expressed support for the proposed continuous specificity dimension and reported on an assessment framework to support it. Finally, we examined four of the current and proposed metrics: how they fit the requirements and how they behave when only a single XML tree formed the recall-base.

Our findings showed that although no single metric met all requirements, the XCG and PRUM metrics showed potential. In addition, the XCG metric seemed to behave the most intuitively (best matching expectation), although PRUM also produced intuitive results when a binary quantisation function was used (Figure not included).

Our future work concentrates on recall-oriented XCG based on [13, 8, 1] adopted to XML, and in particular to INEX.

8. ACKNOWLEDGMENTS

We would like to thank Benjamin Piwowarski for implementing the inex-2002 and inex-2003 metrics in EvalJ, and for his many useful comments and email discussions. Special thanks for providing the ideal recall-base algorithm for PRUM, the basis for the examples in Figure 2 and the explanatory text on PRUM in Section 6.2.

9. REFERENCES

[1] G. Amati. *Probability Models for Information Retrieval based on Divergence from Randomness*. PhD thesis, University of Glasgow, 2003.

[2] W. Cooper. Expected search length: A single measure of retrieval effectiveness based on the weak ordering action of retrieval systems. *American Documentation*, 19(1):30–41, 1968.

[3] A. de Vries, G. Kazai, and M. Lalmas. Tolerance to irrelevance: A user-effort oriented evaluation of retrieval systems without predefined retrieval unit. In *Recherche d'Informations Assistée par Ordinateur (RIAO 2004)*, Avignon, France, Apr. 2004. To appear.

[4] N. Goevert, N. Fuhr, M. Lalmas, and G. Kazai. Evaluating the effectiveness of content-oriented xml retrieval. *Submitted to Information Retrieval*, 2005.

[5] N. Gövert and G. Kazai. Overview of the INitiative for the Evaluation of XML Retrieval (INEX) 2002. In N. Fuhr, N. Gövert, G. Kazai, and M. Lalmas, editors, *Proceedings of the First Workshop of the INitiative for the Evaluation of XML Retrieval (INEX). Dagstuhl, Germany, December 8–11, 2002*, ERCIM Workshop Proceedings, pages 1–17, Sophia Antipolis, France, March 2003. ERCIM. http://www.ercim.org/publication/ws-proceedings/INEX2002.pdf.

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

[7] K. Järvelin and J. Kekäläinen. IR evaluation methods for retrieving highly relevant documents. In N. Belkin, P. Ingwersen, and M.-K. Leong, editors, *Proceedings of the 23rd Annual ACM SIGIR Conference on Research and Development in Information Retrieval*, pages 41–48, Athens, Greece, 2000.

[8] K. Järvelin and J. Kekäläinen. Cumulated Gain-based evaluation of IR techniques. *ACM Transactions on Information Systems (ACM TOIS)*, 20(4):422–446, 2002.

[9] G. Kazai. Report of the inex 2003 metrics working group. In N. Fuhr, M. Lalmas, and S. Malik, editors, *Proceedings of the 2nd Workshop of the INitiative for the Evaluation of XML retrieval (INEX), Dagstuhl, Germany, December 2003*, pages 184–190, April 2004.

[10] G. Kazai, M. Lalmas, and A. de Vries. The overlap problem in content-oriented XML retrieval evaluation. In *Proceedings of the 27th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval, Sheffield, UK, 2004.*, pages 72–79. ACM, July 2004.

[11] G. Kazai, M. Lalmas, and A. de Vries. Reliability tests for the xcg and inex-2002 metrics. In N. Fuhr, M. Lalmas, S. Malik, and Z. Szlavik, editors, *Advances in XML Information Retrieval. Third Workshop of the INitiative for the Evaluation of XML Retrieval INEX 2004, Schloss Dagstuhl, 6-8 December 2004*, 3493, pages 60–72. Lecture Notes in Computer Science, 2005.

[12] G. Kazai, M. Lalmas, and J. Reid. Construction of a test collection for the focussed retrieval of structured documents. In F. Sebastiani, editor, *Advances in Information Retrieval, Proceedings of the 25th European Conference on IR Research, Pisa, Italy*, volume 2633 of *Lecture Notes in Computer Science*, pages 88–103. Springer, April 2003.

[13] J. Kekäläinen and K. Järvelin. Using graded relevance assessments in IR evaluation. *Journal of the American Society for Information Science and Technology*, 53(13):1120–1129, 2002.

[14] M. Lalmas. Inex 2005 retrieval task and result submission specification. Technical report, Queen Mary, University of London, 2005.

[15] M. Lalmas and S. Malik. Inex 2004 retrieval task and result submission specification. In N. Fuhr, M. Lalmas, S. Malik, and Z. Szlavik, editors, *Advances in XML Information Retrieval. Third Workshop of the INitiative for the Evaluation of XML Retrieval INEX 2004, Schloss Dagstuhl, 6-8 December 2004*, 3493, pages 237–240. Lecture Notes in Computer Science, 2005.

[16] B. Piwowarski and P. Gallinari. Expected ratio of relevant units: A measure for structured document information retrieval. In N. Fuhr, M. Lalmas, and S. Malik, editors, *Proceedings of the 2nd Workshop of the INitiative for the Evaluation of XML retrieval (INEX), Dagstuhl, Germany, December 2003*, pages 158–166, April 2004.

[17] B. Piwowarski, P. Gallinari, and G. Dupret. Precision Recall with User Modelling: Application to XML retrieval. *Submitted for publication*, 2005.

[18] V. Raghavan, P. Bollmann, and G. Jung. A critical investigation of recall and precision. *ACM Transactions on Information Systems*, 7(3):205–229, 1989.

[19] J. Reid, M. Lalmas, K. Finesilver, and M. Hertzum. Best entry points for structured document retrieval - part i: Characteristics. *Information Processing & Management*, 2005. In press.

[20] J. Reid, M. Lalmas, K. Finesilver, and M. Hertzum. Best entry points for structured document retrieval - part ii: Types, usage and effectiveness. *Information Processing & Management*, 2005. In press.

[21] E. Sormunen. Liberal relevance criteria of trec - counting on negligible documents? In K. Järvelin, M. Beaulieu, R. Baeza-Yates, and S. Myaeng, editors, *Proceedings of the Twenty-Fifth Annual ACM SIGIR Conference on Research and Development in Information Retrieval*, Tampere, Finland, 2002.

[22] T. Tombros, B. Larsen, and S. Malik. The interactive track at INEX 2004. In N. Fuhr, M. Lalmas, S. Malik, and Z. Szlavik, editors, *Proceedings of the 3rd Workshop of the INitiative for the Evaluation of XML retrieval (INEX), Dagstuhl, Germany, December 2004*, 2005.

[23] S. Wong and Y. Yao. On modeling information retrieval with probabilistic inference. *ACM Transactions on Information Systems*, 13(1):38–68, 1995.

APPENDIX

A. METRICS

A.1 The inex-2002 metric

The inex-2002 metric [5] applies the measure of *precall* [18] to document components and computes the probability $P(rel|retr)$ that a component viewed by the user is relevant:

$$P(rel|retr)(x) : \frac{x \cdot n}{x \cdot n + esl_{x \cdot n}} \quad = \quad \frac{x \cdot n}{x \cdot n + j + \frac{s \cdot i}{r+1}} \quad (1)$$

where $esl_{x \cdot n}$ denotes the *expected search length* [2], i.e. the expected number of non-relevant elements retrieved until an arbitrary recall point x is reached, and n is the total number of relevant components with respect to a given topic. In $esl_{x \cdot n}$, let l denote the rank from which the $x \cdot n$th relevant component is drawn. Then j is the score of non-relevant information within the ranks before rank l, s is the relevant score to be taken from rank l, and r and i are the relevant and non-relevant scores in rank l, respectively.

To apply the above metric, the two relevance dimensions are first mapped to a single relevance scale by employing a quantisation function, $\mathbf{f}_{quant}(e, s)\colon ES \rightarrow [0, 1]$. There are a number of quantisation functions currently in use in INEX, e.g. strict or generalised (see Equations 2 and 3 in [9]), each representing a different set of user preferences. The "specificity-oriented generalised" (SOG) quantisation function proposed in [10] is given as:

$$\mathbf{f}_{SOG}(e, s) := \begin{cases} 1 & \text{if} \quad (e, s) = (3, 3) \\ 0.9 & \text{if} \quad (e, s) = (2, 3) \\ 0.75 & \text{if} \quad (e, s) \in \{(1, 3), (3, 2)\} \\ 0.5 & \text{if} \quad (e, s) = (2, 2) \\ 0.25 & \text{if} \quad (e, s) \in \{(1, 2), (3, 1)\} \\ 0.1 & \text{if} \quad (e, s) \in \{(2, 1), (1, 1)\} \\ 0 & \text{if} \quad (e, s) = (0, 0) \end{cases} \quad (2)$$

A.2 The inex-2003 metric

The inex-2003 metric incorporates component size and overlap within the definition of recall and precision (Equations 3 and 4). (For the derivation of the formulae based on an interpretation of the relevance dimensions within an ideal concept space [23] refer to [6].) Instead of measuring, e.g., precision or recall after a certain number of document components retrieved, the total size of the retrieved document components is used as the basic parameter, while overlap is accounted by considering only the increment to the parts of the components already seen. The calculations here assume that relevant information is distributed uniformly throughout a component.

$$\text{recall}_o \frac{\sum_{i=1}^{k} e\,(c_i) \cdot \frac{|c'_i|}{|c_i|}}{Rel^U} \quad (3)$$

$$\text{precision}_o \frac{\sum_{i=1}^{k} s\,(c_i) \cdot |c'_i|}{\sum_{i=1}^{k} |c'_i|} \quad (4)$$

Components $c_1, \ldots, c_k$ in Equations 3 and 4 form a ranked result list, N is the total number of components in the collection, $e(c_i)$ and $s(c_i)$ denote the quantised assessment value of component c_i according to the exhaustivity and specificity dimensions, respectively, $|c_i|$ denotes the size of the component, while $|c_i'|$ is the size of the component that has not been seen by the user previously. Given a component representation such as a set of (term, position) pairs, $|c_i'|$ can be calculated as:

$$|c_i'| = |c_i - \bigcup_{c \in C[1,n-1]} (c)| \qquad (5)$$

where n is the rank position of c_i in the output list, and $C[1, n-1]$ is the set of components retrieved between the ranks $[1, n-1]$.

A.3 The XCG metrics

The XCG metrics are extensions of the cumulated gain (CG) based metrics of [8]. The motivation for the CG metrics was to develop a measure for multi-grade relevance values, i.e. to credit IR systems according to the retrieved documents' degree of relevance. The motivation for XCG was to extend CG in such a way that the problem of overlapping result and reference elements can be addressed within the evaluation framework.

The Cumulated Gain (CG) measure, accumulates the relevance scores of retrieved documents along the ranked list G, where the document IDs are replaced with their relevance scores. The cumulated gain at rank i, $CG[i]$, is computed as the sum of the relevance scores up to that rank:

$$\mathbf{CG[i]} := \sum_{j=1}^{i} G[j] \qquad (6)$$

For each query, an ideal gain vector, I, can be derived by filling the rank positions with the relevance scores of all documents in the recall-base in decreasing order of their degree of relevance. A retrieval run's CG vector can then be compared to this ideal ranking by plotting the gain value of both the actual and ideal CG functions against the rank position. We obtain two monotonically increasing curves (levelling after no more relevant documents can be found).

By dividing the CG vectors of the retrieval runs by their corresponding ideal CG vectors, we obtain the normalised CG (nCG) measure. Here, for any rank the normalised value of 1 represents ideal performance. The area between the normalised actual and ideal curves represents the quality of a retrieval approach.

XCG makes use of both the CG and nCG metrics. The extension of these metrics to XML documents, and in particular to INEX, lies partly in the way the relevance score for a given document - or in this case document component - is calculated via the definition of so-called relevance value (RV) functions, and partly in the definition of the ideal recall-bases.

While I is derived from the ideal recall-base, the gain vectors, G, for the runs under evaluation are based on the full recall-base in order to enable the scoring of near-miss components. All relevant components of the full recall-base that are not included in the ideal recall-base are considered as near-misses.

In order to obtain a given component's relevance score (both for I or G) at a given rank position, XCG defines the following result-list dependent relevance value (RV) function:

$$rv(c_i) = f(quant(assess(c_i))) \qquad (7)$$

where $assess(c_i)$ is a function that returns the assessment value pair for the component c_i, if given within the recall-base and $(0, 0)$ otherwise. The $rv(c_i)$ function then returns, for a not-yet-seen component c_i, the quantised assessment value pair $quant(assess(c_i))$, where quant is a chosen quantisation functions, e.g. *sog*. In this case $f(x) = x$. For a component, which has been previously fully seen by the user, we have $rv(c_i) = (1 - \alpha) \cdot quant(assess(c_i))$, i.e. $f(x) = (1 - \alpha) \cdot x$. With α set to 1, the RV function returns 0 for a fully seen, hence redundant, component, reflecting that it represents no value to the user any more. Finally, if c_i has been seen only in part before (i.e. some descendant nodes have already been retrieved earlier in the ranking), then $rv(c_i)$ is calculated as:

$$rv(c_i) = \alpha \cdot \frac{\sum_{j=1}^{m} (rv(c_j) \cdot |c_j|)}{|c_i|} + (1 - \alpha) \cdot quant(assess(c_i)) \qquad (8)$$

where m is the number of c_i's relevant child nodes.

In addition to the above, the final RV score is obtained by applying a normalisation function, which ensures that the total score for any group of descendant nodes of an ideal result element cannot exceed the score achievable if retrieving the ideal node itself. For example, in Figure 1 the two ideal result nodes for the quantisation function *sog* are `sec[4]` and `sec[6]`. Since these results represent the best nodes for the user, a system returning these should be ranked above others. However, if another system retrieved all the leaf nodes, it may achieve a better overall score if the total RV score for these nodes exceeds that of the ideal nodes. The following normalisation function safeguards against this by ensuring that for any $c_j \in S$:

$$\sum_{c \in S} rv(c) \leq rv(c_{ideal}) \qquad (9)$$

where S is the set of retrieved descendant nodes of the ideal node and where c_{ideal} is the ideal node that is on the same relevant path as c_j.

A.4 T$_2$I

T$_2$I is based on an alternative definition of correct results. The main idea is that a user merely needs an entry-point into the document that is 'close' to relevant information. Taking this view, a retrieval system produces a ranked list of entry points. The user starts reading the retrieved article from the suggested entry point, giving up when no relevant information is found for some number of words or sentences. So, the user processes the retrieved information until his or her *tolerance to irrelevance* (T$_2$I) has been reached, at which point the user proceeds to the next system result.

This discourages systems from returning fragments that are too large, since if the entry-point is too far away from the relevant reference component, the user's tolerance to irrelevance will have been exhausted before the relevant information has been reached. The problem with multiple system results intersecting the same reference component is eliminated by extending the definition of irrelevance, according to which a previously seen reference fragment is no longer considered relevant.

T_2I variants of three existing evaluation metrics for system performance are given in [3]. Their common underlying principle is that retrieval systems are ranked on their ability to maximise the number of relevant fragments shown to the user while minimising the amount of user effort wasted on irrelevant information. The tolerance to irrelevance is expressed by a single parameter, τ_{NR}, that represents the maximum amount of non-relevant text the user is expected to read before giving up. The length of retrieved relevant components is ignored, assuming that each result has equal value to the user.

A.5 PRUM

The PRUM (Precision-Recall with User Modelling) metric [17] is an extension of the probabilistic precision recall proposed by Raghavan. While the latter supposes a simple user model, where the user consults retrieved elements (elements returned by the retrieval system) independently, PRUM "allow" the user to consult the context of retrieved elements: For each element in the list returned by the retrieval system, the user consults the context of the element. In the context of XML Retrieval, this context is possibly made of the siblings, ancestors and descendants of a retrieved element. Note that this behaviour is defined stochastically, that is we only know that the user has seen a context element with a given probability. For instance, if the user consults a section in the retrieved list, we know that the user has seen this section with a probability 1, and that (s)he has seen also its first paragraph with a probability .95, etc.

Like some other metrics (e.g. XCG), PRUM supposes a set of ideal results, which are the most appropriate non-overlapping elements of the XML database to return to the user. The PRUM metric is then defined as the probability that the user sees a newly relevant element when (s)he consults the context of a retrieved element, knowing that the user wants to see a given amount of relevant units:

$$PRUM(l) = P(Lur|Retr, L = l, Q = q) \tag{10}$$

where l is the recall level between 0 and 1, q the topic for which PRUM is computed; Retr is the event "the element is in the list consulted by the user" while searching for I% of the relevant units, and Lur is the event "the element Leads to an Unseen Relevant unit".