FORMULATING MEASURES FOR STRUCTURED DOCUMENT RETRIEVAL
SEARCH TASKS USING EXTENDED STRUCTURAL RELEVANCE

by

Mir Sadek Ali

A thesis submitted in conformity with the requirements
for the degree of Doctor of Philosophy
Graduate Department of Mechanical and Industrial Engineering
University of Toronto

Copyright © 2012 by Mir Sadek Ali
Abstract

Formulating Measures for Structured Document Retrieval Search Tasks using Extended Structural Relevance

Mir Sadek Ali
Doctor of Philosophy
Graduate Department of Mechanical and Industrial Engineering
University of Toronto
2012

Structured document retrieval (SDR) systems minimize the effort users spend to locate relevant information by retrieving sub-documents (i.e., parts of, as opposed to entire, documents) to focus the user’s attention on the relevant parts of a retrieved document. SDR search tasks are differentiated by the multiplicity of ways that users prefer to spend effort and gain relevant information in SDR. The sub-document retrieval paradigm has required researchers to undertake costly user studies to validate whether new IR measures, based on gain and effort, accurately capture IR performance.

We propose the Extended Structural Relevance (ESR) framework as a way, akin to classical set-based measures, to formulate SDR measures that share the common basis of our proposed pillars of SDR evaluation: relevance, navigation and redundancy. Our experimental results show how ESR provides a flexible way to formulate measures, and addresses the challenge of testing measures across related search tasks by replacing costly user studies with low-cost simulation.
Acknowledgements

This work is dedicated to my father Mir Maswood Ali (PBUH) and my son Mir Omar Ali (born Sep. 19, 2011). I must first thank my wife, Muna Ali, for her unconditional support; and, my mother, Suraiya W. Ali, for her constant prayers and encouragement. I thank my supervisor Mariano Consens for his perserverance and confidence; my co-author Mounia Lalmas for her insights and guidance in developing the work; Adnan Ali and Ayesha Ali for their many comments and insights over the years; committee member J. C. Beck for his thorough review and help in making the necessary corrections in the maths and verbiage; and, my labmate Shahan Katchadourian for giving time to proofread this work and to critique my practice presentations.
5  Extended Structural Relevance

5.1  Extended Structural Relevance Framework .................................................. 89
5.2  ESR Evaluation Measures .............................................................. 95
  5.2.1  Structural Relevance ................................................................. 97
  5.2.2  Highlighting XML Retrieval Evaluation ......................................... 99
  5.2.3  Extended Cumulated Gain .......................................................... 101
  5.2.4  Precision-Recall with User Modeling (PRUM) ............................... 104
5.3  Calculating ESR Measures ................................................................. 108
5.4  Additional Measures ................................................................. 112
5.5  Summary ................................................................. 113

6  Compatible Measures

6.1  Scenario Stability Analysis .............................................................. 116
  6.1.1  Properties of SDR Measures ......................................................... 117
  6.1.2  Scenarios in SDR ................................................................. 123
  6.1.3  Model Collection and Randomized Outputs .................................. 128
6.2  Comparing ESR to INEX Measures .................................................. 130
  6.2.1  Experimental Measures .......................................................... 130
  6.2.2  Results of Bias Comparison ....................................................... 132
  6.2.3  Results of Range Comparison ..................................................... 133
  6.2.4  Results of Relative Performance Comparison ................................ 134
  6.2.5  Results of Overall Performance Comparison .................................. 135
  6.2.6  Summary ......................................................... 136
6.3  INEX System Rankings Using ESR Measures ....................................... 138
  6.3.1  Experimental Setup .............................................................. 138
  6.3.2  INEX 2006 Focused Task - nXCG and iP ...................................... 139
  6.3.3  INEX 2006 Best In Context Task - EPRUM .................................... 141
  6.3.4  INEX 2007 Focused Task - iP .................................................... 142
List of Figures

1.1 (a) Extract from a book in XML markup, (b) Tree structure of book . . . 3
1.2 Different SDR approaches; (a) document/element/passage, (b) tree . . . 4
1.3 Example of redundancy. . . . . . . . . . . . . . . . . . . . . . . . . . . . 5

3.1 Tree structure of book in Figure 1.1 . . . . . . . . . . . . . . . . . . . . . 40
3.2 (a) Tree structure of an article, (b) Possible navigations in document, and (c) User navigation graph based on partition . . . . . . . . . . . . . . . . . . . . . . . . . . . 49
3.3 Examples of routes navigated . . . . . . . . . . . . . . . . . . . . . . . . . 49
3.4 Retrieved Book XML subtrees . . . . . . . . . . . . . . . . . . . . . . . . 65

4.1 Tree structure of book in Figure 1.1 . . . . . . . . . . . . . . . . . . . . . 72
4.2 $p^*$ (left) and $p^*|c$ (right) summaries . . . . . . . . . . . . . . . . . . 73
4.3 User navigation graph for Interactive Track at INEX 2006. . . . . . . . . 77
4.4 Fidelity of SRP in Thorough Task . . . . . . . . . . . . . . . . . . . . . . 86

6.1 Proposed model collection . . . . . . . . . . . . . . . . . . . . . . . . . . 128
List of Tables

<table>
<thead>
<tr>
<th>Table</th>
<th>Description</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>3.1</td>
<td>Cases of gain for tree retrieval</td>
<td>44</td>
</tr>
<tr>
<td>3.2</td>
<td>Elementary user navigation weights with probabilities (in parentheses)</td>
<td>51</td>
</tr>
<tr>
<td>3.3</td>
<td>Summary model weights and navigation probabilities (in parentheses) for</td>
<td>52</td>
</tr>
<tr>
<td></td>
<td>nodes S1, S2, and S3 in Figure 3.2(b).</td>
<td></td>
</tr>
<tr>
<td>3.4</td>
<td>Summary of statistical model for SDR gain</td>
<td>64</td>
</tr>
<tr>
<td>3.5</td>
<td>Relevance value of assessments rel(e), for tree e in A</td>
<td>66</td>
</tr>
<tr>
<td>3.6</td>
<td>User navigation (\tilde{p}(e_i; e_j))</td>
<td>67</td>
</tr>
<tr>
<td>3.7</td>
<td>Three example system outputs</td>
<td>68</td>
</tr>
<tr>
<td>4.1</td>
<td>Steady-state probabilities in (p^*) summary weighted by extent size of</td>
<td>74</td>
</tr>
<tr>
<td></td>
<td>children</td>
<td></td>
</tr>
<tr>
<td>4.2</td>
<td>Steady-state probabilities in (p^*</td>
<td>c) summary weighted by character length.</td>
</tr>
<tr>
<td>4.3</td>
<td>Number of visits (mean time spent) from Interactive Track at INEX 2006</td>
<td>78</td>
</tr>
<tr>
<td>4.4</td>
<td>Navigation models in Interactive Track study</td>
<td>79</td>
</tr>
<tr>
<td>4.5</td>
<td>Correlation p-values of User Navigation Models and Summary Navigation Models</td>
<td>80</td>
</tr>
<tr>
<td></td>
<td>for k = 10 (and k = 50).</td>
<td></td>
</tr>
<tr>
<td>4.6</td>
<td>Evaluation (p-values) of SRP across tasks</td>
<td>84</td>
</tr>
<tr>
<td>5.1</td>
<td>Summary of statistical model for ESR evaluation</td>
<td>93</td>
</tr>
<tr>
<td>5.2</td>
<td>Hits, near-misses, and misses using binary relevance (relevance by length).</td>
<td>96</td>
</tr>
<tr>
<td>5.3</td>
<td>Recall-base using binary relevance (relevance by length).</td>
<td>96</td>
</tr>
<tr>
<td>5.4</td>
<td>Example of PRUM for (P(e_1 \sim e_4) = 0.2)</td>
<td>106</td>
</tr>
</tbody>
</table>
## Glossary

<table>
<thead>
<tr>
<th>Term</th>
<th>Definition</th>
</tr>
</thead>
<tbody>
<tr>
<td>accurate</td>
<td>Property of a measure that specifies how reliably it scores what it is designed to measure.</td>
</tr>
<tr>
<td></td>
<td>See also fidelity, stable, validation, 15</td>
</tr>
<tr>
<td>consult</td>
<td>User action to retrieve a result from the output for the purpose of visiting the result. See visit, navigate, 39</td>
</tr>
<tr>
<td>document</td>
<td>Basic unit of textual records contained in a collection, e.g. XML document. See also element, node, passage, tree, 2</td>
</tr>
<tr>
<td>element</td>
<td>A logical structural component of XML documents, also called an XML tag. See also document, node, passage, tree, 2</td>
</tr>
<tr>
<td>fidelity</td>
<td>Property of a measure the specifies how reliably it scores better systems higher and worse systems lower. See also accurate, stable, validation, 15</td>
</tr>
</tbody>
</table>
hit  A relevant sub-document in the system output. See also miss, near-miss, 9,44

miss [1]  A relevant sub-document not in the system output. See also hit, near-miss, 9,44

miss [2]  Unrealized gain from a relevant sub-document not seen because it is neither in the output nor navigated to by the user. See also hit, near-miss, 44

navigate  User action to browse out of a retrieved sub-document to seek relevant information by visiting a different sub-document. See consult, visit, 39

near-miss  A relevant sub-document not in the system output that is seen by navigating to it. See also hit, miss, 44

node  An element in a tree that represents an XML element. See also document, element, passage, tree, 3,38

overlap [1]  The same fixed region of text in two or more sub-documents, 60
<table>
<thead>
<tr>
<th>Term</th>
<th>Definition</th>
</tr>
</thead>
<tbody>
<tr>
<td>overlap [2]</td>
<td>The text encapsulated by descendant elements in an XML tree, 60</td>
</tr>
<tr>
<td>overlap [3]</td>
<td>A user preference to retrieve sub-documents that do not result in user’s seeing the same relevant text by consulting the output, 60</td>
</tr>
<tr>
<td>relevance</td>
<td>A judgment made by a human assessor on whether the subject matter of some information is meaningful to a given information need. See also relevance value, 44</td>
</tr>
<tr>
<td>relevance value</td>
<td>A scalar valued revaluation of relevance judgments to account for wasted effort. See also relevance, 44</td>
</tr>
<tr>
<td>see</td>
<td>See definition for visit, 38</td>
</tr>
<tr>
<td>stable</td>
<td>Property of a measure that specifies whether it reliably distinguishes the binary preference relationship of a user judging whether one system performs better (more effective) than another. See also accurate, fidelity, validation, 15</td>
</tr>
</tbody>
</table>
sub-document A partial record of a textual record whose encoding delineates a fixed region(s) of the text in the record and is often chosen based on how the user prefers to interact with the system to locate relevant information. See also document, element, node, passage, tree, 2

tree Logical representation of a document where nodes in a tree represent elements in the document. See also document, element, node, passage, 37

validation An experimental result for a measure to show that it is stable by either (1) testing fidelity and accuracy using outputs that are biased to predictably improve or degrade performance of systems, or (2) comparing system rankings from our experimental measure to rankings from a known stable measure. See also accurate, fidelity, stable, 15

visit User interaction with system to see (i.e. read) text in sub-document. Seeing of text only occurs during a visit. See also consult, navigate, 39
Chapter 1

Introduction

Modern IR systems are moving toward a sub-document retrieval paradigm where IR systems retrieve the relevant parts of documents as opposed to the entire document. In Newby’s prescient article in 2000 [85], the author theorized that retrieving sub-documents would become a practical necessity for improving end-user satisfaction when searching the massive, and growing, collection of documents and data available today on the web. In 2002, this retrieval paradigm became first codified and named structured document retrieval (SDR). Newby’s contentions in improving end-user satisfaction were later confirmed across numerous user studies for retrieval tasks such as passage retrieval for question-answer systems [29], XML element retrieval [121], and personalized web retrieval [83]. SDR represents a broad (and growing) range of search tasks.

Our work considers the problem of developing a common set of stable IR measures to evaluate performance across the range of tasks studied in SDR research. A measure is stable if it reliably evaluates system performance the same as a human judge in terms of how well the system serves the user to satisfy the user’s information need. This is an active area of research. Currently, each task in SDR is evaluated using a measure that has been specifically developed and validated as stable for the task. However, each measure captures differently how users judge the performance of an IR system. As a
result, it is difficult to compare scores from different measures for the same SDR system.

Our solution is the Extended Structural Relevance (ESR) framework which defines a common basis upon which to evaluate any ad-hoc SDR search task; i.e. tasks where a user seeks verbatim answers from a fixed collection. We show how stable SDR measures can be derived from the ESR framework by proposing measures in ESR that match our current SDR measures. The key benefit is that our ESR measures use a common model of how users judge performance, a common set of measure parameters, and a common data set, i.e. a test collection, upon which to evaluate systems.

The outline of this chapter is as follows. Section 1.1 illustrates structured document retrieval. Section 1.2 is a review of the basic concepts in IR that are used throughout this work. Section 1.3 describes our methodology for experimental evaluation of SDR. In Section 1.4 we introduce our problem. In Section 1.5 we enumerate the major research contributions of this work. Finally, in Section 1.6 we present the outline of this thesis.

1.1 Structured Document Retrieval

Structured document retrieval (SDR) systems exploit textual markup to retrieve sub-documents (i.e., parts of, as opposed to entire, documents) from structured document collections. An SDR system is a type of focused system \[114\]. A focused system retrieves answers that minimize the effort that a user spends to locate relevant information in documents. Examples of SDR search tasks include book search (e.g., retrieve chapters), semantic search (e.g., retrieve sets of linked semantic associations from resource description framework (RDF) documents), and enhanced web search (e.g., retrieve hypertext markup language (HTML) snippets annotated with links to related pages).

Recent research in SDR has flourished due to the widespread adoption of XML (eXtensible Mark-up Language) as a means to format documents. XML is used to explicitly delineate the logical structure of a document as a tree of documents elements. SDR is
Chapter 1. Introduction

Figure 1.1: (a) Extract from a book in XML markup, (b) Tree structure of book

particularly advantageous when dealing with long documents, or documents covering a wide variety of topics. In the literature, three types of structured document collections have been considered: (i) Documents encoded using a schema, such as XML collections like Wikipedia [37], Microsoft Books [62], and IEEE Articles [15]; (ii) Documents that reference each other via hyperlinks, such as web-based HTML documents [48]; and (iii) Documents that contain linked data, such as semantic data in RDF documents [14].

SDR systems exploit the structure of information in two ways to reduce the effort users spend to locate relevant information. First, referred to as a structural hint [109], a system ranks a sub-document based on whether its encoding can help the end-user to more easily locate relevant information. Second, referred to as a structural constraint [116], a user may direct the system to refine, or specialize, search to sub-documents with a desired encoding (using a query language such as NEXI [118] or XQueryFT [12]). For instance, the paragraphs in a section of an article that contain the word “Computer” can be queried in NEXI with //article//sec[about(/p, Computer)] and equivalently in XQueryFT with for $p$ in doc("article.xml")/article/sec/p where contains($p, ‘Computer’’) return $p.
Several types of sub-document results exist in SDR, each of them “modelling” how users locate relevant information. We illustrate these through examples. Let a collection contain the extract of the book (formatted in XML) shown in Figure 1.1(a). The document structure of the book is, in this case, a tree, which is shown in Figure 1.1(b); the tags have been abbreviated as follows: book (bk), front matter (fm), body (bd), description (d), name (n), meta (m), and chapter (c). The line numbers of elements shown in Figure 1.1(a) correspond to the node ID of each corresponding node in Figure 1.1(b).

Consider the query “ship captain in Moby Dick”. The query matches terms in different parts of the book extract; specifically, node 4 (match on “Moby Dick”), node 15 (on “ship”) and node 16 (on “captain”) in Figure 1.1(b). For a document retrieval task, the system returns the root node, which models the user accessing the whole book. For a focused retrieval task (defined by Kazai, Lalmas & Rölleke [64]), as illustrated in Figure 1.2(a), the system may return nodes (as elements or text passages\(^1\)) at separate ranks, which provides the user with focused information at the cost of having to examine results at multiple rank positions. Finally, in Figure 1.2(b), the system returns subtrees at separate ranks (the first rank corresponds to a subtree taken from the book extract),

\(^1\)In this work, nodes correspond to passages which match exactly with the boundaries of content in document nodes, which is sufficient to evaluate our proposed ESR framework for a number of INEX tasks (Section 6.3).
which provides the user with a single result per book that directs the user to the relevant parts of each book.

An important difference between SDR and much of the work in classical IR is that SDR systems support users who may navigate to relevant information. The user experiences gain by seeing relevant information either in retrieved sub-documents or by navigating from retrieved sub-documents. Let $e_i$ denote the element represented by node $i$ in the document shown in Figure 1.1 (for instance, the element $e_{15}$ is node 15). In Figure 1.3, we show an output $R = e_4, e_{15}, e_{16}$ of elements from the document shown in Figure 1.1. Each of the retrieved elements contain relevant text. Consider the user who navigates from $e_4$ to both $e_{15}$ and $e_{16}$ (as shown in Figure 1.3). At rank 1, the user would experience maximum gain by seeing all of the relevant information in the document. However, at ranks 2 and 3, the retrieved elements would be redundant to the user because the elements had already been seen by navigating from the element $e_4$ at rank 1. Thus, in contrast to classical IR where performance is largely based on relevance, the basis of evaluation in SDR depends on the relationship between relevance, navigation and redundancy. We refer to relevance, navigation and redundancy as the pillars of SDR evaluation, and understanding these concepts is critical to understanding our work.

The relationship between our pillars and performance has been explored in the lit-
Chapter 1. Introduction

In de Vries, Kazai & Lalmas [36], the authors posit that, by seeing relevant information redundantly, users experience lower IR performance. In SDR, users see relevant information redundantly because of what Ma & Schewe [76] call information fragmentation, i.e. documents are fragmented into sub-documents. As illustrated above in Figure 1.3 SDR users may see information redundantly when retrieved sub-documents overlap (as shown in Hammer-Aebi et al. [47] and Kazai [63]) or by navigating between sub-documents (see Piwowarski, Gallinari & Dupret [94]).

Overlap, in our work, is a special case of redundancy. Our definition combines definitions from Clarke [27] and de Vries, Kazai & Lalmas [36]. Clarke [27] defines overlap in terms of document structure where retrieved nodes are from the same document on the same branch in an XML document tree. de Vries, Kazai & Lalmas [36] define overlap as a user preference where users waste effort when the number of times that the same information is seen exceeds the number of times that the user is willing to see the same information. In our work, overlap occurs when two or more sub-documents contain the same region(s) of text and the user considers their gain in relevant information diminished because the effort spent to see the information was wasted.

Early IR studies, such as in Salton, Allan & Buckley [106], investigate information fragmentation in passage retrieval in full-text search, where the system retrieves non-overlapping text excerpts of varying size in response to statements of user interest. However, the authors do not consider the effect of redundancy (vis à vis user navigation and overlap) on IR performance. In Keskustalo, Järvelin & Pirkola [67], the authors show how navigation can be used to refine search results with relevance feedback by simulating how users prefer to spend effort reading documents. Studies in web retrieval demonstrate how performance is improved by ranking results based on predictions of user navigation within web pages (either modelled from user clicks [41] or, more explicitly based on tracking navigation via user-system interaction [1]). Drosou & Pitoura [38] consider redundancy in search result diversification stemming from information duplica-
tion, i.e. the same information appears in more than one document. Bernstein & Zobel [18] propose how information duplication in documents can be inferred from term frequencies and relevance judgments. Based on information duplication, researchers, such as Clarke et al. [30] and Agrawal et al. [2], demonstrate how results can be diversified to minimize the risk of dissatisfaction of the average user (due to redundancy in the content of retrieved documents). We will revisit diversification in Section 2.5 to more deeply explore how it relates to the present work, however we note that diversification research has largely focused on the IR ranking problem whereas our work focuses on the IR evaluation problem.

Navigation can also improve performance because navigation allows the user to locate relevant information not explicitly retrieved by the system as shown in Piwowarski, Gallinari & Dupret [94]. Newby [84] originally proposed user navigation (in document retrieval) as a fundamental concept for evaluating information retrieval. The work by Piwowarski, Gallinari & Dupret [94] defines a probabilistic model of navigation (akin to navigation in our example in Figure 1.3) but limits their evaluation measure, PRUM , to element retrieval systems. Our work goes beyond existing SDR measures (reviewed in Section 2.1), such as XCG, PRUM, and HiXEval, that are limited today to document, passage, and element retrieval paradigms.

This thesis differs from previous works in two key ways. First, we distinguish our work from work in diversification, such as presented in Clarke et al. [30] and Agrawal et al. [2], by our research areas of IR evaluation (as opposed to ranking) and redundancy from information fragmentation (as opposed to information duplication). Second, we consider evaluation, currently not possible with existing SDR measures (such as those presented in Section 2.1), that goes beyond document, element and passage retrieval.
1.2 Basic Concepts in IR

We now briefly review the basic concepts in information retrieval that are used throughout this work. This section is meant for readers not familiar with information retrieval. We recommend that the interested reader refer to Baeza-Yates & Ribiero-Neto [13] and Büttcher, Clarke & Cormack [23] for an in-depth treatment of the concepts presented here.

*Information retrieval* (IR) is the science of computerized systems that search *collections* of electronic information (in the form of documents or records) in response to a machine-interpretable *query*. We refer to the person using the system as the *user* and the computerized system as simply the *system*. There are many types of IR systems. We limit our investigation to ad-hoc search which is defined as a user who seeks verbatim answers from a fixed collection. A *topic* is the subject matter or question for which the user seeks answers from the collection. A *query* is a machine-interpretable representation of the topic, e.g. an online search query is typically composed of a bag of keywords. To search, the user *poses* their topic to the system as a machine interpretable *query*. The system returns search results as a *ranked list* ordered by relevance to the topic (called the probability ranking principle by Robertson [100]). We refer to the ranked list as an *output* of the system and the answers in the output as the *results*.

Our work here is in the area of *evaluation* of IR effectiveness. *Effectiveness* is a measure of performance in terms of how effectively the system supports the user’s *information seeking behaviour*. This is in contrast to *efficiency* which is a measure of performance in terms of the speed of the computerized system. In Case [25](p. 5), the author defines information seeking as “a conscious effort to acquire information in response to a need or gap in your knowledge”. Case then defines an *information need* as “a recognition that your knowledge is inadequate to satisfy a goal that you have”. The user *tasks* the system. An IR *search task* is a specification of how the system is expected to serve the user in obtaining the answers that *satisfy the user’s information need*. The most general search
task in classical IR is for the system to find and rank the documents in the collection that are relevant to the user’s topic.

Relevance is the basis of classical IR evaluation because a document retrieval system is tasked with finding and ranking the documents in the collection that are the most relevant to the user’s query. Relevant documents returned in the output are called hits. The relevant documents not in the output are called misses.

A user model describes how a user prefers to interact with the system to satisfy his/her information need. In classical document retrieval, the typical user model is as follows. The user retrieves documents from the collection by successively consulting the output in rank order (from the highest ranked document to the lowest). The user consults the output until either the user’s information need is satisfied or the user exhausts the results in the output. A user that abandons the system is modelled by limiting the length of the output to a given top-\(k\), e.g. the effectiveness of a commercial web search engine is often evaluated with ranked lists of length 10 or less because users typically seek answers from only the first page of search results.

Experimentation in classical IR follows the Cranfield method described by Cleverdon [32] (originally proposed in Cleverdon [31]). Cranfield represents the experimental foundation for most modern research in IR systems. This methodology was defined through a series of experiments conducted by a team of researchers lead by Cyril Cleverdon (at Cranfield University in Cranfield, UK) in the 1960’s to create the first information retrieval test collections. The test collection serves as a standard for measuring system performance. The Cranfield method utilizes precision and recall to capture how users of document retrieval systems perceive system performance. Most search tasks in classical IR can be tested with the Cranfield method.

The search corpus is a fixed collection of documents. However, in the literature, the term collection (or test collection ) denotes not only the corpus, but also, a set of topics or queries (called the topics), and a set of relevance assessments for each query (called
the assessments). For a given topic, assessments are a gold standard for evaluating the performance of a system. An IR measure evaluates performance by comparing the system output to the assessments. Assessments are determined in a user study where human assessors judge whether candidate search results are germane to answering a given topic. The experimental conditions of the user study are called the assessment methodology. The methodology specifies how researchers collect and then combine judgments from study participants to determine judgments of relevance for each candidate search result.

In developing an assessment methodology, key challenges for researchers include how to obtain (1) assessor agreement because users often disagree on what is relevant; and, (2) complete assessments because it can be costly to judge all of the relevant information in a large collection.

The two most widely known IR measures are precision and recall. Precision is the proportion of hits in the output. Recall is the proportion of the hits in the collection that have been output. They are often used together (called precision-recall) where we calculate precision at the rank cut-off where the user experiences their desired recall. Desired recall models the user’s information need being satisfied by the user retrieving a portion of the relevant information in the collection. It is specified in the user model as a fixed recall point.

Precision-recall allows us to compare systems in terms of precision under the condition that the user has satisfied their information need. However, comparing systems using precision-recall is complicated to understand because we require a given desired recall. A better approach is to evaluate average precision (AP) across a range of recall-points, e.g. 11-point AP evaluated across recall points at 0.1 intervals in [0, 1]. To compare systems across a range of topics, we evaluate the mean average precision (MAP) which is AP per system averaged across topics. MAP scores are used for system ranking where the performance of systems are ordered from best to worst (matching how users judge performance). Buckley & Voorhees [21] defines MAP as a stable measure because it
is a reliable measure for capturing system ranking in the same order as users judge performance.

In summary, to experimentally compare systems that implement different search strategies, we use the Cranfield method to define a search task, user model and test collection. First, we define the two key concepts of search task and user model based on experimental observations of users interacting with IR systems. Then, we define an assessment methodology that captures how users judge relevance in our task. Next, we assemble our test collection. It consists of a collection of documents, a set of queries, and, for each query, assessments that identify the relevant documents in the collection. Finally, we develop measures, based on the search task and user model, that compare our system outputs to the gold standard defined by the test collection. The most often used measure is MAP which is calculated using our more basic measures of precision and recall. We determine the best strategy(ies) by ranking our systems based on MAP scores.

The Cranfield method and MAP have been widely adopted by IR researchers. In Buckley & Voorhees [21], precision and recall are found to be stable across most document retrieval tasks. In Bollman [20], the author explains how well-known classical measures (such as precision, recall, and fallout) are consistent because they share the same common basis of evaluation, i.e. relevance. Precision and recall are compatible with each other because they are both stable for the same set of search tasks, i.e. all document retrieval tasks. Thus, we justify the approach of combining precision and recall in MAP for any document retrieval task because precision and recall are stable, consistent and compatible.
1.3 Experimental Evaluation of SDR

The experimental results in this work involve validating the stability of experimental measures. However, as shown by Hiemstra & Mihajlovic [50], the classical test collection and the Cranfield method, described in the last section, are not sufficient to validate measures in SDR. They show how SDR evaluation differs in two key ways. First, the classical IR user model does not consider navigation in SDR. Second, classical IR relevance assessment is not sufficient to characterize how redundancy may affect system performance.

Let us first state the basic difference between the classical and SDR user models. In both SDR and classical IR, the user consults the output to retrieve results until either the user has satisfied their information need or has exhausted the results in the output. However, in SDR, the user may navigate from the retrieved sub-document to seek relevant information in other sub-documents in the collection. Consultation and navigation are different types of effort that users may spend to seek for relevant information. Effort is defined as how the user interacts with the system. In classical IR, the user is limited to spending effort in consultation.

A sub-document retrieval paradigm means that users may see the same relevant information more than once because answers may contain overlapping text or because navigation to the same information is possible. In our user model, we say that users experience redundancy when they see the same relevant information more than once. We account for redundancy as a type of wasted effort. The user experiences best performance when they do not waste effort and they see enough relevant information to satisfy their information need.

Navigation is not considered in classical IR evaluation but is an important factor in evaluating SDR. In the literature, there are heuristic [59, 89] and probabilistic [94] navigation models. Heuristic models propose rules that capture the retrieval conditions where users experience redundancy (such as overlap [63]). Probabilistic models define
navigation within a sub-document space weighted by observations of effort from user studies. Our work employs probabilistic models.

Our model is derived by observing users interacting with the system to gauge the users’ *willingness to spend effort* by navigating to relevant information. The user’s willingness stems from cognitive perceptions of enticements (such as likelihood of finding more relevant information), expertise (such as knowledge of where to look in a document for relevant information), or need (such as the derived benefit or utility of the answer). The deep concepts that explain how and why users navigate is a branch of information seeking (IS) described in Case [25] that is beyond the scope of our work. In our work, we experimentally determine navigation in terms of clearly observable events such as clicks, time spent, or scrolling events. However, in IR, effort can also be inferred from cognitive analysis based on asking a user in situ whether he or she expects by taking a given action to gain relevant information [66] or avoid non-relevant information [36].

Now, let us consider why classical IR relevance is not sufficient to characterize SDR performance. In classical IR, users satisfy their information need by retrieving hits (retrieved relevant documents) from a ranked list of documents. The effectiveness of the system is then scored largely dependent on the number of relevant documents in the output. In SDR, users satisfy their information need by retrieving *hits* (seen, retrieved, relevant sub-documents) and navigating to *near-misses* (seen, not retrieved, relevant sub-documents). Moreover, effectiveness is scored dependent on matching the encoding of retrieved sub-documents to the encoding of the relevant sub-documents judged by assessors. For instance, in [26], Clarke demonstrates how assessing passages of text can be used to evaluate element retrieval by matching the text boundaries of elements within assessed passages. However, vice versa, assessed elements cannot always be used in this same way to evaluate passage retrieval. Unlike classical IR, SDR performance must consider user preferences in sub-document encodings and navigation.

Piwowarski, Trotman & Lalmas [96] noted that a significant challenge in SDR is
to obtain assessor agreement for judging relevance given user preferences. Their work defines an assessment methodology (which we also use in this work) for collecting relevance judgments based on human judges who highlight relevant passages of text in retrieved (pooled) documents. Highlighted passages of text can be interpreted as either binary relevance where highlighted regions are relevant and others not; graded relevance where judged regions are scored according to assessors’ agreement of relevance \cite{66}; or, relevant text length which is the number of characters contained in highlighted regions \cite{90}. However, our relevance assessments cannot be interpreted as multi-level (or multi-dimensional) relevance where an assessor’s judgment consists of two or more graded aspects of relevance \cite{86}.

In SDR, to calculate gain, we do not use relevance judgments directly. We calculate the user’s gain in relevant information based on a function called relevance value \cite{59} that revalues relevance judgments to account for user preferences that are not explicitly assessed. In SDR, there exists a crucial distinction between relevance (judgments) and relevance value (revalued judgments).

Other examples of basic user preferences in SDR that do not apply to classical IR include the following. Users exploit knowledge of a schema by posing queries that include structural context to find sub-documents that either contain or are near relevant information \cite{118,69}. For long documents, users expect relevant information to be broken up into regions of the document that are about a specific topic \cite{95,86}. For short documents, users prefer finding the best single point in the document from which to see the relevant part(s) of the document \cite{99,54}. Typically, users do not tolerate systems that output sub-documents that overlap \cite{90}. Inter-navigable sub-documents in the output should be logically ordered \cite{120} (e.g. in-context ranking orders sub-documents as they appear in documents \cite{55} or ideal ranking orders sub-documents as judged by assessors \cite{59}). Users may prefer answers encoded as XML elements \cite{19}, passages of text \cite{112}, or trees of information (see Chapter 3).
Our experimental method to validate the stability of measures (for results in Section 6.3 and Chapter 4) is taken from the methodology defined in Buckley & Voorhees [21]. The authors define a stable measure as one that reliably captures system ranking; i.e. the binary preference relationship of a user judging whether one system performs better (more effective) than another. We use the notation \( A \succ B \) to denote the binary preference relationship of System A being better than System B which implies that the evaluation score of A is higher than the score of B. An accurate measure reliably scores what it is designed to measure. A measure with good fidelity reliably scores better systems higher and worse systems lower. Based on Voorhees & Buckley [21], we validate that an experimental measure is stable (1) by testing fidelity and accuracy using outputs that are biased to predictably improve or degrade performance of systems, or (2) by comparing system rankings from our experimental measure to rankings from a known stable measure. In Chapter 4, we validate an experimental measure by testing its accuracy and fidelity. In Section 6.3, we validate our experimental measures by comparing them with known stable measures.

We complete our discussion of our experimental method by describing how we compare system rankings using Spearman’s Rho.\(^2\) Spearman’s Rho (\( \rho \)) indicates whether two separate rankings are positively (\( \rho > 0 \)) or negatively (\( \rho < 0 \)) ordered. If \( \rho = 1 \), then the ranked lists are in exactly the same order. If \( \rho = -1 \), then the ranked lists are in exactly opposite order. The p-value is the probability that the compared rankings are not correlated. If a p-value is less than a given threshold (typically less than 0.1) then two measures are considered correlated in terms of how they order systems. So, in

---

\(^2\)Kendall’s Tau would also be an appropriate measure for comparing systems however we believe that Spearman’s Rho is better suited for our work because of both the large number of systems (critical values for Kendall’s Tau may not be well-suited to set sizes greater than 20) and the wide range of scores in our test systems [108]. Kendall’s Tau (\( \tau \)) indicates whether two separate rankings, as generated in our case by two evaluation measures (an ESR measure and an original INEX measure), are positively (\( \tau > 0 \)) or negatively (\( \tau < 0 \)) ordered. The p-value is the probability that the compared rankings are not correlated. If a p-value is less than 0.05 then the two measures are correlated in terms of how they order systems.
comparing system rankings, the rankings will be either positively correlated, negatively correlated, or not correlated. We conclude that a measure is stable if its system rankings are positively correlated. This completes our description of our experimental method.

1.4 Problem Statement

In the SDR literature, we found numerous works that cited the lack of a stable measure as a barrier to continuing research. Researchers have identified three key challenges with SDR measure stability. In a series of short papers by Trotman & Lalmas in 2006 [111, 116, 117], the authors explored the challenge of developing measures that capture the numerous (and sometimes opposing) ways to judge effectiveness in SDR. In 2007, Kazai [58] showed that some SDR measures were unstable because they were overly sensitive to the assessment methodology used to judge system performance. In 2008, Piwowarski, Trotman & Lalmas [96] addressed problems with assessment methodology however their work brought to light the problem that current SDR measures do not share consistent (or sometimes even comparable) definitions of what is regarded as good system performance. As a result, no current SDR measure is stable across all tasks in the SDR sub-tasks of document, element and passage retrieval. Moreover many SDR tasks outside of these tasks remain unexplored. One of the main goals of our work is to provide a way to formulate measures that avoids these pitfalls.

We argue, especially for new tasks, that these problems impel researchers to avoid evaluation because of either (a) the challenges in proposing a new measure, (b) the cost borne in user studies to validate the stability of a measure, or (c) both. Our work addresses three key problems in the state-of-the-art: stability (measures do not share a common basis of evaluation and are overly sensitive to the assessment methodology [60, 105, 88]); consistency (different, non-comparable calculations of relevance, user navigation and redundancy [29, 36, 50, 26]); and compatibility (stable for different sets of
search tasks \cite{116, 111, 112}.

In Chapter 3, our solution begins by proposing our pillars of SDR evaluation to evaluate SDR tasks on the common basis of relevance, navigation and redundancy. In Chapter 4, we experimentally validate the stability of the Structural Relevance (SR) measure which is based on our pillars. Then, in Chapter 5, we use our pillars to define the Extended Structural Relevance (ESR) framework which provides a way to re-formulate current SDR measures using consistent measure parameters akin to how precision and recall are defined in the Cranfield method. Finally, in Chapter 6, we propose a low-cost simulation, called Scenario Stability Analysis, to test whether our ESR measures are compatible across ad-hoc search tasks.

We have limited this work as follows. We consider a user model constrained to our pillars of SDR evaluation i.e. relevance, navigation and redundancy. Our work is limited to ad-hoc tasks in SDR which represents a broad range of tasks including numerous distinct sub-tasks in element, passage, and tree retrieval. Our work exploits how trees can represent documents, elements, and passages. We argue that our pillars are sufficient to capture performance across tasks in tree, document, element, and passage retrieval. Our experimental results are largely in element and passage retrieval which are the most studied tasks in SDR. Finally, we limit the comparison of our proposed measures to selected measures of XCG, PRUM, and HiXEval (which have all been official measures at the annual INEX workshops described in Section 2.1).\footnote{http://www.inex.otago.ac.nz/}

### 1.5 Contributions of Work

The main focus of this thesis is the problem of formulating stable, consistent, and compatible SDR measures. Our major contributions are three-fold. First, we address stability by proposing relevance, navigation and redundancy (referred to as the pillars of SDR
evaluation) as a common basis for SDR evaluation (Chapter 3 based on Ali, Consens, Kazai & Lalmas [7]). We then experimentally validate our pillars by testing the stability of the measure Structural Relevance (Chapter 4 based on Ali, Consens & Lalmas [9], Ali, Consens, Kazai & Lalmas [7], and Ali, Consens & Larsen [6]). Our second major contribution addresses consistency by proposing the Extended Structural Relevance framework (Chapter 5 based on Ali, Consens & Lalmas [5]) which is a way to formulate SDR measures that are based on consistent definitions of relevance, navigation and redundancy. We limit our work in ESR to re-formulating existing SDR measures. Finally, our third major contribution is a low-cost simulation, called Scenario Stability Analysis, to test whether our ESR measures are compatible across ad-hoc SDR tasks (Chapter 6). Our contributions are described in detail, as follows:

**Pillars of SDR Evaluation**

1. In Section 3.1 we argue that SDR tasks can be evaluated by modelling SDR tasks (in element, passage and document retrieval) as tree retrieval and evaluating performance in terms of relevance, navigation and redundancy. We demonstrate how measure stability is a problem in SDR stemming from measure dependencies on the relevance assessment methodology in SDR.

2. We propose our pillars of SDR evaluation: relevance (Section 3.2), navigation (Section 3.3) and redundancy (Section 3.4). We show how our relevance and navigation can be determined from user studies, and argue that the effect of redundancy on performance can be inferred from relevance judgments and user preferences in navigation. In Chapter 4 we present results that validate the stability of our approach by testing the precision-based measure SR (which is based on our proposed pillars) across different search tasks using data from INEX 2006 and INEX 2007.

**Extended Structural Relevance**

1. In Section 5.1 we use our pillars to define the Extended Structural Relevance
(ESR) framework which is a way to formulate consistent SDR measures, akin to precision and recall in the Cranfield method, based on relevant, retrieved, and seen sub-documents.

2. In Section 5.2, we re-formulate currently inconsistent SDR measures into consistent measures using ESR that can account for task-specific performance but share common definitions of relevance, navigation and redundancy. Our main results are proposals of ESR-based measures for XCG, PRUM, and HiXEval.

Scenario Stability Analysis

1. In Section 6.1, we propose scenario stability analysis, a low-cost simulation of SDR, to test compatibility of measures (whether ESR or not). Our simulation models an exhaustive set of use cases that describe how SDR users may experience gain and redundancy across a range of related tasks.

2. In Section 6.2, we use scenario stability analysis to test ESR measure compatibility across ad-hoc tasks in SDR. Finally, in Section 6.3, we confirm our simulation-based tests with costly experimental validation using datasets from INEX 2006 and 2007.

1.6 Outline of Thesis

The outline of this thesis is as follows. In Chapter 2, we first review existing SDR measures (XCG, PRUM, and HiXEval); then we describe our problems of stability, consistency, and compatibility; and, finally, we discuss how diversity goes beyond our work. In Chapter 3, we propose tree retrieval and our pillars of relevance, navigation and redundancy to propose a stable basis of SDR evaluation. In Chapter 4, we provide experimental results to validate the stability of a measure based on our pillars. Next, in Chapter 5, we use our pillars to propose the Extended Structural Relevance (ESR) framework which defines set-based performance parameters that we use to re-formulate inconsistent SDR
measures into consistent measures. In Chapter 6 we propose Scenario Stability Analysis to test the compatibility of our measures for ad-hoc SDR search tasks. Finally, we complete our study in Chapter 7 with a statement of our conclusions and future work.
Chapter 2

Problems in SDR Evaluation

In this chapter, we present the background to our main problems. We review in Section 2.1 the INEX measures (XCG, PRUM, and HiXEval) used in our experiments. We identify our problems of stability (Section 2.2), consistency (Section 2.3), and compatibility (Section 2.4). We complete our background in Section 2.5 by discussing how diversification is related to, but goes beyond, our work.

2.1 INEX Measures

Much of the earliest work in SDR (summarized recently in Piwowarski, Trotman & Lalmas [116]) was undertaken at the annual Initiative for the Evaluation of XML retrieval (INEX) workshops.¹ INEX is a collaborative and international effort dedicated to the development of effective XML retrieval systems. Since 2002, INEX has investigated a wide range of SDR search tasks. Other research themes explored at the INEX include adapting the Cranfield method to evaluate SDR [44], collecting exhaustive (information breadth) and specific (topical focus) relevance assessments [81], whether users prefer vague (content-oriented) or strict (context-oriented) structural queries [79], modelling

¹http://www.inex.otago.ac.nz/
performance with ideality [78], judging relevance based on how users highlight relevant
text [80], and testing focused access to information [43]. These themes are discussed in
more detail in Section 2.3. Our work in Chapter 3 (and related experiments in Chapter 4)
was undertaken at the INEX 2007 to 2008. Our subsequent work (in Chapters 5 and 6),
although not conducted at the INEX, have nonetheless been strongly influenced by work
at INEX.

It is widely known that the evaluation of the range of SDR tasks has challenged
INEX since its beginnings and is still an active area of research [113]. The first SDR
systems investigated in the context of INEX were ad-hoc element retrieval systems. The
aim of these systems was to retrieve relevant XML elements, i.e., nodes in XML trees,
from a collection of XML documents. The first measures proposed at INEX to evaluate
performance consisted of adaptations of classical IR measures where the notion of a
relevant document was replaced by the notion of a relevant element (as simple hits and
misses). These measures proved to be unstable because they did not account for user
navigation and redundancy as illustrated in Figure 1.3 in Section 1.1.

Extended cumulated gain (XCG) [59] is a family of stable cumulated gain (CG) [52]
measures for evaluating element retrieval. CG is a classical measure based on graded
relevance where users find (1) highly relevant documents more useful when output in
earlier ranks, and (2) highly relevant documents are more useful than marginally relevant
documents, which, in turn, are more useful than irrelevant documents [66]. XCG adapts
CG to element retrieval. The two main contributions of XCG are evaluating redundancy
as wasted user effort [63] and the proposition of replacing desired recall in precision-recall
analysis (Section 1.2) with ideality [120]. Desired recall models a user’s information need
being satisfied at a given recall point. The user’s judgment of system performance is not
penalized if the user retrieves more relevant information than they desire. Ideality models
a user’s information need being satisfied at a given recall point with results optimally
ordered and encoded with preferred context (called, respectively, an ideal ranking and

...
an *ideal element*). However, if the system retrieves elements out of order or encoded without the preferred context then the user penalizes system performance. By the user spending the same effort, XCG compares the user’s gain from elements ranked in a given output (called *cumulated gain*) to elements ideally ranked (called *ideal cumulated gain*). We leave the presentation of the detailed calculation of XCG to Section 5.2.3. The main drawback of XCG is that it requires assessments in which obtaining assessor agreement is costly because the ideal conditions of optimal ordering and preferred context differs across topics and search tasks [61].

Precision-Recall with User Modelling (*PRUM*) [94] is a family of measures that extend PRRecall [98] where navigation to ideal elements in XML documents is stochastic. Ideal elements are defined in PRUM as those that minimize the effort a user spends to navigate to relevant information in the collection [99]. The main contribution of PRUM is that it proposes a probabilistic model for user navigation that can be tested through user studies. PRUM is based on the probability of the different scenarios where the user navigates different routes to satisfy his/her information need. PRUM is calculated, in general, by the ratio of the number of ranks where the user sees information in ideal elements and the number of ranks the user consults. We leave the presentation of the detailed calculation of PRUM to Section 5.2.4.

Like XCG, PRUM is an ideal measure (although it does not require an ideal ranking). In Piwowarski, Gallinari & Dupret [94], PRUM is interpreted as the probability that the user sees a previously unseen relevant element when consulting the (XML) context of a retrieved element assuming that the user wants to see a given number of relevant (ideal) elements:

\[
PRUM(l) = P(Lui|Retr, L = l, Q = q)
\]

(2.1)

where \(l\) is the desired recall level, \(q\) is the query, \(Retr\) is the probability that the user consults the element, and \(Lui\) is the probability that the element leads to previously unseen ideal elements.
The earliest PRUM measure at INEX, Expected Ratio of Relevant Documents (ERR), was the first INEX measure to take into account the user's navigation between elements in retrieved XML documents [93]. ERR is obtained by the ratio of the expected number of ideal XML elements a user sees (exactly once) when consulting the list of the first $k$ returned results divided by the expected number of relevant (however not necessarily ideal) XML elements a user sees (exactly once) whilst exploring the whole collection:

$$ERR = \frac{\mathbb{E}(N_R|N = i)}{\mathbb{E}(N_R|N = |Rel|)},$$  \hspace{1cm} (2.2)

where $N_R|N = i$ is the number of relevant XML elements in the first $i$ results, and $N_R|N = |Rel|$ is the number of relevant XML elements in the collection.

Both ERR (Equation 2.2) and PRUM (Equation 2.1) are complicated to calculate and limited to binary judgments of relevance (binary relevance). In practice, researchers use a significantly simpler, but related, variant of PRUM called Expected Precision-Recall with User Modelling (EPRUM) [91] which allows for graded relevance. Our experimental analyses of PRUM in Chapters 5 and 6 are calculated using EPRUM which is obtained, at a given recall-point, by taking the ratio of the minimum rank that achieves the given recall in an ideal system and the minimum rank that achieves the given recall in the actual system:

$$EPRUM(l) = \mathbb{E} \left( \frac{\text{min}_I(l)}{\text{min}_S(l)} \right),$$  \hspace{1cm} (2.3)

where $\text{min}_I(l)$ is the minimum number of consulted elements to achieve a recall $l$ in an ideal output, and $\text{min}_S(l)$ is the minimum number of consulted elements over all possible scenarios to achieve a recall $l$ in the actual output. In this work, we refer interchangeably to ERR, PRUM and EPRUM as simply PRUM. Our work in Chapters 3 and 5 borrow from concepts in PRUM for modelling navigation and enumerating the scenarios where users satisfy their information need.

The final INEX measure that we review in this section is Highlighting XML Evaluation (HiXEval) proposed in Pehcevski & Thom [89], and further finalized in Kamps et
HiXEval was developed to evaluate the performance of systems that retrieve (or can be modelled as retrieving) passages, i.e. passage retrieval, where a passage is a block of text, delineated or not with XML tags (when delineated, the passage is an XML element). HiXEval measures are adaptations of classical IR precision and recall. The main contributions of HiXEval are that it defines interpolated and generalized measures for SDR. In an interpolated measure, the relevance of a sub-document is considered independently of other sub-documents except, in cases of overlap, where performance is penalized based on the size of the overlapping region. In a generalized measure, the relevance of a sub-document is dependent on the total size of the relevant text in the original document. The importance of the distinction between interpolated and generalized measures is significant because, on the one hand, users prefer answers from regions of the document that are about a specific topic \[95, 86\] (i.e. interpolated performance); and, on the other hand, users seek the best single point in a document from which to access all of the relevant information in the document \[99, 54\] (i.e. generalized performance). We leave the presentation of the detailed calculation of interpolated precision and recall in HiXEval to Section \[5.2.2\]. Different HiXEval measures have been validated for most of the official SDR tasks investigated at the INEX. Moreover, unlike XCG and PRUM, HiXEval does not rely on ideality. However, in HiXEval, user navigation does not extend beyond the boundaries of retrieved passages. This limits HiXEval to tasks where user navigation does not play a significant role in performance evaluation. This completes our review of INEX measures.

### 2.2 Stability

IR evaluation is the foundation of modern IR research. Evaluation allows us to systematically explore, contrast and compare complex search strategies. Buckley & Voorhees \[21\] defines a stable measure as one that reliably captures system ranking; i.e. the binary
preference relationship of a user judging whether one system performs better (more effectively) than another. Without stable measures, we cannot reliably capture how human judges evaluate performance (due to both methodological and practical challenges) which significantly limits research. Our problem is that in ad-hoc SDR search there is no single stable measure, for all tasks, that captures relevance, navigation and redundancy.

An IR measure evaluates performance by comparing the output of a system to a gold-standard defined by human judgments of relevance. In Section 1.3 we describe our methodology for validating stability of a measure by either testing fidelity and accuracy, or by comparing the measure to a known stable measure. Regardless, the stability of a measure strongly depends on the quality of the assessments in the test collection and whether the assessments reliably capture the basis of evaluation, i.e. the criteria by which humans evaluate performance. In classical IR, the basis of evaluation is relevance whereas the basis in SDR is our pillars of relevance, navigation and redundancy.

The assessment methodology in SDR has been an area of significant research in recent years where our pillars have been considered in a number of different ways (summarized in Piwowarski, Trotman & Lahmas [96]). Early assessments (used in XCG and PRUM) at INEX considered best entry points [71] which are the (ideal) elements for the user to see all of the relevant text in a document with the least amount of navigation. Best entry points were assessed at INEX 2004 [79] and INEX 2005 [115] using multi-level judgments to identify the ideal elements (and ranking of elements) in the collection. Ideal elements, ideal ranking and ideality are described in Section 1.3. These early judgments suffered from poor assessor agreement. In Clarke & Terra [29], researchers noted that relevance assessment of sub-documents, unlike classical assessments of documents, are not independent. For instance, in an article encoded in XML, the relevance of a title element may depend on an associated section element that contains relevant text. In [111], Trotman theorized, based on poor assessor agreement in multi-level judgments of element retrieval, that SDR end-users interact with the system as if it were a passage retrieval
system (where relevance judgments for passages are analogous to classical IR judgments for documents as posited in Kaszkiel & Zobel [57]). In later work [112], Trotman argued that, as opposed to multi-level judgments, judgments of relevant information in XML are best captured by having assessors highlight relevant text and mark the location of the best entry point from which to begin reading the relevant text. Experiments in SDR assessment at the INEX are summarized in Piwowarski, Trotman & Lalmas [96] (p. 34) where the authors conclude that “multi-level judgments may be beneficial in information retrieval evaluation, but eliciting them may and often will impact on the reliability of the assessments due to the lack of assessor agreement”.

These challenges in SDR assessment motivated the work in Kazai [61] where the author tested the stability of XCG in element retrieval. Using a simulation, Kazai concluded that XCG (and other ideal SDR measures such as PRUM) are sensitive to the assessment methodology for judging ideality and apt to being unstable. Later ad-hoc retrieval studies at INEX 2007 in Fuhr et al. [43] and INEX 2008 in Kamps et al. [53], which compared element to passage retrieval, were used by Pal, Mitra & Kamps [88] (p. 377) to confirm that ideal precision metrics (such as XCG and PRUM) can be “unstable and must be used with caution when comparing the performance of systems”.

Lack of assessor agreement for ideality is symptomatic of instability in SDR measures and ultimately not the cause of instability. For instance, in Section 3.1.3, the challenges of developing a stable measure for tree retrieval go beyond achieving assessor agreement and include issues such as variable encodings of relevant information into trees, collecting complete assessments, and the challenge of extending non-tree measures to tree retrieval. Numerous SDR systems in the INEX literature that output trees cannot be evaluated because of a lack of a stable (ideal or not) measure. These include systems that use XML query languages such as XQueryFT [12], searching via matching trees to document fragments [24], returned passages as multiple ranges of information from XML documents [26], and other tree-based systems (such as XRANK [46], XKeyword [51], XXL [110],
and XIRQL \cite{42} that provide keyword search on XML document collections. However, none of these works propose a tree retrieval task. Moreover, in Piwowarski, Trotman & Lalmas \cite{96}, the authors analyse the history of SDR assessments at INEX and propose a methodology to address the challenges of assessor agreement and completeness. Their methodology captures our basis of relevance, navigation and redundancy. However, they do not proffer a measure that captures our basis of SDR that is also stable across tasks. We call this our problem of stability.

Chapter 3 (and originally published in Ali, Consens, Kazai & Lalmas \cite{7}) addresses our problem of stability. We propose our pillars of SDR evaluation to evaluate SDR tasks on the common basis of relevance, navigation and redundancy. Experimental validation of our proposal is shown in Chapter 4 where we evaluate tasks at INEX, modelled as tree retrieval, with our precision-based measure called Structural Relevance (SR). We test the fidelity and accuracy of SR to establish stability across tasks. We show that current SDR measures cannot evaluate tree-based tasks and that SR is the first stable IR measure in the literature for tree retrieval.

### 2.3 Consistency

*Consistent* measures use comparable calculations of relevance, navigation and redundancy. In Section 1.3 we stated that in SDR we calculate the user’s gain in relevant information based on a function called *relevance value* \cite{59} that revalues relevance judgments to account for user preferences that are not explicitly assessed. Our problem is that in SDR measures there is no consistent way to calculate gain based on relevance value. To understand the problem, we present a brief historical review of assessments at INEX. Our review is largely based on work in Piwowarski, Trotman & Lalmas \cite{96}.

To capture how users interact with XML elements, at INEX 2002 (and later published in 2007 in Lalmas & Tombros \cite{72}), researchers considered grading answers on a two-
dimensional scale of topical relevance and component coverage. Relevance was assessed on a four-point scale of: irrelevant (0), marginally relevant (1), fairly relevant (2), or highly relevant (3). Coverage was scored on a separate four-point scale of: no coverage (N), too-large (L), too-small (S), or exact coverage (E). Kazai, Massood & Lalmas [65] found coverage to be too confusing for human assessors and resulted in unreliable assessments. Classical IR measures of precision and PRecall (a stochastic measure of precision) were adapted to evaluate graded elements. However, given the unreliable assessments, these adaptations of classical measures were found to be unstable.

For judging elements at INEX 2003 and 2004, researchers adopted multi-level judgments of exhaustivity and specificity proposed in Kekäläinen & Järvelin [66]. Exhaustivity is the extent to which an answer covers or discusses a given topic. Specificity is the extent to which an answer focuses on the topic. To evaluate element retrieval systems, researchers proposed the measure XCG [59] (described in this work in Sections 2.1 and 5.2.3) which is based on retrieving ideal elements. In Malik, Lalmas & Fuhr [79], exhaustivity was found to not be reliable for evaluation because assessor agreement in judgments was found to be best at the extremes of the exhaustivity dimension (i.e., to users exhaustivity is analogous to binary relevance in classical IR). Hiemstra & Mihajlovic [50] concluded that multi-level judgments (like exhaustivity and specificity) neither capture how end-users perceive performance nor are simple enough for researchers to reliably achieve assessor agreement.

At INEX 2005 and 2006, Trotman [105] posed the question of whether users consider sub-document results more akin to structured information (i.e. XML elements) or passages of text. On the one hand, in [24], Clarke noted element and passage judgments could be made consistent by having assessors judge relevance in XML by highlighting text and having the encapsulating element inherit the judgment. To evaluate passage retrieval, Pehcevski & Thom [89] proposed the measure HiXEval (described in Sections 2.1 and 5.2.2). On the other hand, in [116], Trotman & Lalmas argued that sub-document
results are considered as structured information where users pose strict and vague queries to govern the granularity of search results (target elements) and the places in a document to navigate from to locate answers (support elements). To evaluate element retrieval tasks where users navigate, Piwowarski, Gallinari & Dupret [94] proposed the measure PRUM (described in Sections 2.1 and 5.2.4).

At INEX 2007 and 2008, passage retrieval was selected as superior to element retrieval for XML retrieval. Highlighting of relevant text passages was adopted as the official INEX assessment methodology [96]. HiXEval was refined in 2007 in Kamps et al. [56] to capture generalized document-recall based on retrieval of all relevant passages in a document as opposed to considering the relevance of each passage retrieved from a document as being independent. HiXEval measures were adopted as the official measures for most ad-hoc document, passage and element retrieval tasks. However, in tasks where navigation played a significant role, PRUM was adopted.

Based on our historical account above, we note that our INEX measures (XCG, PRUM and HiXEval described in Section 2.1) were proposed and differ in conjunction with advancements in assessment methods. Relevance in XCG is based on ideal ranking and ideal elements judged using multi-level relevance. Relevance in PRUM is based on ideal elements judged using binary relevance. Relevance in HiXEval is based on the character length of relevant text. Navigation in HiXEval and XCG is limited to modelling overlap [63]. In PRUM, navigation is a stochastic process of how users locate ideal elements in a document. Redundancy in XCG and HiXEval is based on overlapping text [36]. Whereas in PRUM, redundancy is captured only for ideal elements (which represent all of the relevant information in the collection but do not capture all elements that contain relevant information). We argue that these differences make INEX measures non-comparable because our pillars of SDR evaluation are defined differently for each measure. We call this our problem of consistency.

We address our problem in Chapter [5] by proposing the Extended Structural Relevance
(ESR) framework which we use to reformulate selected measures of HiXEval, XCG, and PRUM using consistent calculations for relevance, navigation, and redundancy.

2.4 Compatibility

Compatible measures are stable for the same set of search tasks. In [20], Bollman shows that measures based on Cranfield test collections (like precision and recall) are part of a family of set-based measures that can be calculated based on a partition of the test collection into sets of relevant and non-relevant, and, retrieved or not retrieved, documents. Bollman goes on to show that measures that are monotonic (like recall) and that can be iteratively computed across rank cut-offs from test collection statistics (like precision and recall) are sufficient to ensure that the measures will reliably order IR systems by performance given varying outputs, sets of topics, relevance judgments, and documents in the test collection. In other words, precision and recall are compatible for any document retrieval search task that can be tested using the Cranfield method. We argue that a driving force in the widespread adoption of the Cranfield method has been that most document retrieval search tasks can be tested in Cranfield using precision and recall.

Hiemstra & Mihajlovic [50] found the Cranfield method to be inadequate to capture the effect of navigation on SDR performance. Moreover, as posited in Piwowarski, Trotman & Lalmas [96], a single test collection does not readily capture how preferences in navigation can broaden the possible search tasks in SDR. The broad range of tasks in SDR makes it costly to evaluate if new test collections and measures must be proposed and validated each time a new task is researched.

In an address by Robertson [102] (p. 83), the speaker argues that “there is a strong compatibility argument for researchers to use the same methods as each other unless there is very good reason to depart from the norm”. In SDR, we argue that the shortcomings
of the Cranfield method for conducting experimentation justify such a departure. For instance, Betsi et al. [19] note that a Cranfield test collection is based on a document retrieval paradigm whereas in SDR users retrieve sub-documents. At INEX 2004, Malik, Lalmas & Fuhr [79] noted that Cranfield test collections do not capture how users navigate and experience redundancy in SDR. Trotman & Lalmas [116] argue that queries in classical IR specify only content whereas queries in SDR specify both content and structure. Finally, gain in classical IR is based on relevance judgments [20] whereas gain in SDR is based on relevance value which is sensitive to both how relevance is judged in SDR [96] and how users prefer to interact with an SDR system [59].

Current SDR measures at INEX capture task-specific performance that is difficult to apply across SDR tasks. For instance, ad-hoc retrieval for both elements and passages at INEX is composed of thorough tasks (find all relevant sub-documents); focused tasks (find all relevant sub-documents that do not overlap); best-in-context tasks (find the best point in each document to see relevant information); and, relevant-in-context tasks (for each document, find all relevant sub-documents that do not overlap and rank them in the order they appear in the original document). We use XCG to evaluate thorough tasks; interpolated HiXEval measures to evaluate focused tasks; PRUM to evaluate best-in-context tasks; and, generalized HiXEval measures to evaluate relevant-in-context tasks. There is no known single measure for ad-hoc retrieval that is stable across all of these tasks.

Chapters 3 and 5 address the challenges of defining SDR measures, akin to Cranfield. However, the cost of proving that our measures are stable by validating them across search tasks remains. We call this our problem of compatibility.

Our solution, presented in Chapter 6, is to propose that we replace costly validation with a simulation-based test to predict the stability of measures across an exhaustive range of tasks. Our test results in Chapter 6 predict that our proposed ESR measures from Chapter 5, unlike their INEX counterparts from which we derived our ESR mea-
sures, capture performance across a broad range of ad-hoc tasks. We verify our simulated predictions with costly experimental validation using data from INEX 2006 and 2007.

2.5 Diversity, Novelty and Coverage

In this section, we complete our problem description by discussing how novelty, diversity and coverage differ from the current work and represent an important direction for our future work.

*Diversity* is the user preference(s) in a population of users that top-ranked results contain different and complementary information [124]. *Novelty* is the user preference(s) in a population of users for each individual to direct their seeking of answers towards finding more information on specific subtopics of interest, rather than an undirected quest for any new information [38]. *Coverage* is the user preference(s) in a population of users that the breadth of subtopics in a given topic be reflected in ranked results [123]. Collectively, we refer to tasks that serve such preferences as diversity tasks or diversification because coverage and novelty are considered sub-types of diversity [124]. Much of the work in diversity tasks has been published in the Diversity Track at the annual Text REtrieval Conference (TREC) [28] where the basic approach has been to rank documents that contain the same information by penalizing the scores of the least relevant, redundant documents [38].

Diversity tasks and SDR tasks share the feature that users experience redundancy (by seeing the same information more than once). A key difference is that redundancy in diversity tasks stems from information duplication whereas in SDR it stems from information fragmentation (as mentioned in Section 1.1). In diversity tasks, users experience redundancy by reading retrieved documents that contain identical content [124]. Whereas, in SDR (illustrated in Figure 1.3), users experience redundancy when retrieved sub-documents overlap [63] or the user navigates between sub-documents [36]. However,
in Liu, Sun & Chen [74], the argument is made that overlap [63] should be used to rank sub-documents because it represents duplicated information. To our knowledge, Liu, Sun & Chen [74] is the only work in the literature that has considered diversity tasks in a sub-document retrieval paradigm.

Diversity tasks represent an important area of future work. An on-going challenge in the Diversity Track at TREC [28] (and a likewise challenge in SDR at INEX [116, 50]) has been to propose stable measures. Diversity task measures, such as intent-aware precision [2] and $\alpha$-nDCG [30], evaluate performance in terms of how well the system serves users who pose the same query but have various interpretations of the query (called sub-topics). Intent-aware precision is a weighted average of precision computed across sub-topics as if each sub-topic were distinct (i.e. precision from more popular sub-topics will have greater effect on the evaluation score). $\alpha$-nDCG calculates cumulated gain based on relevance assessments weighted by sub-topics (i.e. the more sub-topics in which a document appears the higher its relevance). In contrast, SDR measures (described above in Section 2.1) calculate gain based on how the user prefers relevant information to be encoded and tolerates overlap (neither of which are considered in diversity tasks). Germane to this work, Clarke et al. [30] argue the need for measures that capture diverse information needs (represented via sub-topics). This is similar to our argument in Section 2.4 where we argue the need for compatible measures for ad-hoc tasks in SDR.

We do not further consider diversity tasks because our work focuses on evaluation (as opposed to ranking), sub-document retrieval (as opposed to document retrieval), and capturing how users experience redundancy given fragmentation (as opposed to duplication). We believe that there exists a unification between diversity tasks and SDR tasks that would more fully capture the breadth of how users experience redundancy in IR. However, this unification is beyond the scope of this work.
Chapter 3

Pillars of SDR Evaluation

Users judge the performance of an SDR system based on relevance, navigation and redundancy. We call these our pillars of SDR evaluation. In this chapter, we define our pillars for tree retrieval. We then exploit tree retrieval to avoid the instability of existing SDR measures (discussed Section 2.2) when evaluating systems across the broad range of search tasks in SDR.

We begin in Section 3.1 by defining tree retrieval that, in general, cannot be evaluated using the INEX measures in Section 2.1. However, like passage retrieval, tree retrieval can be used to model (and evaluate) a broad range of SDR tasks. We next show how tree retrieval can be evaluated in terms of relevance (Section 3.2), navigation (Section 3.3) and redundancy (Section 3.4). We then summarize our proposed evaluation in Section 3.5 with a toy example. Finally, we conclude this chapter in Section 3.6. This chapter is based on our work originally published in Ali, Consens, Kazai & Lalmas [7] and Ali, Consens, & Lalmas [5].

3.1 Desiderata for the Evaluation of Tree Retrieval

We begin by defining the tree retrieval task; describing the user’s information seeking behaviour; and stating the requirements for evaluating tree retrieval.
3.1.1 Tree Retrieval Task

Tree retrieval is the task of returning trees that provide the user with access to nodes of an XML document that are relevant to their information need. It is an approach to SDR; as are element and passage retrieval. Tree retrieval differs from other SDR approaches because trees can represent information items that go beyond flat text units, such as documents, elements or passages. As with other SDR tasks, the task of returning trees to satisfy an information need requires a more complex notion of relevance that extends beyond the classical content-based criterion. The relevance of a tree depends on both its content and its context. Tree retrieval involves not only finding relevant information, but also finding trees that afford users access to this information. Finally, tree retrieval users find relevant information by either consulting the output to retrieve relevant subtrees, or by navigating from the nodes in retrieved subtrees to other nodes in the document. The relevance of a tree depends not only on its content and structure, but also on the content and structure of other trees in the same document. Different subtrees from the same document may afford users access to the same content. This redundancy between trees (and, similarly for elements and passages) is a pivotal difference between evaluation in classical IR and tree retrieval (and SDR, in general).

Tree retrieval results are ranked lists of trees extracted from documents in a collection. We denote the tree retrieval task with the following definition of the output.

**Definition 3.1.1 (Tree retrieval)** The task of a tree retrieval system is to output a ranked list of document subtrees (trees) \( R = t_1, t_2, \ldots, t_k \), where \( R \) is a ranked list, \( k \) is the length of the list, and \( t_i \) is a subtree retrieved from a collection \( C \) of trees such that \( t_i = t_j \) iff \( i = j \) for all \( i, j \in [1, k] \). We refer to the ranked list \( R \) as the output.

The document collection is represented as a forest of rooted trees where each distinct tree in the forest represents a distinct and whole document in the collection. A document in the collection is represented as follows.
Definition 3.1.2 (Tree) A tree is a simple connected graph $T = \{T_V, T_E\}$ where $T_V$ is a set of nodes, $T_E$ is a set of edges between pairs of nodes from $T_V$, $T_E \equiv \{\{x, y\}, x \in T_V, y \in T_V, x \neq y\}$, a specified node in $T_V$ is the root of the tree, and there are no cycles in the graph.

Each node in a document tree represents a document component. For instance, in XML, a node will be one or more XML elements. The set of nodes in the tree that are connected by edges to a given node represent the context of that node. Thus, information is contextualized in trees as subsets of nodes from the document that are connected by paths that form a subtree, which itself is a tree. Formally, a subtree of this type is referred to as an induced subtree.

We assume that the subtrees in an output $R$ are all different, but that different subtrees may overlap each other. We define trees as being different if one contains a node or edge not found in the other. In this work, it is sufficient to simply compare nodes s.t. trees $T$ and $S$ are different if $T_V \setminus S_V \cup S_V \setminus T_V \neq \emptyset$.

Definition 3.1.3 (Induced Subtree) An induced subtree $t$ of tree $T$ is given as $(t_v, t_e)$ such that the set of nodes $t_v \subseteq T_V$, with a set of edges $t_e$, must form a tree according to paths defined in the set of edges $T_E$ of the tree $T$. We refer to induced subtrees simply as trees. Moreover, when we refer to the tree $t$ as a set, it refers to its set of nodes $t_v$.

An induced subtree is a tree, and we use the terms interchangeably, unless stated otherwise. The simplest tree is a single node called a singleton. We model element retrieval as systems that retrieve singletons.

Definition 3.1.4 (Singleton) A singleton is an induced subtree $t$ of tree $T$ where the set of nodes $t_v$ is a single node $e$, with an empty set of edges $t_e = \emptyset$.

A passage is a block of text, delineated or not with XML tags. A passage can span elements in an XML document. In our work, we assume that passages can be represented
as either a single node where the passage text is completely contained in a single node; or as sibling nodes in a tree where the passage is completely contained in the sibling nodes.

We refer interchangeably to induced subtrees with a single node as either singletons or nodes. A ranked list of nodes $R = e_1, e_2, \ldots, e_k$ is considered to be the same as a ranked list of singletons $R = t_1, t_2, \ldots, t_k$ where $t_i = \{e_i\}$. Let $R_i$ denote the sublist of $R$ up to rank $i$. We differentiate between nodes (singletons) and trees using the variables $e$ and $t$, respectively. Specific to XML retrieval, we refer to nodes as elements. XML elements are nodes in the document tree of an XML document. Additionally, a sub-document refers to parts of the document that may or may not be representable as a tree.¹ Our work is limited to evaluating systems where the collection, sub-document assessments and sub-document results can be modelled using trees.

We believe that this work could be expanded beyond trees, however we would need to revisit whether our pillars and information seeking process are sufficient to capture user preferences in systems that go beyond trees.

To evaluate performance, human assessors judge, for a given topic, the relevance of retrieved sub-documents. Let $a$ be a judged sub-document and $\text{rel}(a) = [0, \infty)$ be the scalar-valued judgment. If $a$ is relevant then $\text{rel}(a) > 0$, otherwise $\text{rel}(a) = 0$ and our sub-tree $a$ is not relevant. We refer to the set of judged sub-documents as the assessments. Let $A = a_1, a_2, \ldots, a_n$ denote a set of $n$ assessments for some given topic. In this work, we assume that our judged trees do not overlap and judgments of sub-trees are assessed independently of each other. In Section 3.2 below, we provide a more in-depth discussion of relevance in SDR and show how relevance values are obtained in this work.²

¹For example, neighbourhoods of nodes defined by hyperlinks in the collection form a sub-document graph that is generally not representable using trees.

²This is a correction to Ali, Consens, Kazai & Lalmas [7] where we incorrectly stated on p. 1155 that “each assessment is a different subtree and that judgments are independent”.

3.1.2 Information Seeking Behaviour

As described above, tree retrieval systems output ranked lists of trees. A tree is an example of a sub-document. A sub-document is a fixed region in a document that has definite boundaries but is not necessarily contiguous. For instance, a sub-document taken from a book XML document may be composed of the body element of the book and two chapters that are not adjacent to each other, e.g. chapters 1 and 3. Other examples of sub-documents include nodes, passages and entire documents all of which can be represented as trees (as shown in Section 1.1 in Figure 1.2).

In SDR, users gain relevant information by seeing relevant text. Our model of how users see relevant text consists of three main user behaviours: (1) consulting the output, (2) visiting retrieved trees, and (3) navigating between trees. The user interacts with the system as follows. A user consults the output by retrieving a tree from the search results and having the system display it. The user visits the retrieved tree by reading the text contained in it. After completing their visit, the user navigates by interacting with the user interface to change the system display from the current tree to somewhere outside of the current tree. The user may either navigate to another part of the document or abandon the document to resume consulting the output. If the user navigates to another part of the document, then, once the navigation has completed, the user will visit the part of the document that is displayed by the system. While either consulting or navigating, the user does not see any content and does not gain relevant information. Only during a visit can the user see text and thereby gain relevant information. In our work, we limit our model to the user making at most two visits per consultation. The first visit is always to the retrieved tree. The second visit is to the tree to which the user navigates to from the retrieved tree.

We assume that a user consults the output in rank order starting from the highest ranked tree. The user continues consulting the output until their information need has been satisfied (or they have exhausted the results in the output). During this process of
Chapter 3. Pillars of SDR Evaluation

seeking relevant text the user may encounter already seen nodes, thus redundant content. If the user tolerates redundant content, then the relevant content that is seen a number of times remains relevant to the user. If the user does not tolerate redundant content, then relevant content is considered non-relevant if already seen.

![Tree structure of book](image)

**Figure 3.1:** Tree structure of book in Figure 1.1

For example, consider the ranking \( R = t_1, t_2 \), where \( t_1 \) is the subtree formed by the nodes 1, 2, 3, and 4 in Figure 3.1 and \( t_2 \) is the subtree containing the nodes 14 and 15. In consulting \( R \) for relevant information, the user first visits \( t_1 \), seeing each node in \( t_1 \). After completing their visit, the user might seek additional relevant information by navigating out of \( t_1 \) into the rest of the document. By navigating, the user may visit (and thereby see) all or part of tree \( t_2 \). Eventually, the user will abandon the retrieved document to consult the system to visit the next result \( t_2 \), thereby seeing each node in \( t_2 \). Prior to visiting \( t_2 \) by consulting the output a second time, the user’s browsing history contains tree \( t_1 \) for certain, and potentially all or part of \( t_2 \) because of navigation from \( t_1 \). If the user sees \( t_2 \) by both navigation from \( t_1 \) and visiting \( t_2 \) by consulting the output, then \( t_2 \) may be redundant to the user. Redundancy occurs when a user sees the same relevant text more times than they tolerate. During a visit to \( t_2 \), if the user does not tolerate seeing information more than once, then any node in \( t_2 \) that is in the user’s browsing history would be non-relevant in the current visit. This way, the user’s browsing history affects the user’s perceived relevance of a result subtree.
Our user model, presented above, is a highly simplified abstraction of the actual process by which humans interact with SDR systems. Such abstractions are the norm in IR as opposed to the exception. For instance, in the Cranfield method, we model the user consulting the output in rank order where the user does not skip ranks, abandon the search results, or select results by the text snippet associated to it in the output. Another example, Page & Brin [87] justify PageRank based on the “random surfer” who clicks on hyperlinks at random with no regard to content. Our ESR user model is the first in the literature, that we are aware of, for users interacting with tree-based results. We argue that it is sufficient for the present work. Allan et. al. [11] (p. 8) contend that such abstractions have “been incredibly successful in [IR] research to advance rapidly, creating more effective and efficient systems for retrieving and organizing information”. However, Allan et. al. (who represent more than 20 of the leading researchers in IR today) go on to caution that these abstractions force “systems toward user-generic approaches that are ‘good enough’ for everyone, and therefore ‘never great’ for anyone”. In developing better user models, the authors warn (p. 9) that to gather user interaction data “is expensive, labor intensive, and requires expertise that is not universal in the information retrieval community”. Other challenges that they identify are that there are no benchmarks or standardized tests to validate complex models; the state-of-the-art in IR still struggles to deal with personalization (i.e., no general experimental method to compare performance given different user interaction patterns); and, finally, privacy is a significant concern in collecting the personal data needed to form better user models. These issues go beyond our work and we expect significant refinements to our user model in the future.

3.1.3 Requirements for the Evaluation of Tree Retrieval

The evaluation of tree retrieval systems rests on three fundamental requirements:

1. the relevance of retrieved trees in the output are not independent and depend on whether users tolerate redundancy,
2. the purpose of the system is to retrieve trees that afford a user access to relevant information by directly visiting a node in the tree or through navigating from a visited node into the rest of the document, and

3. the same relevant information may be expressed in trees of varying structure.

Traditional IR measures cannot be used for tree retrieval due to the dependencies in the output (first requirement). Existing SDR metrics such as XCG and PRUM, if they were extended to trees, would meet the first (i.e., dependencies) and second requirements (i.e., access to relevant information). However, the third requirement of variable-structured trees is not possible to meet when a metric requires an ideal ranking of trees because of (a) problems in assessor agreement of ideal subtree structure (see Section 2.2), and (b) the cost of collecting judgments for all possible subtrees in a collection. This completes our description of the tree retrieval task.

3.2 Relevance

Our first pillar is relevance. In IR evaluation, the relevance of information is a judgment made by a human assessor on whether the subject matter of the information is meaningful to a given information need. In classical retrieval, given a topic, the relevance of information objects (e.g. documents) in the collection are judged independently of each other. Gain is composed of the sum of independent relevant documents retrieved by the system. de Vries, Kazai & Lalmas [36] argue that this independence does not hold for gain in SDR because we must also include how users interact with the system to see sub-documents to distinguish relevant information from redundant information.

As posited in Piwowarski, Gallinari, and Dupret [94], a user gains relevant information in SDR when the information is seen by either consultation, navigation, or a combination of both. However, a user may consider the information contained in the sub-document, albeit relevant, not useful, i.e. sub-optimal gain, because either it is redundant or its
encoding format does not provide an ideal context \[59\]. We represent the user’s gain from seeing sub-documents with \textit{relevance value} (as defined in Section 2.2) which is a scalar value that is a revaluation of relevance judgments to account for wasted effort when users navigate and consult to locate relevant text. In classical IR, because documents are assumed independent, relevance and relevance value coincide.

How to assess the relevance of sub-documents is an active area of SDR research \[96\]. In Kazai \[61\], the author shows that the assessment methodology, based on ideality, introduces instability into SDR measures (as discussed in Section 2.2). The author suggests that instability can be avoided by: (a) assessing the relevance of information independently of redundancy in the output, (b) assessing relevance without considering how a user may navigate to information, and (c) evaluating system effectiveness based on the effect of user navigation and redundancy on the user gain in relevant information. Suggestions (a) and (b) remove the need to assess ideality. Suggestion (c) implies that good SDR measures evaluate how users spend effort to achieve gain. In this work, we address suggestions (a) and (b) by assuming independence between relevance and user navigation (Assumptions 3.2.3 below). We address suggestion (c) by using expected gains and ignoring relevant information seen with wasted effort.

We recall from Section 1.1 that users gain relevant information from hits and near-misses. Near-misses are defined in Kazai & Lalmas \[59\] as retrieved sub-documents that, may not be relevant, but which can be navigated from by the user to see unretrieved, relevant information. In this work, we reverse this definition. We consider a \textit{near-miss} as a relevant sub-document that has not been retrieved and that can be seen by the user via navigation from retrieved sub-documents. A \textit{hit} is a relevant sub-document in the output. Finally, a \textit{miss} is a relevant sub-document that is not seen by the user because it is neither retrieved nor navigated to. A miss represents unrealized gain because if the system were to output the sub-document at a rank lower than the cut-off at which we evaluate performance, then the user would experience gain from what was earlier a miss.
Case Description

<table>
<thead>
<tr>
<th>Case</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>hits</td>
<td>Relevant sub-tree retrieved.</td>
</tr>
<tr>
<td>misses</td>
<td>Relevant sub-tree not retrieved and not navigated to.</td>
</tr>
<tr>
<td>near-misses</td>
<td>Relevant sub-tree not retrieved but navigated to.</td>
</tr>
</tbody>
</table>

Table 3.1: Cases of gain for tree retrieval.

In contrast, if the user sees a relevant sub-document by first navigating to it and then, in a lower rank, by consulting for it in the output, then the user experiences redundancy, i.e. a waste of effort, and the user does not experience the full relevance of the sub-document. Hits, near-misses and misses define the basis of gain in our work and are summarized in Table 3.1.

Relevance is judged by assessors without considering navigation and redundancy.

**Definition 3.2.1** Let $A$ be called the assessments and let it denote the sub-documents in the collection that have been judged by assessors for relevance to a given topic.

**Definition 3.2.2** Let $rel(a)$ be the relevance value for the assessed tree $a \in A$ where if $rel(a) > 0$ then tree $a$ is relevant, and otherwise $rel(a) = 0$ and $a$ is not relevant.

In evaluation, the user’s gain in relevant information depends on how the user sees information, i.e. hits and near-misses. Thus, relevance value gain depends on navigation and redundancy. In our work, user navigation is random and we can calculate the user’s gain in relevant information by conditioning the relevance value of information by whether the user sees the information in a hit, near-miss, or miss.

To calculate gain, we make the following assumption:

**Assumption 3.2.3 (Structural Relevance Assumption)** Relevance is judged without considering how a user may navigate from search results to locate relevant information. The gain in relevant information that a user experiences depends on whether the user sees the information redundantly, i.e., without wasting effort.
In essence, we are assuming the dependence of relevance gain on the outcome of the user spending effort and not on the amount of effort spent (i.e., ranks consulted or route of navigation).

In order to calculate the expected gain in relevance value from a sub-tree, we characterize statistically a user’s information seeking process.

**Definition 3.2.4 (Browsing History)** Let $R$ denote the ranked list of sub-trees, i.e. the output, that may have been previously seen by the user via consultation. This list is also referred to as the browsing history.

**Definition 3.2.5 (Sub-trees with respect to consultation)** When characterizing a user’s information seeking process, there are two distinct probability distributions that arise, depending on whether the sub-tree being analyzed is in the set of “hits” or “misses”, i.e. whether the sub-tree is seen by consulting the output or not. We formalize this distinction as follows.

Let $C$ denote the condition that a sub-tree is in the output (can be consulted), and let $\overline{C}$ denote that a sub-tree is not in the output (and cannot be consulted). In general, we write $a$ for any element in $A$ (per Definition 3.2.1), however, we use the notation $a_C$ and $a_{\overline{C}}$ to distinguish these conditions, since this condition is known a priori and our probability model must account for this information.

**Definition 3.2.6** Let $\Omega_C$ ($\Omega_{\overline{C}}$, respectively) denote all possible outcomes of an information seeking process with condition $C$ ($\overline{C}$), where a user seeks a sub-tree $a_C$ ($a_{\overline{C}}$) by navigation using a system that outputs a ranked list $R$ of sub-trees.

**Definition 3.2.7** Furthermore, let $N$ denote the statistical event that a sub-tree $a$ is seen by navigating to it from a sub-tree in the output, and let $\overline{N}$ represent the statistical event that sub-tree $a$ is not seen by navigating to it.
Now we have two separate statistical universes $\Omega_C$ and $\Omega_{\overline{C}}$, both of which contain the same 2 possible outcomes. We write

$$\Omega_C = \Omega_{\overline{C}} = \{N, \overline{N}\}. \quad (3.1)$$

**Definition 3.2.8** Next, let $X_C$ ($X_{\overline{C}}$, respectively) be a random variable representing the outcome of a user navigating to see (to not see) sub-tree $a_C$ ($a_{\overline{C}}$) where $a_C$ ($a_{\overline{C}}$) is fixed for a single trial. We write $X_C = x_C$, where $x_C \in \Omega_C$ ($X_{\overline{C}} = x_{\overline{C}}$, where $x_{\overline{C}} \in \Omega_{\overline{C}}$).

To represent the user’s gain from shrinkage (described below), hits, near-misses, and misses, we use functions of the random variables $X_C$ and $X_{\overline{C}}$, which are themselves random variables, as follows. Let

$$\text{Shrink}(X_C) = \text{rel}(a) \text{ if } N \text{ occurs, else } 0, \quad (3.2)$$
$$\text{Hit}(X_C) = \text{rel}(a) \text{ if } \overline{N} \text{ occurs, else } 0, \quad (3.3)$$
$$\text{Near}(X_{\overline{C}}) = \text{rel}(a) \text{ if } N \text{ occurs, else } 0, \quad (3.4)$$
$$\text{Miss}(X_{\overline{C}}) = \text{rel}(a) \text{ if } \overline{N} \text{ occurs, else } 0. \quad (3.5)$$

We briefly note two things. First, in these definitions, we mention but do not further consider $\text{Shrink}(X_C)$, i.e. the gain for event $N$ occurring under condition $C$, where the user navigates to sub-tree $a_C$. We call this *shrinkage* because the user wastes effort to see sub-tree $a_C$ and, in this work has no way to realize the gain from the sub-tree. Secondly, the gain from a “miss” is an *unrealized gain* because the user experiences no actual gain but could experience the gain in future trials.

Now we would like to state the distributions of the random variables $X_C$ and $X_{\overline{C}}$. Let $p(a; R)$ denote the probability that a user navigates to tree $a$ from one or more of the trees given the browsing history $R$. For condition $C$, where the user sees the tree by consulting the output, the browsing history consists of the higher-ranked trees in the output. Let $m$ denote the rank position of $a$ in output $R$, and let $R_{m-1}$ be the sub-list of $R$ up to rank $m - 1$. For condition $\overline{C}$, the browsing history consists of the entire
output $R$. Given the above definitions, the probability distributions of outcomes under conditions $C$ and $\bar{C}$ are

$$
P(X_C = x_C) = \begin{cases} 
p(a; R_{m-1}) & \text{if } x_C = N, \\
1 - p(a; R_{m-1}) & \text{if } x_C = \bar{N},
\end{cases} \quad (3.6)$$

$$
P(X_{\bar{C}} = x_{\bar{C}}) = \begin{cases} 
p(a; R) & \text{if } x_{\bar{C}} = N, \\
1 - p(a; R) & \text{if } x_{\bar{C}} = \bar{N}.
\end{cases} \quad (3.7)$$

We will derive an explicit expression for calculating $p(a; R)$ in Section 3.4.

For a single trial using a fixed sub-tree $a$, let us calculate the expected relevance value of the user’s gain. Consider the expected gain for $\text{Hit}(X_C)$ in Equation 3.3 which is only non-zero, i.e. $\text{rel}(a)$, for event $N$. The expected value of $\text{Hit}(X_C)$ is

$$
E[\text{Hit}(X_C)] = \sum_{x_C \in \Omega_C} \text{Hit}(x_C) \times P(X_C = x_C)
= \text{rel}(a) \times (1 - p(a; R_{m-1})),
$$

where we have used the fact that $\text{Hit}$ is zero for value $N$. Analogous reasoning holds for the $\text{Miss}$ and $\text{Near}$ functions of $X_{\bar{C}}$ defined earlier.

Now let us summarize the expected gains in relevance value from any given sub-tree. The expected gain from a hit at rank $m$ is

$$
E[\text{Hit}(X_C)] = \text{rel}(a) \times (1 - p(a; R_{m-1})), \quad (3.8)
$$

the expected unrealized gain from a miss is

$$
E[\text{Miss}(X_{\bar{C}})] = \text{rel}(a) \times (1 - p(a; R)), \quad (3.9)
$$

and the expected gain from a near-miss is

$$
E[\text{Near}(X_{\bar{C}})] = \text{rel}(a) \times p(a; R). \quad (3.10)
$$
These expectations are crucial to understanding our proposed ESR framework in Chapter 5 because they form the basis of our framework. In this chapter, we will revisit these in Section 3.4 where we describe how to calculate redundancy, i.e. $p(a; R)$.

### 3.3 User Navigation

A navigation is a user interaction with the system user interface where the user browses out of a sub-document to seek (further) relevant information in another sub-document in the collection. In our work, we model user navigation as a random walk of the nodes in a document tree. We represent navigation as the conditional probability of our user navigating to a node from a given node within a given number of steps.

#### 3.3.1 User Navigation Graph

User navigation is part of the information seeking behaviour of SDR users. To model navigation, we introduce the user navigation graph which is a graph of a partition of the nodes in the collection. We determine user navigation, given a user navigation graph, by first weighting the navigable edges in the graph, then row normalizing these weights in a matrix, and finally finding our probability by iteratively multiplying the matrix with itself the number of steps that we assume that the user will take. In certain circumstances, we can iteratively multiply the matrix with itself until all rows are equal to obtain steady-state probabilities [103].

The user navigation graph allows modelling different navigational strategies in tree retrieval (e.g. navigation via document structure, contextual markup, semantic linking) at different granularities. The weight between two nodes reflects the effort associated with the user navigating between them. Weights can be derived through the analysis of clicks, time spent, common routes, or retinal focus. We then use our weights to calculate our navigational probabilities.
Figure 3.2: (a) Tree structure of an article, (b) Possible navigations in document, and (c) User navigation graph based on partition

Route 1  \( e_4 \leftarrow e_2 \leftarrow e_1 \leftarrow e_3 \)

Route 2  \( e_5 \leftarrow e_4 \leftarrow e_2 \leftarrow e_3 \)

Route 3  \( e_6 \leftarrow e_1 \leftarrow e_3 \)

Figure 3.3: Examples of routes navigated

Let us demonstrate with a toy example of how to formulate a user navigation graph where we observe users navigating along routes in structured documents. Consider observing how users may navigate in an XML document. Pictured as a spanning tree, Figure 3.2(a) represents an XML document containing elements article (a) with sections (section and ss1, respectively) and paragraphs (p) where, given an information need, nodes \( e_3 \) and \( e_4 \) are relevant. The node identifiers are shown beside each respective node, and their character lengths are shown in parentheses. So, for instance, node \( e_3 \) is 30 characters long.

The tree (also called a document tree in XML) shows the logical structure of all of the elements in the document. In our work, the document tree itself is called the elementary user navigation graph. For illustrative purposes, we assume for this example
that the sole interface that a user interacts with to navigate is by clicking on hyperlinks such that a node is visited if and only if the user clicks on a link corresponding to the node. The assumed user interface for navigation means that the edges in our elementary user navigation graph in Figure 3.2(a) are all bi-directional. Therefore, as shown in Figure 3.2(b), from any node in our example tree, the user can navigate to any other node in the tree in a single step. Navigation, in our work, is based on weighting the edges of the graph shown in Figure 3.2(b).

By tracking the successive clicks on our example user interface, we could collect the routes that users take to navigate. Figure 3.3 shows three examples of routes. Route 1 describes a user who entered the document via node $e_3$, then stepped to node $e_1$ then $e_2$ then $e_4$. Route 1 is composed of three steps; $e_1 \leftarrow e_3$, $e_2 \leftarrow e_1$, and $e_4 \leftarrow e_2$. For illustrative purposes, as a possible measure of effort, let the number of times a step is observed indicate the ease with which users navigate it. Let us assume that the effort spent is independent across routes and steps. For instance, based on the routes shown in Figure 3.3, step $e_1 \leftarrow e_3$ requires less effort than step $e_2 \leftarrow e_3$ because $e_1 \leftarrow e_3$ occurs twice whereas step $e_2 \leftarrow e_3$ occurs only once. Normally, we would include the terminators on routes to be counted as abandonments i.e. $e_j \leftarrow e_j$, however, for simplicity, we ignore our terminators in this example.

Let us next weight the edges of the graph in Figure 3.2(b) using the steps in the routes shown in Figure 3.3. For instance, step $e_1 \leftarrow e_3$ occurs twice in our example routes and corresponds to weights $w(e_1; e_3) = w(e_3; e_1) = 2$. We can then use our weighted graph to calculate our navigation probabilities. Table 3.2 summarizes the weights and probabilities in parentheses for the routes shown in Figure 3.3.

In practice, the weights in the elementary user navigation graphs cannot be determined (e.g. via human studies) because the graphs can be very large. Indeed, if $N$ is the number of nodes in the collection, for each node $e_i$, there will be $N - 1$ probabilities $\tilde{p}(e_i; e_j)$ needed to define navigation. Therefore, $N \times (N - 1)$ weights are needed.
to calculate \( \tilde{p}(e_i; e_j) \) for all nodes in the collection, which is impractical to assess in user studies for large \( N \). We therefore consider a simplified model for navigation where the user navigates from one node subset to another, and the subsets form a partition of the nodes in the collection. For instance, the nodes in the tree shown in Figure 3.2(a) can be grouped by their XML tags: \textbf{article} node \( e_1 \); \textbf{section} nodes \( e_2, e_3, \) and \( e_6 \); and \textbf{paragraph} nodes \( e_4 \) and \( e_5 \). Figure 3.2(c) shows a graph based on this partitioning scheme where \( S_1 \) contains \textbf{article} nodes; \( S_2 \) contains \textbf{section} nodes; and, \( S_3 \) contains \textbf{paragraph} nodes. Again, in terms of navigation, the user could navigate in a single step between any of the partitions in the graph shown in Figure 3.2(c).

Using this partition, we weight the directed edges between partitions. For instance, in Table 3.2, the outgoing steps from the nodes in \( S_2 \) are: \( e_3 \leftarrow e_1 \) twice, \( e_2 \leftarrow e_3, e_4 \leftarrow e_2 \) twice. We sum our outgoing steps (shown as weights in Table 3.2) between partitions to obtain the following weights on the edges from \( S_2 \): \( w(S_1; S_2) = 2 \), \( w(S_2; S_2) = 1 \), and \( w(S_3; S_2) = 2 \). Table 3.3 summarizes the weights and our navigation probabilities (in parentheses) for this model.

In the above examples shown in Tables 3.2 and 3.3 we have calculated our navi-
Chapter 3. Pillars of SDR Evaluation

Nodes

\[
\begin{array}{ccc}
S1 & S2 & S3 \\
S1 & 0 & 2(1.0) & 0 \\
S2 & 2(0.4) & 1(0.2) & 2(0.4) \\
S3 & 0 & 0 & 1(1.0) \\
\end{array}
\]

Table 3.3: Summary model weights and navigation probabilities (in parentheses) for nodes \(S1\), \(S2\), and \(S3\) in Figure 3.2(b).

Navigational probabilities for single step navigation. We call this a transition matrix. We calculate navigation for an arbitrary number of steps by iteratively multiplying the transition matrix with itself the arbitrary number of times.

Our approach for measuring effort in navigation is inspired by the study in Hammer-Aebi et al. [47], where for a given information need, the user is tasked with finding, judging and marking the relevant parts of a retrieved document. The study begins by presenting the user with a document where retrieved information has been highlighted. The user’s attention is directed to an initial highlight, referred to as the entry point. The user then navigates within the document using whatever means provided by the graphical user interface (such as scrollbars or hyperlinks in a table of contents). The user navigation is recorded as steps between nodes along a route starting from the entry point. The effort spent to make each step is measured. Examples of observations that capture the user’s willingness to spend effort include cumulated gain [32], tolerance to irrelevance [36], expected search length [35], or time taken to read documents [39]. In the three remaining subsections, we more formally present the probabilistic model for navigation used in our work.
3.3.2 Navigation Between Nodes

Navigation $\tilde{p}(e; f)$ is the probability that node $e$ is seen after the user has taken a given number of steps from node $f$. A navigation involves one or more steps in a random walk of the navigation graph. Node $e$ may be seen either because of the navigation (i.e. in visit after given navigation) or because nodes $e$ and $f$ overlap (i.e. in visit before given navigation). Our user navigation is thus obtained by calculating $\tilde{p}(e; f) = Pr(\text{Visit} \lor \text{Navigation})$ where $Pr(\text{Visit})$ is the probability that $e$ has been seen before our navigation and $Pr(\text{Navigation})$ is the probability that our user sees node $e$ immediately after our given navigation to node $e$.

Let us first derive our before probability $Pr(\text{Visit})$. If the region of text bounded by node $e$ is contained in the region of text bounded by node $f$ then we assume that node $e$ has been seen with certainty without navigation and $Pr(\text{Visit}) = 1$. Otherwise, $Pr(\text{Visit}) = 0$. Let $\delta(e; f)$ denote our probability $Pr(\text{Visit})$. In an elementary navigation model, every node in the navigation model corresponds to a single, unique document element in the collection. Our probability of previous visitation is zero, $Pr(\text{Visit}) = 0$ for all pairs of nodes except when $e = f$. However, in more practical models, a single node in the navigation graph may correspond to one or more child document elements. We assume in this case that if the child element is visited then the entire parent is seen. For example, a paragraph node in the navigation graph may include all presentation mark-up, such as boldface or italicized text and the navigation node is paragraph for any presentation elements retrieved from the paragraph. Continuing our example, if two presentation elements $e_i$ and $e_j$ are in the output from the same paragraph $p$, then $\delta(e_i; e_j) = \delta(p; p) = Pr(\text{Visit}) = 1$ because we assume that the entire paragraph is seen after visiting either element.

Let us now derive our after probability $Pr(\text{Navigation})$. The user navigates to node $e$ with the conditional probability $Pr(e; f)$. We next derive $Pr(e; f)$ (shown in Equation 3.11 below) which equals $Pr(\text{Navigation})$. 
Navigation, in our work, is based on experimental observations in controlled studies of users navigating while seeking relevant information in a document. Our observations consist of pairs of nodes \((e_i \leftarrow e_j)\) where each pair represents a trial of the user navigating from \(e_j\), and completing the trial by navigating to the region of text bounded by node \(e_i\). Our special case of abandonment is represented as the observed navigation \((e_j \leftarrow e_j)\).

Let \(N_T\) denote the number of trials observed of users navigating in document \(T\). Let \(n_{ij}\) denote the number of observed one-step navigations from node \(e_j\) to node \(e_i\). We note that our observations of navigations undergo a mapping of nodes in our navigation graph to elements in the document where a node in the graph may correspond to a parent element that includes all of its child elements, e.g. presentation mark-up included as part of a paragraph node.

We model navigation as a random walk of the nodes in the navigation graph. The probability of navigating from \(e_j\) to \(e_i\) is the total number of observed navigations from \(e_j\) to \(e_i\) divided by the total number of observed navigations from \(e_j\) to anywhere. Our probabilistic model of navigation is as follows. Let \(Pr(e_j) = \sum_k n_{kj}/N_T\) denote the probability that the user navigates from node \(e_j\). Let \(Pr(e_i \leftarrow e_j) = n_{ij}/N_T\) denote the joint probability of the user navigating from node \(e_j\) and navigating to node \(e_i\). Let \(Pr(e_i; e_j) = Pr(e_i \leftarrow e_j)/Pr(e_j) = n_{ij}/\sum_k n_{kj}\) denote the conditional probability of the user navigating from node \(e_j\) to node \(e_i\). We re-state our conditional probability below:

\[
Pr(e_i; e_j) \equiv \text{Navigation from given node } e_j \text{ to node } e_i = \frac{n_{ij}}{\sum_k n_{kj}} \text{ where } i, j, k \in [1, |T|],
\]

where \(T\) is a document in the collection, \(|T|\) is the number of nodes in the document \(T\), \(e_i\) and \(e_j\) are nodes in the document \(T\), \(n_{ij}\) is the number of navigations to node \(e_i\) from node \(e_j\), and subscripts \(i, j, k\) are indices to nodes in document \(T\).

We can now define user navigation between nodes \(p(e; f)\) as follows.
Definition 3.3.1 (User Navigation Between Nodes) The probability that node \( e \) is seen after navigating from node \( f \) is as follows:

\[
\tilde{p}(e; f) = Pr(\text{Visit} \lor \text{Navigation}) = \delta(e; f) + Pr(e; f) - \delta(e; f) \times Pr(e; f) \tag{3.12}
\]

where \( e \) and \( f \) are nodes, \( \delta(e; f) = 1 \) if the region of text of node \( e \) is contained in the region of text of node \( f \), otherwise \( \delta(e; f) = 0 \); and \( Pr(e; f) \) is the conditional probability of navigating to node \( e \) given a navigation from node \( f \) shown in Equation 3.11. If \( \delta(e; f) = 0 \) then \( \tilde{p}(e; f) = Pr(e; f) \). Throughout the derivations in this work, we assume that \( \delta(e; f) = 0 \) which implies that \( \tilde{p}(e; f) = Pr(e; f) \), unless otherwise stated.\(^3\)

User navigation between nodes models the user as follows. Prior to beginning navigations, the user consults the output and visits the node retrieved from the output. The user stops navigating in a known number of steps at which point the user visits the final destination node. The user then abandons the document and returns to consulting the output. In our model presented thus far, we have modelled the user navigating in a single step. However, in general, we may consider any number of steps by combining our single step probabilities into a transition matrix where our nodes denote states and our probabilities denote the transition probabilities between our states. We can determine the probability of navigation terminating in a state after a given number of steps \( n \) by multiplying the transition matrix with itself \( n \) times. After a large number of steps, the probabilities may converge and we call these steady-state probabilities. In our experimental results in Chapters 4 and 6, we used steady-state probabilities for navigation.

Originally, in Ali, Consens, Kazai & Lalmas [7], we stated that our work uses the same probability for the user’s navigation between nodes \( \tilde{p}(e; f) = P(f \leadsto e) \) as PRUM [94]. This is incorrect. \( P(f \leadsto e) \) in PRUM is different from \( \tilde{p}(e; f) \) in Equation 3.12.

\(^3\)In our original work in Ali, Consens, Kazai & Lalmas [7], we write our equation for \( \tilde{p}(e; f) \) in Equation 3.12 in a different form (using constraint-based cases of \( e \neq f \) and \( e = f \)) than what we use here. However, the two equations are equivalent.
is the probability that a user who has seen element \( f \) has also seen element \( e \). Whereas, \( \tilde{p}(e; f) \) is the probability that a user who navigates from element \( f \) will navigate to element \( e \). The difference lies in how user navigation is assessed in user studies. \( P(f \rightsquigarrow e) \) is determined by asking the reader post-assessment whether specific ideal elements were seen or not. In contrast, \( \tilde{p}(e; f) \) is determined by tracking the reader’s attention and assuming that navigation is independent of relevance.

### 3.3.3 Navigation Between Trees

Akin to our previous section, navigation \( \tilde{p}(t_i; t_j) \) is the probability that tree \( t_i \) has been seen given a navigation from tree \( t_j \) either because of the navigation (i.e. in visit after given navigation) or because our trees overlap (i.e. in visit before given navigation). Our navigation is \( \tilde{p}(t_i; t_j) = Pr(\text{Visit} \lor \text{Navigation}) \) where \( Pr(\text{Visit}) \) is the probability that \( t_i \) has been seen before our navigation and \( Pr(\text{Navigation}) \) is the probability that our user sees tree \( t_i \) after navigating from the given tree \( t_j \).

Let us derive \( Pr(\text{Visit}) \). We may see a current tree before navigating from a given tree because the nodes in our current tree appear in the given tree. In our work, for simplicity, we approximate our before probability \( P(\text{Visit}) \) using an heuristic that captures tree overlap. Our heuristic is

\[
Pr(\text{Visit}) \approx \delta(t_i; t_j) = |t_j \cap t_i|/|t_i|
\]

where \( |t_j \cap t_i| \) is the number of overlapped nodes in \( t_i \) and \( t_j \). Our heuristic captures the following cases. If none of the nodes in our destination tree \( t_i \) are contained in our source tree \( t_j \) then \( Pr(\text{Visit}) = 0 \). If all of the nodes in our destination tree \( t_i \) are contained in our source tree \( t_j \) then \( Pr(\text{Visit}) = 1 \). If all of the nodes in our destination tree \( t_i \) are not contained in our source tree \( t_j \) then \( Pr(\text{Visit}) < 1 \) because we do not assume for \( Pr(\text{Visit}) \) here that the user navigates out of subtree \( t_j \). We consider our definition here analogous to our definition of \( Pr(\text{Visit}) \) for nodes in Definition 3.3.1. We acknowledge
that our approach to calculate $Pr(\text{Visit})$ is crude, at best, and should be revised in future work.

Let $Pr(\text{Navigation})$ denote the probability that the user sees tree $t_i$ after navigating from tree $t_j$. We next derive the conditional probability of navigating to tree $t_i$ given a navigation from tree $t_j$ which equals $Pr(\text{Navigation})$.

Let $Pr(t_i; t_j)$ denote the conditional probability of navigating to the region of the document bounded by tree $t_i$ given a navigation from the region of the document bounded by tree $t_j$ where $t_i$ and $t_j$ are both subtrees of document $T$. To calculate probability $Pr(t_i; t_j)$ precisely, we would observe whether users navigate between the two regions in the document with the methodology described above in the previous section for testing navigation between nodes (to determine marginal, joint, and conditional probabilities). However, in practice, researchers are unlikely to collect navigational observations for trees more complex than singletons because testing navigation between every possible pair of subtrees in a document is not feasible (e.g. in the Wikipedia collection, the average document contains 161 nodes [37]).

Instead, let us adapt the node-level observations described above for Equation 3.11 to calculate tree navigation. Let $t_i = \{e_1, e_2, \ldots\}$ and $t_j = \{f_1, f_2, \ldots\}$ denote a pair of trees. Let $Pr(t_j)$, $Pr(t_i \leftarrow t_j)$, and $Pr(t_i; t_j)$ denote our marginal, joint and conditional probabilities of navigation, respectively. We can calculate our marginal probability of navigating from tree $t_j$ by summing the node-level marginal probabilities,

$$Pr(t_j) = Pr(\forall f \in t_j) = \sum_{f \in t_j} Pr(f).$$

Similarly, we calculate our joint probability by summing the joint probabilities of all pairs of nodes from the two trees,

$$Pr(t_i \leftarrow t_j) = Pr(\forall e \in t_i \vee \forall f \in t_j e \leftarrow f) = \sum_{e \in t_i} \sum_{f \in t_j} Pr(e \leftarrow f).$$

Thus, our conditional probability of navigation to tree $t_i$ given a navigation from tree $t_j$
Chapter 3. Pillars of SDR Evaluation

is

\[ Pr(t_i; t_j) = \frac{Pr(t_i \leftarrow t_j)}{Pr(t_j)} = \frac{\sum_{e \in t_i} \sum_{f \in t_j} Pr(e \leftarrow f)}{\sum_{f \in t_j} Pr(f)}. \]

A key problem arises in calculating our conditional probability \( Pr(t_i; t_j) \) if the joint and marginal probabilities are not known. If our node-level navigation model includes at least the conditional probability \( Pr(e; f) \), then we can address this problem by using the following maximum log-likelihood estimate (MLE) of \( Pr(t_i; t_j) \).

First, to make our estimate, we assume that our observations of trials \((e_i \leftarrow e_j)\) are independent and identically distributed (iid). Second, we make a strong simplifying assumption that, from every node in the collection, we have collected a large number of observations of users originating navigations from it. Let the number of navigations \( \hat{N}_T \) from each node \( e_j \) in our observations \((e_i \leftarrow e_j)\) be sufficiently large such that \( \hat{N}_T \approx \sum_k n_{kj} \) for all nodes \( e_j \). In essence, we are making a uniform assumption in that we assume that the number of observations of navigations from each node is very large and that it is about equal for all nodes. However, we do not make this assumption for the number of observations of navigations to each node.

Let us now estimate our marginal, joint and conditional probabilities for node-level navigation by replacing \( \sum_k n_{kj} \) with our large number \( \hat{N}_T \). The estimate of our marginal probability, for any node \( e_j \), is

\[ \hat{P} = Pr(e_j) = \frac{\sum_k n_{kj}}{N_T} \approx \frac{\hat{N}_T}{N_T}. \]

Our joint probabilities remain unchanged,

\[ Pr(e_i \leftarrow e_j) = \frac{n_{ij}}{N_T}. \]

We estimate our conditional navigational probability to be

\[ Pr(e_i; e_j) \approx Pr(e_i \leftarrow e_j)/\hat{P}. \]

This completes our MLE. Our marginal probability for tree \( t_j \) may now be approximated
as follows.

\[ Pr(t_j) = \sum_{f \in t_j} Pr(f) \approx |t_j| \times \hat{P}. \quad (3.14) \]

Now, let us substitute our MLE for marginal probability for trees \( Pr(t_j) \) and conditional probability for nodes \( Pr(e; f) \) into the conditional probability for trees \( Pr(t_i; t_j) \).

\[
Pr(t_i; t_j) \approx \frac{\sum_{e \in t_i} \sum_{f \in t_j} Pr(e \leftarrow f)}{\sum_{f \in t_j} Pr(f)}
\]

\[
\approx \frac{1}{|t_j|} \times \sum_{e \in t_i} \sum_{f \in t_j} \frac{Pr(e \leftarrow f)}{\hat{P}}
\]

\[
\approx \frac{1}{|t_j|} \times \sum_{e \in t_i} \sum_{f \in t_j} Pr(e; f)
\quad (3.15)
\]

We can now define user navigation between trees.

**Definition 3.3.2 (User Navigation Between Trees)** The probability that tree \( t_i \) is seen after having navigated from tree \( t_j \) is as follows:

\[
\tilde{p}(t_i; t_j) = Pr(Visit \lor Navigation)
= Pr(Visit) + (1 - Pr(Visit)) \times Pr(Navigation)
\]

\[
\approx \delta(t_i; t_j) + (1 - \delta(t_i; t_j)) \times Pr(t_i; t_j)
\]

\[
\approx \delta(t_i; t_j) + (1 - \delta(t_i; t_j)) \times \frac{1}{|t_j|} \times \sum_{e \in t_i} \sum_{f \in t_j} Pr(e; f)
\quad (3.17)
\]

where \( t_i \) and \( t_j \) are trees, and \( Pr(e; f) \) in Equation 3.11 is the conditional probability of navigating to node \( e \) given a navigation from node \( f \). If \( \delta(t_i; t_j) = 0 \) then \( \tilde{p}(t_i; t_j) = 0 \).
$Pr(t_i; t_j)$. Throughout the derivations in this work, we assume that $\delta(t_i; t_j) = 0$ which implies that $\tilde{p}(t_i; t_j) = Pr(t_i; t_j)$, unless otherwise stated. We adopt the $\tilde{p}$ nomenclature to remind us that navigation includes whether our current and given trees overlap.\(^4\)

User navigation between trees models the user as follows. Prior to navigation, the user consults the output and retrieves a tree. The user visits all nodes in the retrieved tree. The user then navigates a known number of steps from a node in the retrieved tree to immediately visit a final destination node. If the destination node is any of the nodes in our destination tree then we assume that the user will visit all nodes in the destination tree.

### 3.3.4 Overlap in Tree Retrieval

A key difference between tree retrieval and other SDR tasks is how overlap is accounted. Let us first consider how overlap occurs in element, passage and tree retrieval, respectively.

In passage retrieval, overlap occurs when the text in two or more passages intersect. We assume that the user sees all of the text contained in each retrieved passage. In element retrieval, overlap occurs when two or more nodes are retrieved from the same branch of an XML document. We assume that the user sees the text contained in the element and the text it encapsulates in descendant elements. In tree retrieval, overlap occurs when one or more retrieved trees contain the same nodes. Unlike in element retrieval, nodes in trees do not contain the text of descendant nodes. In our work, for each node in the tree, we assume that the user sees the text contained in the node but not encapsulated by descendant nodes.

For instance, in tree retrieval, if the root node for a document is retrieved then we do not assume that the user will see all of the text in the complete document. We assume

\[^4\text{Our navigation in this work revises our original work in Ali, Consens, Kazai & Lalmas [7] where user navigation between trees is stated to be } \tilde{p}(t_i; t_j) = \frac{\sum_{e \in t_i} \sum_{f \in t_j} \tilde{p}(e; f)}{|t_j| |t_i|} .\]
that the user sees the text in the root node and may navigate to descendant nodes in
the document. To represent analogously in tree retrieval how overlap occurs in element
and passage retrieval, the system would need to retrieve the same node in two or more
different trees. For instance, a user would experience overlap if the system output a leaf
node in rank 1 and the induced sub-tree of the branch defined by the root node and the
leaf node in rank 2.

In the collection, the mapping of document text to nodes is made based on the
assessment methodology for collecting relevance judgments. In the Wikipedia collection,
the judgments on relevant text were largely collected at the document, section, and
subsection level of detail based on highlighted passages of text because otherwise there
were problems with obtaining assessor agreement. Tags within passages, like presentation
tags (such as bold, italics, etc.), were largely ignored when storing the results of relevance
judgments.

The way that a user experiences overlap is defined by de Vries, Kazai & Lalmas [36]
as a user preference that specifies whether the user considers effort spent to see relevant
information in overlapped sub-documents as wasted effort. For instance, in the measures
HiXEval and XCG, overlap is accounted for as an explicit penalty on gain from relevant
text contained in passages or elements that overlap. In tree retrieval, we do not account
for overlap explicitly but note that our navigation captures the effect of overlap as a type
of redundancy.

In this work, we do not explore the best or most appropriate representation of docu-
ments as trees. This is an important issue that we leave for future work. Tree retrieval
can capture overlap albeit limited to nodes in trees. Our approach allows for accounting
for overlap on the same branch by retrieving different sub-trees off of the same branch in
the document. However, unlike XCG and HiXEval, we also capture the case where nodes
on the same branch should not be counted as overlapping (such as in long documents or
the retrieval of the root node).
3.3.5 Review of Navigation

Let us review the presentation of user navigation in our previous sections. In Section 1.1, we illustrated how user navigation can lead to redundancy in SDR. In Section 3.1.2, we showed how user navigation is part of the user’s information seeking behaviour in SDR. In Section 3.3.1, we presented our probabilistic model of user navigation as a random walk of nodes in the collection. We defined the possible navigational steps between nodes in the collection in a navigation graph where the edges are weighted such that navigation over an edge that requires less effort has a greater weight than an edge requiring more effort. Additionally, we show how we can simplify our model by defining the navigation graph over partitions of the nodes in the collection.

In Section 3.3.2 in Equation 3.12, we presented user navigation between nodes \( \tilde{p}(e_i; e_j) \) which is the probability of seeing the destination node \( e_i \) given a navigation from the source node \( e_j \) (denoted as \( e_i \leftarrow e_j \) and analogously \( e_j \leftarrow e_i \) in PRUM). Probability \( \tilde{p}(e_i; e_j) \) considers that the user has either seen the destination node before making the navigation (because the two nodes are the same node) or after having navigated a given number of steps we assume that the user visits our destination node. Finally, in Section 3.3.3, we formulated navigation between trees \( \tilde{p}(t_i; t_j) \) (Equation 3.17) as a function of navigation between nodes. In the next section, we will use navigation to calculate our final pillar redundancy.

3.4 Redundancy

Redundancy, as defined by de Vries, Kazai & Lalmas [36], occurs in SDR when a user sees the same relevant information more times than they tolerate. In our work, information seen more than once is redundant to the user and considered not relevant. In this section, we elaborate on the probability \( p(a; R) \), in Equations 3.8, 3.9 and 3.10 by which we consider redundancy in our calculations of the user’s expected gain in relevance value.
Previously, in Section 3.2, we introduced our pillar of relevance and defined the user’s expected gain in relevance value using a statistical model of the user’s information-seeking process without clarifying redundancy. A summary of our statistical model is shown in Table 3.4 on page 64. Next, in Section 3.3, we introduced our pillar of user navigation which we model with the probabilities \( \tilde{p}(e_i; e_j) \) in Equation 3.12 for navigating between nodes and \( \tilde{p}(t_i; t_j) \) in Equation 3.17 for navigating between trees.

Let us begin our elaboration on redundancy by calculating our probability \( p(a; R) \) in terms of our probabilities \( \tilde{p}(e_i; e_j) \) and \( \tilde{p}(t_i; t_j) \) that model user navigation.

**Example 3.4.1** Figure 3.1 on page 40 shows the document tree of a book encoded in XML. This document contains three chapters (nodes 15, 16, and 17). Let \( R = e_4, e_{15}, e_{16} \) denote a ranked list corresponding to a system outputting node 4, node 15 and node 16 from our book. Based on consulting the output, we assume that the user will first visit \( e_4 \), then \( e_{15} \), and finally \( e_{16} \). Additionally, the user may seek relevant information by navigating into the book between each consultation. Consider the user’s visit to node \( e_{16} \), the user’s browsing history will contain the higher-ranked nodes \( R_2 = e_4, e_{15} \). The probability that the content in \( e_{16} \) has been seen after navigating from \( e_4 \) is \( \tilde{p}(e_{16}; e_4) \) (and similarly \( \tilde{p}(e_{16}; e_{15}) \) after navigating from node \( e_{15} \)). So, for node \( e_{16} \) to not be seen, prior to consulting it in the output, the probability is \( (1 - \tilde{p}(e_{16}; e_4)) \cdot (1 - \tilde{p}(e_{16}; e_{15})) \).

The probability \( p(e_{16}; R_2) \) that node \( e_{16} \) has been seen from the higher-ranked nodes is \( p(e_{16}; R_2) = 1 - (1 - \tilde{p}(e_{16}; e_4)) \cdot (1 - \tilde{p}(e_{16}; e_{15})) \).

Next, let us consider the following example of calculating redundancy where the user navigates between trees.

**Example 3.4.2** Figure 3.4 on page 65 shows the system output \( R = t_l, t_r \) of two book XML subtrees from the document in the previous example. Consider calculating the probability \( p(t_r; R_1) \) of whether tree \( t_r \) has been seen by navigating from the higher-ranked tree \( t_l \) in the output. Let the navigation between nodes probability be \( \tilde{p}(e; f) = 0.05 \) for
### Chapter 3. Pillars of SDR Evaluation

#### Table 3.4: Summary of statistical model for SDR gain

<table>
<thead>
<tr>
<th>Notation</th>
<th>Type, Value</th>
<th>Meaning</th>
</tr>
</thead>
<tbody>
<tr>
<td>$A$</td>
<td>set of sub-trees</td>
<td>Sub-trees assessed for relevance. See Def. 3.2.1</td>
</tr>
<tr>
<td>$R$</td>
<td>ranked list of sub-trees</td>
<td>Output of a system, browsing history. See Def. 3.2.4</td>
</tr>
<tr>
<td>$k$</td>
<td>integer, $k =</td>
<td>R</td>
</tr>
<tr>
<td>$a$</td>
<td>a sub-tree</td>
<td>For a single trial, $a$ is a fixed element of $A$.</td>
</tr>
<tr>
<td>$m$</td>
<td>integer</td>
<td>Rank position of $a$ in $R$.</td>
</tr>
<tr>
<td>$rel(a)$</td>
<td>scalar-valued function</td>
<td>The judged relevance value of the sub-tree $a$. Always greater than 0 for a relevant sub-tree, and equal to 0 for a non-relevant sub-tree. See Def. 3.2.2</td>
</tr>
<tr>
<td>$C, \overline{C}$</td>
<td>conditions</td>
<td>$C$ is the condition that sub-tree $a$ can be seen ($\overline{C}$ for cannot be seen) by consulting it from $R$. See Def. 3.2.5</td>
</tr>
<tr>
<td>$N, \overline{N}$</td>
<td>events</td>
<td>$N$ is the event that sub-tree $a$ seen ($\overline{N}$ for not seen) by navigating to it from $R$. See Def. 3.2.7</td>
</tr>
<tr>
<td>$\Omega_C, \Omega_{\overline{C}}$</td>
<td>set of all possible outcomes, $\Omega_C = \Omega_{\overline{C}} = {N, \overline{N}}$</td>
<td>Outcomes of an information seeking process, where a user seeks the sub-tree $a$ by interacting with a system that outputs $R$. See Eq. 3.1</td>
</tr>
<tr>
<td>$X_C, X_{\overline{C}}$</td>
<td>random variables, $X_C = x_C \in \Omega_C$, $X_{\overline{C}} = x_{\overline{C}} \in \Omega_{\overline{C}}$</td>
<td>Random outcomes of the navigating to see fixed sub-tree $a_C$ (and $a_{\overline{C}}$, resp.), dependent on the user’s information seeking process. See Def. 3.2.8</td>
</tr>
<tr>
<td>$\tilde{p}(e; f)$</td>
<td>$= \delta(e; f) + Pr(e; f)$ $- \delta(e; f) \times Pr(e; f)$</td>
<td>The probability that node $e$ is seen after navigating from node $f$. See Def. 3.3.1</td>
</tr>
<tr>
<td>$p(a; R)$</td>
<td>$= 1 - \prod_{j=1}^{k} (1 - \tilde{p}(a; t_j))$</td>
<td>The probability that a user navigates to sub-tree $a$ from one or more entries in $R$. See Eq. 3.18</td>
</tr>
<tr>
<td>$P(X_C = x_C)$</td>
<td>$= \begin{cases} p(a; R_{m-1}) &amp; \text{if } x = N, \ 1 - p(a; R_{m-1}) &amp; \text{if } x = \overline{N}, \end{cases}$</td>
<td>Probability distribution of $X_C$. See Eq. 3.6</td>
</tr>
<tr>
<td>$P(X_{\overline{C}} = x_{\overline{C}})$</td>
<td>$= \begin{cases} p(a; R) &amp; \text{if } x = N, \ 1 - p(a; R) &amp; \text{if } x = \overline{N}, \end{cases}$</td>
<td>Probability distribution of $X_{\overline{C}}$. See Eq. 3.7</td>
</tr>
<tr>
<td>Hit($X_C$)</td>
<td>$= rel(a)$ if $N$ occurs, else 0.</td>
<td>Hit is a random variable (a function of random variable $X_C$) that represents a user’s gain in relevance value from a sub-tree in $A$ that is a “hit”, i.e. a relevant sub-tree that is in $R$. See Eq. 3.3</td>
</tr>
<tr>
<td>Miss($X_{\overline{C}}$)</td>
<td>$= rel(a)$ if $\overline{N}$ occurs, else 0.</td>
<td>Miss represents gain in relevance value from a sub-tree in $A$ that is a “miss”, i.e. a relevant sub-tree that is not in $R$ and is not navigated to. See Eq. 3.5</td>
</tr>
<tr>
<td>Near($X_{\overline{C}}$)</td>
<td>$= rel(a)$ if $N$ occurs, else 0.</td>
<td>Near represents gain in relevance value from a sub-tree in $A$ that is a “near-miss”, i.e. a relevant sub-tree that is not in $R$, but is navigated to. See Eq. 3.4</td>
</tr>
<tr>
<td>$E[\text{Hit}(X_C)]$</td>
<td>$= rel(a) \times (1 - p(a; R_{m-1}))$</td>
<td>Expected gain from a hit at rank $m$. See Eq. 3.8</td>
</tr>
<tr>
<td>$E[\text{Miss}(X_{\overline{C}})]$</td>
<td>$= rel(a) \times (1 - p(a; R))$</td>
<td>Expected unrealized gain from a miss. See Eq. 3.9</td>
</tr>
<tr>
<td>$E[\text{Near}(X_{\overline{C}})]$</td>
<td>$= rel(a) \times p(a; R)$</td>
<td>Expected gain from a near-miss. See Eq. 3.10</td>
</tr>
</tbody>
</table>
all nodes. Our subtrees \( t_l \) and \( t_r \) overlap at nodes 1 (bk), 2 (fm), and 14 (bd). There are 30 pairwise possible ways to see nodes in \( t_r \) from \( t_l \) because the subtrees contain 6 and 5 nodes each, respectively. We apply Definition 3.3.2 to calculate our navigation between trees probability \( \tilde{p}(t_r; t_l) \) as follows: \( |t_l| = 6, \ |t_r| = 5, \ \sum_{e \in t_r} \sum_{f \in t_l} \tilde{p}(e; f) = 30 \times 0.05 = 1.5, \delta(t_r; t_l) = |t_r \cap t_l| / |t_l| = 3 / 5 = 0.6, \) and, thus, \( \tilde{p}(t_r; t_l) = 0.6 + (1 - 0.6) \times 1.5 / 6 = 0.6 + 0.4 \times 0.25 = 0.7. \) Therefore, our desired probability is \( p(t_r; R_1) = 1 - (1 - \tilde{p}(t_r; t_l)) = \tilde{p}(t_r; t_l) = 0.7. \)

We define the redundancy of a tree (whether singleton or otherwise) as whether the region of text bounded by the tree has been seen more times than the user tolerates. We measure redundancy by the probability of the tree being seen given the user’s browsing history of search results and how the user prefers to navigate within the collection to find relevant information.

In Section 3.2, we defined redundancy as the conditioning probabilities for expected gains from hits, near-misses and misses (unrealized gain). We recall that \( p(a; R) \) denotes the probability \( P(a \text{ is seen once}; R) \) that tree \( a \) is seen at least once by navigating from the trees in the output \( R \). Assume that the navigation from each tree in the output is independent. From the above examples, we can calculate \( p(a; R) \) for any tree (singleton or otherwise), as follows

\[
p(a; R) = 1 - \prod_{j=1}^{k} (1 - \tilde{p}(a; t_j)) \tag{3.18}
\]
where \( a \) is a tree in the collection, \( R = t_1, t_2, \ldots, t_k \) is an output of \( k \) trees, and \( \tilde{p}(a; t_j) \) is shown in Equation 3.17. As an aside, we have not considered weakly ordered lists, as defined by Raghavan, Bollmann & Jung [98], where the user randomly accesses search results in tied ranks. The calculation of redundancy given weak ordering is shown in Appendix A. Additionally, in defining redundancy, we have assumed that the user tolerates only seeing the same content once. In Appendix B, we show the effect on our redundancy calculation where the user may tolerate seeing the same information more than once (as defined in de Vries, Kazai & Lalmas [36]). This completes our pillars of relevance, navigation, and redundancy.

### 3.5 Example Toy System and Models

We now present a toy example of an SDR test collection based on our pillars. We revisit this toy example later, in Section 5.1, to demonstrate the calculation of our Extended Structural Relevance (ESR) framework and later still, in Section 5.3, to demonstrate performance evaluation with IR measures formulated in ESR.

For a given query, consider the retrieval of elements from the article shown in Figure 3.2(a) on page 49. Let the (judged) relevant elements in our article be \( e_3 \) and \( e_4 \), i.e. \( A = e_3, e_4 \). When relevance is binary, let

\[
\text{rel}(e_3) = \text{rel}(e_4) = 1.
\]

<table>
<thead>
<tr>
<th></th>
<th>( e_3 )</th>
<th>( e_4 )</th>
</tr>
</thead>
<tbody>
<tr>
<td>Binary Relevance</td>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td>Relevance by Length</td>
<td>30 characters</td>
<td>20 characters</td>
</tr>
</tbody>
</table>

Table 3.5: Relevance value of assessments \( \text{rel}(e) \), for tree \( e \) in \( A \)
Chapter 3. Pillars of SDR Evaluation

<table>
<thead>
<tr>
<th>Nodes</th>
<th>$e_1$</th>
<th>$e_2$</th>
<th>$e_3$</th>
<th>$e_4$</th>
<th>$e_5$</th>
<th>$e_6$</th>
</tr>
</thead>
<tbody>
<tr>
<td>$e_1$</td>
<td>0</td>
<td>0.53</td>
<td>0.16</td>
<td>0.11</td>
<td>0.11</td>
<td>0.11</td>
</tr>
<tr>
<td>$e_2$</td>
<td>0.63</td>
<td>0</td>
<td>0</td>
<td>0.133</td>
<td>0.133</td>
<td>0.133</td>
</tr>
<tr>
<td>$e_3$</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>$e_4$</td>
<td>0.5</td>
<td>0.5</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>$e_5$</td>
<td>0.5</td>
<td>0.5</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>$e_6$</td>
<td>0.5</td>
<td>0.5</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
</tbody>
</table>

Table 3.6: User navigation $\tilde{p}(e_i; e_j)$

When we use the number of highlighted characters to measure relevance, then let

$$rel(e_3) = 30, \text{ and } rel(e_4) = 20.$$  

The relevance values for binary relevance and relevance by length are shown in Table 3.5.

User navigation is shown in Table 3.6, and Table 3.7 shows the outputs for three systems for the query. System 1 ($R1$) retrieves a near-miss in rank 1 and hits in ranks 2 and 3. System 2 ($R2$) retrieves three near-misses. System 3 ($R3$) retrieves a near-miss in rank 2 and hits in ranks 1 and 3.

We expect System 2 to have the worst performance as it does not retrieve any hits, and System 3 to have the best performance as it retrieves a hit in rank 1, whereas System 1 does not retrieve a hit until rank 2. We expect System 1 to be the second best performing system. This completes our example toy system which demonstrates a test collection in our work.

---

5 Our example navigation probabilities are derived from a PRUM user navigation model, for hierarchical navigation, which can be found in Equation 9 (p. 22) in Piwowarski, Gallinari & Dupret [94]. In hierarchical navigation, the user navigates downwards, from parent nodes to child nodes, and never vice versa. For the current example, we use PRUM probabilities as weights on the edges of the article graph shown in Figure 3.2(a) on page 49 and obtain our user navigation as follows: $\tilde{p}(e_i; e_j) = p(e_j \leadsto e_i) / \sum_{i} p(e \leadsto e_i)$. There are, of course, other means to estimate $\tilde{p}(e_i; e_j)$. Indeed, our calculation of navigation is purely illustrative. Our intention is not to suggest that navigation in our work and PRUM share equivalent navigation models (because they do not - see Footnote 9 in Section 5.2.4).
### Table 3.7: Three example system outputs.

<table>
<thead>
<tr>
<th>Ranks</th>
<th>1</th>
<th>2</th>
<th>3</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>System 1 (R1)</strong></td>
<td>(e_1)</td>
<td>(e_3)</td>
<td>(e_4)</td>
</tr>
<tr>
<td><strong>System 2 (R2)</strong></td>
<td>(e_1)</td>
<td>(e_2)</td>
<td>(e_6)</td>
</tr>
<tr>
<td><strong>System 3 (R3)</strong></td>
<td>(e_3)</td>
<td>(e_1)</td>
<td>(e_4)</td>
</tr>
</tbody>
</table>

3.6 **Summary**

SDR encompasses many tasks, many of which (if not all) can be considered as special cases of tree retrieval. There is a significant and growing interest in tree retrieval in many areas (e.g., web IR, XML databases, book search). Yet, many of the proposals in the literature (described in Section 2.2) cannot be evaluated due to the lack of an adequate IR performance measure. The existing evaluation measures have been developed for specific SDR tasks, mainly in the context of the tasks considered at the INEX workshop. In this chapter we argued for the need of a general measure for tree retrieval, and we enumerated its requirements.

Our key contribution is the development of our pillars as a way to evaluate the general task of tree retrieval. Our approach relates how users navigate in documents to how they experience redundancy in SDR. In the next chapter, we experimentally validate our pillars by proposing a precision measure based on our pillars, developing a user navigation model for the Wikipedia collection, and finally testing the stability of our measure in thorough, focused and relevant in context tasks in element retrieval.
Chapter 4

Experimental Results for Structural Relevance

In this chapter, we present experimental results to validate our pillars of SDR evaluation. The outline of this chapter is as follows. We begin in Section 4.1 by defining, based on our pillars, the measure Structural Relevance in Precision (SRP). Next, in Section 4.2 we approximate user navigation by inferring our navigation probabilities from the structural statistics of document components. In Section 4.3 we then experimentally compare SDR evaluation with SRP using empirical navigation models derived from observations of users navigating versus our approximated models derived from structural statistics. Finally, in Section 4.4 we present experimental results that validate the stability of SRP. This chapter is based on published results in Ali, Consens, & Lalmas [9], Ali, Consens, Kazai & Lalmas [7], and Ali, Consens & Larsen [10].

4.1 Structural Relevance

Structural Relevance (SR) is the user’s expected gain in relevance value from hits. We use it to define a precision-based measure for tree retrieval. We first proposed SR in Ali, Consens, & Lalmas [9] and later refined it in Ali, Consens, Kazai & Lalmas [7].
Akin to $E[\text{Hit}(X_C)]$ in Equation 3.8 in Section 3.2, the calculation of $SR$ depends on the user’s browsing history, which is the set of trees that a user has viewed previously. We calculate $SR$ for a ranked list output as follows:

$$SR(R) = \sum_{i=1}^{k} rel_{tree}(t_i) \cdot (1 - p(t_i; R_{i-1}))$$ (4.1)

where $R = t_1, t_2, \ldots, t_k$ is a ranked list of $k$ subtrees, $t_i$ is the $i$-th subtree in $R$, $R_{i-1}$ is the sublist of $R$ up to rank $i - 1$, $rel_{tree}(t_i) \in [0, 1]$ is the inferred relevance value of subtree $t_i$ (explained below in Equation 4.2), and $p(t_i; R_{i-1})$ is the probability that a user visits tree $t_i$ by navigating from one or more of the trees in the browsing history $R_{i-1}$ (defined in Equation 3.17).

The main difference between $SR$ and $E[\text{Hit}(X_C)]$ (Equation 3.8 in Section 3.2) is the relevance value function. We go into more detail on this issue in Section 5.2.1. In $E[\text{Hit}(X_C)]$, the relevance value calculation is for a tree that has been judged for relevance. Whereas, here we use inferred relevance value which is the relevance value of trees in the output inferred from node-level judgments. Our inferred relevance value function $rel_{tree}(t)$ is the average relevance value of the nodes in an output tree:

$$rel_{tree}(t) = \frac{\sum_{e \in t} rel(e)}{|t|}$$ (4.2)

where $t$ is a tree, $e$ is a node in the tree, $|t|$ is the number of nodes in the tree, and $rel(e)$ is the relevance value of node $e$. For a given query, let each node $e$ have a relevance value $rel(e)$, where for a fully relevant node, $rel(e) = 1$; for a fully irrelevant node, $rel(e) = 0$; and for all other cases, $0 < rel(e) < 1$.

Other formulations for inferring relevance value could be used such as the maximum relevance value of nodes. However, we did not pursue comparing other formulations for inferred relevance value because the average was simple to understand and sufficient for the present work.

The relevance of a tree is difficult to assess. Not only is it a complex task (in the
context of INEX, Piwowarski, Trotman, & Lalmas [96] show that the complexity of the assessment process affects the reliability of the assessment data), but there are typically many different trees that can equivalently represent the same instance of relevant information. It is therefore more practical and in fact recommended here to determine the relevance of a tree based on the relevance of its nodes. How to assess the relevance of nodes in SDR has been studied within INEX (see Piwowarski, Trotman & Lalmas [96]), where now a stable and reliable process has been established.

$SR$ is limited to precision-based measurement because inferred relevance value only accounts for gain from relevant nodes in trees in the output, i.e. hits. $SR$ is related to $PRUM$ in that both measures account for navigation and redundancy. PRUM measures near-misses vis á vis the number of ideal elements that a user sees from the output. Whereas, $SR$ measures hits in terms how redundancy reduces gain for relevant nodes in the output.

We use $SR$ to evaluate precision by replacing the number of hits in precision with $SR$ as follows.

$$SRP = SR/k$$

where $SR$ is given by Equation 4.1 and $k$ is the size of the ranked list output. $SRP$ is limited to evaluation at rank cut-offs.

### 4.2 Approximating Navigation Using Summaries

To use $SRP$ in evaluation, we require both relevance assessments of nodes and a user navigation model. INEX test collections provided us with assessments however not a user navigation model. In Section 3.3 we showed how a model of user navigation can be determined from observations collected in user studies. However, this approach comes with three unavoidable challenges. First, navigation models derived from observations of users are specific to the study topics, tasks and collections. Second, for a large col-
lection, it is often not economic to observe user navigation for all possible pairs of nodes in the collection. Third, in lieu of a complete set of observations of navigation for all pairs of nodes in the collection, user navigation must be inferred for the entire collection based on an incomplete set of observations. To address the above problems (specificity, economy, and incompleteness) in empirically derived navigation models, we propose that researchers exploit the structure of XML documents to infer navigation from the summary statistics of document components in the collection. This section is based on our previously published work in Ali, Consens & Larsen [6].

A *summary* is a partitioning scheme of the nodes in the trees in the collection based on node-level path labels. Our approach is based on AxPRE summaries as presented in Consens, Rizzolo & Vaisman [34]. AxPRE summaries define a broad range of the different XML structural summaries available in the literature using an axis path regular expression (AxPRE) language that is capable of describing a plethora of partitioning schemes. AxPRE is an extension of regular expressions for specifying classes of hierarchical paths based on child-parent relationships in a tree. For example, a $p^*$ summary partitions the XML elements based on their incoming paths, since $p^*$ is the axis path regular expression describing paths of parent ($p$) axis traversals. A refinement to this is a $p^*|c$ summary, which partitions elements based on their incoming paths and the labels of their children,
since $p^*|c$ is the axis path regular expression describing paths of parent axis traversals unioned with a single child ($c$) axis step. Conversely, a less refined $c$ summary, partitions elements only by the labels of their child elements, and has the same level of detail as the information given in the content model of the document’s DTD or XML Schema.

Consider the XML document shown in Figure 4.1 (which is our book XML document used earlier in Figure 1.1 in Chapter 1) where, for each node, its label is shown in the node, its node identifier is shown in bold, and the length of the text contained in it is shown in parentheses. For instance, node 17 is labelled $c$ and has a length of 50. Figure 4.2 shows two different summary graphs of our document ($p^*$ and $p^*|c$ summaries, respectively). Each partition in a summary is associated to a node in the summary graph, and is referred to using a summary node identifier (SID). Figure 4.2 is labeled with the SID in bold and the extent of the partition in curly braces. For instance, /bk/fm/d/m in the $p^*$ summary in Figure 4.2 (left) is SID $S5$ whose extent is nodes 5 and 6. The number of elements in each partition is the extent size.

The structure of an XML document often matches how users navigate the information encoded in it. For instance, Laender et. al. [68] noted that Pareto’s rule applies to XML schema descriptions (XSD), in that 80% of schema definitions are context (i.e. helps the user to navigate to information) in nature and 20% are data definitions. Barbosa, Mignet & Veltri [17] found, in a large-scale sampling of actual XML documents from the
### Table 4.1: Steady-state probabilities in \( p^* \) summary weighted by extent size of children

<table>
<thead>
<tr>
<th></th>
<th>S1</th>
<th>S2</th>
<th>S3</th>
<th>S4</th>
<th>S5</th>
<th>S6</th>
<th>S7</th>
</tr>
</thead>
<tbody>
<tr>
<td>S1</td>
<td>0</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>0</td>
</tr>
<tr>
<td>S2</td>
<td>1</td>
<td>0</td>
<td>2</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>S3</td>
<td>0</td>
<td>2</td>
<td>0</td>
<td>6</td>
<td>2</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>S4</td>
<td>0</td>
<td>0</td>
<td>6</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>S5</td>
<td>0</td>
<td>0</td>
<td>2</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>S6</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>3</td>
</tr>
<tr>
<td>S7</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>3</td>
<td>0</td>
</tr>
</tbody>
</table>

\( \pi^m \) 0.07 0.1 0.33 0.2 0.07 0.13 0.10

Table 4.1: Steady-state probabilities in \( p^* \) summary weighted by extent size of children web, that over 50% of XML markup contextualized information and that the nodes of the XML document tree at depths of less than 5 tended to be context as opposed to data-centric by almost a factor of 10 : 1. Summaries offer an efficient and portable way to estimate navigation when the document structure corresponds to how users navigate.

We select summary node statistics, such as the extent size or content length of nodes, to weight the edges of the summary graph. We approximate user navigation by using our weights to derive transition probabilities that model step-wise navigations as described earlier in Section 3.3.2. We call these summary navigation models. In our work, we bi-directionally weight the edges between parent and child nodes in the summary graph with a summary statistic of the child summary node. All other weights are zero. We note that our summary graphs contain no edges where the vertices of our edges are the same summary node.

We use the weighting matrix to derive single-step transition probabilities (in a transition matrix) by dividing each row by its sum. If we want to model a user who navigates \( n \) steps, then we can iteratively multiply our transition matrix with itself \( n \) times. After
Chapter 4. Experimental Results for Structural Relevance

Table 4.2: Steady-state probabilities in $p^*|c$ summary weighted by character length

<table>
<thead>
<tr>
<th></th>
<th>S1</th>
<th>S2</th>
<th>S3</th>
<th>S4</th>
<th>S5</th>
<th>S6</th>
<th>S7</th>
<th>S8</th>
<th>S9</th>
</tr>
</thead>
<tbody>
<tr>
<td>S1</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>S2</td>
<td>0</td>
<td>0</td>
<td>10</td>
<td>3</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>S3</td>
<td>0</td>
<td>10</td>
<td>0</td>
<td>133</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>S4</td>
<td>0</td>
<td>3</td>
<td>133</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>S5</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>14</td>
<td>73</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>S6</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>14</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>S7</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>73</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>S8</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>73</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>221</td>
</tr>
<tr>
<td>S8</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>73</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>221</td>
<td>0</td>
</tr>
</tbody>
</table>

$\pi^{io}$

|     | 0   | 0.01| 0.14| 0.13| 0.22| 0.01| 0.07| 0.21| 0.21|

A large number of steps, the rows in our iteratively multiplied transition matrix may converge to a single value. We call these steady-state probabilities. In our work, we use steady-state probabilities however we note that these do not exist for all navigation models. In the next two examples, we demonstrate two summary navigation models.

Example 4.2.1 Let us approximate user navigation between nodes using the summary statistic of extent size (number of elements in the summary node partitions) for the incoming $p^*$ summary (shown in Figure 4.2 on the left) of our book XML document shown in Figure 4.1. We weight the edges of the graph in Figure 4.2 (left) with the extent of the child node for each pair of nodes. Table 4.1 shows the weighting matrix for our model and the resulting steady-state probabilities $\pi^{in}$ in the last row. Consider calculating navigation from node $e_{14}$ in summary node $S6$ to $e_{15}$ in summary node $S7$. If the user’s navigation ends in a single-step, then from row $S6$ in Table 4.1 we calculate...
\[ \tilde{p}(e_{15}; e_{14}) \approx \tilde{p}(S7; S6) = 3/4 = 0.75. \] The steady-state probability is shown in the last row of Table 4.1 and is \[ \tilde{p}(e_{15}; e_{14}) \approx \pi_{io}^{in} = \pi_{S7}^{in} = 0.1. \]

**Example 4.2.2** Next, let us approximate user navigation between nodes \( \tilde{p}(e_{15}; e_{14}) \) using the summary statistics of length of content in nodes for the incoming-outgoing \( p^*|c \) summary (shown in Figure 4.2 on the right) of our book XML document shown in Figure 4.1. Table 4.2 shows the weighting matrix for our model and the resulting steady-state probabilities \( \pi^{io} \) in the last row. For instance, node \( e_{15} \) is contained in the extent of summary node \( S9 \). We obtain our steady-state probability for navigation from Table 4.2 as follows \[ \tilde{p}(e_{15}; e_{14}) \approx \pi_{io}^{in} = \pi_{S9}^{io} = 0.21. \]

Overall, we note that summary models are not easy to interpret and much work still needs to be done to address what types of summaries and statistics accurately capture navigation behaviour.

### 4.3 Experimental Results for Navigation

In this section, we compare scoring performance using empirical user navigation models versus our summary navigation models (presented in the last section). Our results are from a study conducted in the Interactive Track at INEX 2006.

#### 4.3.1 Interactive Track at INEX 2006

The Interactive Track at INEX\(^1\) is an international, multi-year user study to identify XML elements that are helpful in solving given search tasks. The 2006 user study consisted of 83 participants for 12 topics with observations of user activity recorded for 818 documents from the INEX 2006 Wikipedia collection.\(^77\)

---

\(^{1}\)INitiative for the Evaluation of XML retrieval (INEX): http://www.inex.otago.ac.nz
Chapter 4. Experimental Results for Structural Relevance

Figure 4.3: User navigation graph for Interactive Track at INEX 2006.

The Interactive Track prototype XML retrieval system is a scrollable window that contains a frame to display the entire article (with the returned elements highlighted in context), and a separate frame that displays the table-of-contents as a set of links. User events are time-stamped and the duration of an event is the length of time between its start time and the start time of the next event in the participant’s session. To review articles, the participant clicks on a result from a list of hyperlinks (DETAIL QUERY). The system does not track navigation via the scrolling behavior of users. It does however track whether the participant uses the table-of-contents to navigate to different parts of the article (DETAIL BROWSE). A visit to an element is defined as a participant who enters an element via a DETAIL QUERY or a DETAIL BROWSE.

The Interactive Track study models navigation over the graph shown in Figure 4.3. The participants were observed to navigate among five types of article nodes in the 2006 user study; namely, ARTICLE, SEC, SS1, SS2, and OTHER. These correspond to elements whose label paths are the root /article (ARTICLE), a section path /article/body/section (SEC), a subsection path SEC/section (SS1), a sub-subsection path SS1/section (SS2), and all other elements’ paths (OTHER). We note that our nodes, in an experimental setting, are fully inter-connected much like our example in the last chapter shown in Figure 3.2 on page 49.

Table 4.3 tabulates the visits and mean time spent in visits to elements. For instance,
Table 4.3: Number of visits (mean time spent) from Interactive Track at INEX 2006 participants visited SS2 elements and then navigated to element ARTICLE 4 times. The mean time spent in SS2 before navigating to element ARTICLE was on average 12.3 seconds. This led to an overall time, which we refer to as an episode, of 12.3 \times 4 = 49.2 seconds. The most visited element was SEC, and the largest mean time spent occurred in navigations to SEC elements from ARTICLE.

### 4.3.2 Navigation Models

Table 4.4A shows the weighting matrix, based on our observations of users in the Interactive Track, using number of visits (and mean time spent in parentheses) to weight edges. The user navigation models are summarized in Table 4.4B. We report the steady-state probabilities calculated for each model. Our models used different weighting schemes; namely, the number of visits (Visit), the overall amount of time spent in elements (Episode), and the mean time spent in each element (Time spent).
### A. Transition Matrix for Visits

<table>
<thead>
<tr>
<th>Source</th>
<th>ARTICLE</th>
<th>SEC</th>
<th>SS1</th>
<th>SS2</th>
<th>OTHER</th>
</tr>
</thead>
<tbody>
<tr>
<td>ARTICLE</td>
<td>0.0</td>
<td>0.87</td>
<td>0.11</td>
<td>0.01</td>
<td>0.01</td>
</tr>
<tr>
<td>SEC</td>
<td>0.40</td>
<td>0.54</td>
<td>0.06</td>
<td>0.0</td>
<td>0.0</td>
</tr>
<tr>
<td>SS1</td>
<td>0.31</td>
<td>0.34</td>
<td>0.0</td>
<td>0.0</td>
<td>0.01</td>
</tr>
<tr>
<td>SS2</td>
<td>0.21</td>
<td>0.11</td>
<td>0.68</td>
<td>0.0</td>
<td>0.0</td>
</tr>
<tr>
<td>OTHER</td>
<td>0.58</td>
<td>0.0</td>
<td>0.08</td>
<td>0.0</td>
<td>0.33</td>
</tr>
</tbody>
</table>

### B. Steady-state Probabilities for User Navigation Models

<table>
<thead>
<tr>
<th></th>
<th>ARTICLE</th>
<th>SEC</th>
<th>SS1</th>
<th>SS2</th>
<th>OTHER</th>
</tr>
</thead>
<tbody>
<tr>
<td>Visit</td>
<td>0.281</td>
<td>0.606</td>
<td>0.105</td>
<td>0.002</td>
<td>0.006</td>
</tr>
<tr>
<td>Episode</td>
<td>0.410</td>
<td>0.531</td>
<td>0.050</td>
<td>0.001</td>
<td>0.009</td>
</tr>
<tr>
<td>Time spent</td>
<td>0.318</td>
<td>0.209</td>
<td>0.129</td>
<td>0.028</td>
<td>0.317</td>
</tr>
</tbody>
</table>

### C. Steady-state Probabilities for Summary Navigation Models

<table>
<thead>
<tr>
<th></th>
<th>ARTICLE</th>
<th>SEC</th>
<th>SS1</th>
<th>SS2</th>
<th>OTHER</th>
</tr>
</thead>
<tbody>
<tr>
<td>Path</td>
<td>0.361</td>
<td>0.537</td>
<td>0.087</td>
<td>0.014</td>
<td>0.001</td>
</tr>
<tr>
<td>Content</td>
<td>0.103</td>
<td>0.434</td>
<td>0.089</td>
<td>0.013</td>
<td>0.361</td>
</tr>
<tr>
<td>Depth</td>
<td>0.309</td>
<td>0.435</td>
<td>0.067</td>
<td>0.008</td>
<td>0.181</td>
</tr>
</tbody>
</table>

Table 4.4: Navigation models in Interactive Track study.

For our summary navigation models, we randomly selected 2343 Wikipedia articles. We simplified the elements by removing sub-tags in the XML of our articles such that label paths were strictly the root `/article` (ARTICLE), section paths `/article/body/section` (SEC), subsection paths `SEC/section` (SS1), sub-subsection paths `SS1/section` (SS2), and all other elements’ paths (OTHER). We summarized our articles using a $p^*$ summary. The
Correlation $p$-values for $k=10$ (and $k=50$)

<table>
<thead>
<tr>
<th>Summary Models</th>
<th>path</th>
<th>content</th>
<th>depth</th>
</tr>
</thead>
<tbody>
<tr>
<td>episode</td>
<td>0.005 (0.010)</td>
<td>0.099 (0.127)</td>
<td>0.037 (0.061)</td>
</tr>
<tr>
<td>visit</td>
<td>0.004 (0.012)</td>
<td>0.111 (0.118)</td>
<td>0.054 (0.039)</td>
</tr>
<tr>
<td>time spent</td>
<td>0.109 (0.087)</td>
<td>0.033 (0.062)</td>
<td>0.043 (0.058)</td>
</tr>
</tbody>
</table>

Table 4.5: Correlation $p$-values of User Navigation Models and Summary Navigation Models for $k = 10$ (and $k = 50$).

The resulting summary graph is the same as the graph in Figure 4.3. We consider three different summary statistics for weighting our graph. Path weights are the extent size of the child summary node on each edge. Content weights are the number of characters of content in the elements in the extent of the child summary node. Finally, depth weights are the same as content but damped (divided) by the average path depth of the elements in the extent of the child summary node. Table 6.9C shows the resulting summary navigation models based on path, content and depth weights.

### 4.3.3 Results from the Interactive Track

Our experiment compares system rankings with $SRP$ parameterized with both empirical models from a user study (user navigation models) and approximated models using summaries (summary navigation models). The user and summary navigation models shown in Table 4.4B and Table 4.4C were used to evaluate Wikipedia runs using mean-average Structural Relevance in Precision.

We used the INEX 2006 Wikipedia collection with 15 systems across 107 topics in the Ad-hoc Thorough task at rank cut-offs of $k=10$ and $k=50$. We evaluated $SRP$ parameterized with the six different navigation models (visit, episode, time spent, path, content and depth). The six system rankings were then compared using Spearman’s Rho.
p-value correlations (p-value < 0.1 meant correlated rankings, and p-value < 0.05 meant strongly correlated rankings). Table 4.5 shows the p-value (correlations) between our models.2

The path model had strong correlation (p-value < 0.05) with both the episode and visit user models. The content model showed correlation (p-value < 0.1) with the time spent model. Overall, the depth model demonstrated the best results, in that, it showed correlation with all user models for both $k = 10$ and $k = 50$. These results suggest that the depth model could be used as a general user navigation model. The strength of our approach is that summary navigation models can be economically applied to new collections, because they are only validated using assessments, but calculated using summaries and structural properties of the elements in the collection.

Our results suggest that summary navigation models can be both robust and accurate in evaluation (compared to the more costly to obtain user navigation models), especially when we consider that it is not feasible to test navigation for all pairs of nodes for all topics in a large collection. We argue that our proposal is better than using arbitrary weights or heuristics. In saying this, we acknowledge the limitations of our approach but leave further refinements for future work.

4.4 Validation of SRP

Let us next present results from a validation experiment for the measure $SRP$. We limit our experiments here to passage and element retrieval because of lack of access to a tree retrieval test collection. Although our work argues for the need to go beyond passages or elements in SDR evaluation, the majority of the work on the experimental application and validation of measures for SDR falls into these simpler scenarios. With this in mind, both element and passage retrieval systems have been researched extensively at INEX...

---

2Spearman’s Rho was greater than 0.5 for all comparisons.
which provides us with years’ worth of experimental data where several measures have been applied to evaluate the retrieval effectiveness of SDR systems. The section is based on our work originally published in Ali, Consens, Kazai & Lalmas [7].

4.4.1 Experimental Setup

We evaluate three tasks in ad-hoc element retrieval: Thorough, Focused and Relevant in Context. In Thorough retrieval, the user prefers to see all relevant results (that may overlap). *Overlap*, defined in Clarke [27], occurs when retrieved nodes are from the same document on the same branch in the document. In Focused retrieval, the user prefers a single result from a set of related elements (i.e., from a branch of the document tree), where results must not overlap. The Relevant in Context task is similar to the Focused task, but results from the same document are clustered into single, adjacent rank positions in the output.

We validate stability by investigating the accuracy and fidelity of *SRP*. We define accuracy as whether a measure evaluates what it is designed to measure. We define the fidelity of a measure as whether the measure consistently evaluates better and worse system performance. Our experiments use the INEX test collection for the Wikipedia.

In this study, system performance was measured using *SRP* (Section 4.1) and XCG (Section 2.1). The resultant system rankings from *SR* (using *SRP*) and XCG (using *NXCG*) were compared to determine whether *SR* was accurate. System ranking was based on mean-averaged *SRP* across rank cut-offs with a $p^*$ summary (Section 4.2) of the INEX 2006 Wikipedia collection weighted by the extent size. To calculate *NXCG*, the official metric of INEX 2006, the generalised quantization method was used and the overlap parameter of $\alpha$ was set to 1 for all runs (i.e., overlap was punished and near-misses rewarded). The accuracy of *SR* was tested in the INEX 2006 Ad-Hoc Thorough and Focused Tasks for 26 systems across 114 topics. Our results are reported in Section 4.4.2. We used the ESREval software package presented in Appendix C to calculate *SR*.
To determine the fidelity of \( SR \), we tested \( SR \) by systematically perturbing and then comparing the evaluations of runs from both the INEX 2006 Thorough Task and the INEX 2007 Relevant in Context Task. For the Relevant in Context Task, we considered the clustering of results from the same document as induced subtrees. Kamps, Lalmas & Pehcevski [55] defined the Relevant in Context task for users who want to retrieve relevant articles where the relevant information within each article (captured by a set of XML elements) should be correctly identified (ranked adjacent in the output in the order of their appearance in the document). In this work, we modelled the user’s goal in this task as retrieving the relevant information in each article cluster of results encoded into a single sub-tree. We checked the consistency by seeing whether \( SR \) would properly evaluate the predicted improvement in the output due to different systematic perturbations. The perturbations were as follows: (i) ranked higher the relevant results, (ii) ranked lower the relevant results, (iii) removed overlap, (iv) ranked higher the leaf nodes, and (v) ranked higher the mid-branch nodes. The INEX 2007 Relevant in Context Task was evaluated for 10 systems, across 104 topics. The expected ranking of perturbed runs, for both tasks, was \( S(i) \geq S(iii) \geq S(0) \geq S(iv) \geq S(v) \geq S(ii) \), where \( S(0) \) is the SRP score of the original system output, and \( S(\cdot) \) corresponds to the SRP score of a perturbed run. Our results are reported in Section 4.4.3.

For the Thorough and Focused Tasks, we used single node assessments from the INEX 2006 element assessments. For the Relevant in Context Task, where we had the opportunity to consider more complex induced subtrees (as opposed to singletons only), we used the INEX 2007 element assessments with the relevance value function proposed in Equation 4.2.

To compare system rankings, we used Spearman’s Rho (see Section 1.3 for more details). We report the \( p \)-value which is the probability that two rankings are not correlated. A low \( p \)-value (\( p_{val} < 0.1 \)) suggests correlation. In the Focused Task, to test for the accuracy of \( SRP \), the Pearson correlation coefficient was also used (which is the
A. Accuracy - Thorough Task

<table>
<thead>
<tr>
<th>Output Size</th>
<th>k=10</th>
<th>k=100</th>
<th>k=500</th>
<th>k=1000</th>
</tr>
</thead>
<tbody>
<tr>
<td>Overall</td>
<td>0.069</td>
<td>0.070</td>
<td>0.079</td>
<td>0.129</td>
</tr>
</tbody>
</table>

B. Accuracy - Focused Task

<table>
<thead>
<tr>
<th>Output Size</th>
<th>k=5</th>
<th>k=10</th>
<th>k=25</th>
<th>k=50</th>
</tr>
</thead>
<tbody>
<tr>
<td>Overall</td>
<td>0.405</td>
<td>0.207</td>
<td>0.545</td>
<td>0.953</td>
</tr>
<tr>
<td>Relative</td>
<td>0</td>
<td>0.005</td>
<td>0.028</td>
<td>0.0002</td>
</tr>
</tbody>
</table>

C. Fidelity - Relevant-in-Context (RiC) Task

<table>
<thead>
<tr>
<th>Output Size</th>
<th>k=10</th>
<th>k=20</th>
<th>k=50</th>
<th>k=100</th>
</tr>
</thead>
<tbody>
<tr>
<td>Overall</td>
<td>8×10^{-33}</td>
<td>8×10^{-33}</td>
<td>9×10^{-33}</td>
<td>4.35×10^{-30}</td>
</tr>
</tbody>
</table>

Table 4.6: Evaluation (p-values) of SRP across tasks

general case of Spearman’s Rho) to measure the relative change in the rank of systems across rank cut-offs.

4.4.2 Accuracy

Table 4.6(A) shows the results for the Thorough Task. The Thorough Task was evaluated at rank cut-offs of $k = 10, 100, 500, 1000$, and the Focused Task at rank cut-offs of $k = 5, 10, 25, 50$. These cut-offs corresponded to those used for the official results for the respective tasks at INEX 2006. In the Thorough Task, there was strong correlation ($p_{val} < 0.1$) between system rankings using $NXCG$ and $SRP$ for low cut-offs ($k = 10, 100, 500$).

Table 4.6(B) shows the results for the Focused Task (overall) and the relative change of systems across rank-cutoffs in the Focused Task (relative). In the Focused Task (overall), $SRP$ was not correlated ($p_{val} >> 0.1$) to $NXCG$ and we might conclude that $SRP$ is not
an accurate measure.

We see that \textit{NXCG} and \textit{SRP} are different. \textit{NXCG} measures the effect of overlap, whereas \textit{SRP} measures the effect of redundancy. Overlap and redundancy are related in that overlap is a specific case of redundancy. In the Thorough Task, overlapping results were allowed in the runs, whereas in the Focused Task, they were not. This resulted in a discrepancy between the two measures in the Focused Task. Therefore, in the Focused Task, we also looked at the relative change in rank for each system across rank cut-offs to see whether \textit{SRP} changed its system rankings in the same way (although not necessarily to the same degree) as \textit{NXCG}. We observed that the two measures had strong correlation ($p_{\text{val}} < 0.03$) between the relative change of the rank of each system across the rank cut-offs, see Table 4.6(B). This showed that \textit{SRP} and \textit{NXCG} were in agreement in how they measure element retrieval systems. As noted above, the two measures did not completely agree because \textit{SRP} measures redundancy, whereas \textit{NXCG} measures only overlap. In the Focused Task, nodes from the same document could be returned, as long as they were not on the same branch in the document. Hence, overlap was not present, but the user could still navigate (and access) the rest of the document. This case of redundancy (accounted for in \textit{SRP} and thus \textit{SR}) was not considered in \textit{NXCG} (and thus XCG).

4.4.3 Fidelity

For the Thorough Task, Figure 4.4 shows that there was a strong correlation ($p_{\text{val}} < 0.05$) in the expected ranking of perturbed runs across topics. This means that given one of our perturbations of a run, we were able to predict the change in \textit{SRP} relative to the \textit{SRP} score of the original run. We conclude that, in this case, \textit{SRP} appropriately evaluated the performance of the perturbed system outputs.

We evaluated clusters as subtrees for the Relevant in Context Task for outputs with up to $k = 100$ clusters. Table 4.6(C) shows that the resultant p-values were very low ($p_{\text{val}} < 1 \times 10^{-29}$) overall across topics and runs. Thus, \textit{SRP} displayed excellent fidelity
for measuring the Relevant in Context Task as subtrees.

These results are preliminary and will need to be validated in the future with submissions produced and assessments gathered explicitly for a tree retrieval task. The results presented here showed that the fidelity of SRP was consistent for both element retrieval (results for systems that output nodes in the Thorough Task) and tree retrieval (results for systems that output subtrees in the Relevant in Context Task). By modeling SDR tasks as tree retrieval, we believe that our proposed measure will result in a consistent evaluation across SDR retrieval tasks, where a tree-based model of retrieval is applicable.

4.5 Summary

In this chapter, we presented SR, a measure based on our pillars for evaluating tree retrieval. We then explored using summaries to obtain economic, approximate models of navigation. Finally, we applied SR to the evaluation of specific tasks in element retrieval (Thorough, Focused and Relevant in Context tasks). We validated the accuracy and fidelity of SR in precision (SRP) across these tasks by comparing it to the official measure NXCG used at INEX 2006. Thus, we showed that SR was stable for evaluating these tasks. However, we also identified a fundamental difference; while NXCG accounts for just overlap in the results, SR accounts for redundancy (of which overlap is one special
The main weakness of comparing $SR$ to $NXCG$ is that our tests here may not capture potential stability problems in $SR$ where redundancy does not stem from overlap. In the next chapter, we use our pillars to define the Extended Structural Relevance framework and formulate measures, based on our pillars, that go beyond precision.
Chapter 5

Extended Structural Relevance

In this chapter, we build on our pillars of SDR evaluation presented in Chapter 3 to propose the Extended Structural Relevance (ESR) framework as a way to formulate SDR measures for XCG, PRUM and HiXEval that are consistent. In Section 2.3 we showed how the SDR measures XCG, PRUM and HiXEval were inconsistent because they do not use comparable calculations of relevance, navigation, and redundancy. In SDR evaluation, it is a known challenge to overcome the limitations in each measure that we must consider when comparing scores from XCG, PRUM and HiXEval. However, this issue is largely solved if we define our measures using a single framework like ESR. This chapter is based on our work originally published in Ali, Consens & Lalmas [5].

The outline of this chapter is as follows. In Section 5.1 we present our proposal for the Extended Structural Relevance (ESR) framework. Next, in Section 5.2 we use ESR to formulate measures for SR (Section 5.2.1), HiXEval (Section 5.2.2), XCG (Section 5.2.3), and PRUM (Section 5.2.4). Then, in Section 5.3 we illustrate our proposed ESR measures in an example. Finally, we present additional ESR measures in Section 5.4 and conclude our work in this chapter in Section 5.5.
5.1 Extended Structural Relevance Framework

Our proposed Extended Structural Relevance (ESR) framework provides the means to calculate the user’s expected gain in relevance value given redundancy. ESR is motivated by the collection partitioning scheme presented in Bollman [20] (which, in turn, is largely motivated by the much earlier work in Robertson [101]) which shows that a family of document retrieval evaluation measures (such as precision, recall, and fallout) can be derived from the parameters defined by the number of hits and misses in the output and the collection. For SDR measures, the ESR framework represents a similar family of parameters for tree retrieval based on calculating gain from hits, misses and near-misses in the output and the collection.

Let us assume that all trees in the assessments $A$ are relevant, i.e. $\text{rel}(a) > 0$ for all trees in $A$. For a given information need, consider that, of the $n$ relevant sub-trees (i.e. cardinality, or number of sub-trees in the assessments $A$), $n_C$ sub-trees are hits in the output $R$ and $n_C$ sub-trees are not in the output. If we consider the information seeking process in Section 3.2 as a single trial, then by considering repeated trials, we can define the random variable,

$$Y_C = X_C^{(1)} + X_C^{(2)} + X_C^{(3)} + \cdots + X_C^{(n_C)}$$  \hspace{1cm} (5.1)

where $X_C^{(i)}$ are independent trials, each with an associated subtree $a_C^{(i)} \in A$ for which the user seeks. Then, let us consider a function of $Y$ where we sum the random variable $\text{Hit}$ in Equation 3.3 in Section 3.2 as in

$$\text{HITS}(Y_C) = \text{Hit}(X_C^{(1)}) + \text{Hit}(X_C^{(2)}) + \text{Hit}(X_C^{(3)}) + \cdots + \text{Hit}(X_C^{(n_C)}).$$ \hspace{1cm} (5.2)

Each of our hits, i.e. $\text{Hit}(X_C^{(i)})$, has a non-zero contribution over the domain defined by the intersection of the output $R$ and the assessments $A$,

$$D_{\text{HITS}} = R \cap A.$$ \hspace{1cm} (5.3)
The expected gain in relevance value for a single trial \( \text{Hit}(X_C) \), from Equation 3.3, is

\[
E[\text{Hit}(X_C)] = \text{rel}(a) \times (1 - p(a; R_{m-1}))
\]

where \( a \) is the sub-tree that the user seeks, \( \text{rel}(a) \) is the relevance value of \( a \), \( m \) is the rank position of \( a \) in \( R \), and \( R_{m-1} \) is the sub-list of \( R \) up to rank \( m - 1 \). The expected gain in relevance value for repeated trials is

\[
E[\text{Hits}(Y_C)] = E[\text{Hit}(X_C^{(1)})] + E[\text{Hit}(X_C^{(2)})] + \cdots + E[\text{Hit}(X_C^{(n_C)})]
\]

\[
= \sum_{i=1}^{n_C} E[\text{Hit}(X_C^{(i)})] \quad \{ \text{replace additions with a summation} \}
\]

\[
= \sum_{i=1}^{n} \text{rel}(a_i) \times (1 - p(a_i; R_{m-1})) \quad \{ \text{substitute } E[\text{Hit}(X_C)] \text{ per Eq. 3.8} \}
\]

\[
= \sum_{a_i \in D_{\text{Hits}}} \text{rel}(a_i) \times (1 - p(a_i; R_{m-1})) \quad \{ \text{limit summation to trees in } D_{\text{Hits}} \}
\]

Now, let us define an analogous domain for misses, as in

\[
D_{\text{Misses}} = A \setminus R.
\] (5.4)

If we consider the information seeking process in Section 3.2 as a single trial, then by considering repeated trials, we can define the random variable,

\[
Y_C = X_C^{(1)} + X_C^{(2)} + X_C^{(3)} + \cdots + X_C^{(n_C)}
\] (5.5)

where \( X_C^{(i)} \) are independent trials, each with an associated subtree \( a_C^{(i)} \in A \) for which the user seeks. Now, we define analogous functions on \( Y_C \) for misses and near-misses, namely

\[
\text{Misses}(Y_C) = \text{Miss}(X_C^{(1)}) + \cdots + \text{Miss}(X_C^{(n_C)}), \quad \text{and}
\]

\[
\text{NearMisses}(Y_C) = \text{Near}(X_C^{(1)}) + \cdots + \text{Near}(X_C^{(n_C)}),
\] (5.7)

where \( \text{NearMisses} \) is the relevance value of near-misses.

The domain of hits \( D_{\text{Hits}} = R \cap A \), and the domain of misses \( D_{\text{Misses}} = A \setminus R \), taken together define a partition,\(^1\) since \( (A \setminus R) \cap (R \cap A) = \emptyset \) and \( (A \setminus R) \cup (R \cap A) = A \).

---

\(^1\)Equations 5.3 and 5.4 are defined using perfect matches between the trees in the assessments \( A \) and the output \( R \). A more general, albeit more complex, approach is to define hits as the trees seen with certainty \( p(a; R) = 1 \), and misses as the trees not seen with certainty \( p(a; R) < 1 \). This is a practical issue and does not constitute any loss in generality.
We can then define expected values for repeated trials of misses and near-misses in a manner analogous to hits, by substituting their respective expected values for single trials, that is, by using

\[ E[\text{Miss}(X_C)] = \text{rel}(a) \times (1 - p(a; R)) \]

for a miss, and by using

\[ E[\text{Near}(X_C)] = \text{rel}(a) \times p(a; R) \]

for a near-miss. The amount of calculation is reduced by limiting our summation to contributing terms, i.e. over the trees in the domain \( D_{\text{Misses}} \).

We summarize our expected values for gain from hits, misses and near-misses as follows. Here we consider these functions of random variables as random variables themselves, and therefore we elide the implicit \( Y_C \) and \( Y_C^\perp \) random variables from the function parameter lists. Then we have

\[
E[\text{Hits}; R, A] = \sum_{a \in D_{\text{Hits}}} \text{rel}(a) \times (1 - p(a; R_{m-1})), \quad (5.8)
\]

\[
E[\text{Misses}; R, A] = \sum_{a \in D_{\text{Misses}}} \text{rel}(a) \times (1 - p(a; R)), \quad \text{and} \quad (5.9)
\]

\[
E[\text{NearMisses}; R, A] = \sum_{a \in D_{\text{Misses}}} \text{rel}(a) \times p(a; R) \quad (5.10)
\]

where \( R = t_1, t_2, \ldots, t_k \) is a ranked list of \( k \) trees, \( R_i \) is a sublist of \( R \) up to rank \( i \) (such that if \( i \geq k \) then \( R_i = R \), and if \( i \leq 0 \) then \( R_i = \emptyset \)), \( A = \{a_1, a_2, \ldots, a_n\} \) is a set of \( n \) assessments, \( m \) is the rank of \( a \) in \( R \), \( \text{rel}(a) \) is the relevance value of tree \( a \), and \( p(a; R) \) is given in Equation 3.18 and is the probability that the user will see \( a \) once by consulting \( R \), the ranked list.\(^2\)

\(^2\)We do not need to consider gain from shrinkage, as explained above in Section 3.2. The expected value from shrinkage is \( E[\text{Shrinkage}; R, A] = \sum_{a \in D_{\text{Hits}}} \text{rel}(a) \times p(a; R_{m-1}) \).
We define the \textit{recall-base} as the upper limit of gain from hits, misses and near-misses from a series of independent trials. This function, which is also a random variable, is therefore (eliding the implicit $Y = Y_C + Y_{\bar{C}}$ as in Eqs. 5.8 to 5.10)

\[ \text{RecallBase} = \text{Hits} + \text{Misses} + \text{NearMisses}. \]  

(5.11)

We have noted above that our domains $D_{\text{Hits}}$ and $D_{\text{Misses}}$ define a partition. The expected value of gain for the recall-base is then


(5.12)

In the experimental validation for our work in Chapter 3 (shown in Chapter 4 and originally published in Ali, Consens, Kazai & Lalmas [7]), we evaluated system performance based on inferred relevance value (Equation 4.2). This limits evaluation of tree retrieval effectiveness to precision, because gain is limited to the user seeing relevant nodes in the output, i.e. hits. ESR goes beyond this, by calculating the user’s expected (realizable) gain in relevance value from hits, misses and near-misses, and combining these to define recall. A summary of the statistical model presented in this chapter is found in Table 5.1.

We complete this section with an example calculation of our ESR expected gain from hits, misses and near-misses, based on our example in Section 3.5 where we define a toy collection in Figure 3.2(a); a set of relevance judgments in Table 3.5 for both binary relevance and relevance by length for nodes $e_3$ and $e_4$; a toy navigation model in Table 3.6; and, three example outputs System 1 ($R_1$), System 2 ($R_2$), and System 3 ($R_3$) in Table 3.7.

Let us begin by calculating the ESR parameters for example System 1 ($R_1$). We recall that ESR relies on three main parameters expressing, respectively, how the user gains relevant information either by the system retrieving the information directly (hits with $E[\text{Hits}; R, A]$ in Equation 5.8), by the user locating the relevant information via
<table>
<thead>
<tr>
<th>Notation</th>
<th>Type, Value</th>
<th>Meaning</th>
</tr>
</thead>
<tbody>
<tr>
<td>$A$</td>
<td>set of sub-trees</td>
<td>Sub-trees assessed for relevance. See Def. 3.2.1</td>
</tr>
<tr>
<td>$R$</td>
<td>ranked list of sub-trees</td>
<td>Output of a system, browsing history. See Def. 3.2.4</td>
</tr>
<tr>
<td>$D_{HTS}, D_{MISSES}$</td>
<td></td>
<td>Domains for hits and misses. Defines a partition of $A$.</td>
</tr>
<tr>
<td>$X^{(i)}<em>C, X^{(i)}</em>{\bar{C}}$</td>
<td>random variables, $X^{(i)}<em>C = x^{(i)}<em>C \in \Omega_C$, $X^{(i)}</em>{\bar{C}} = x^{(i)}</em>{\bar{C}} \in \Omega_{\bar{C}}$</td>
<td>Random outcomes of the navigating to see fixed sub-tree $a^{(i)}<em>C \in D</em>{HTS}$ (and $a^{(i)}<em>{\bar{C}} \in D</em>{MISSES}$, resp.), dependent on the user’s information seeking process. See Def. 3.2.8</td>
</tr>
<tr>
<td>$n$</td>
<td>$n_C + n_{\bar{C}}$, $n_C =</td>
<td>D_{HTS}</td>
</tr>
<tr>
<td>$Y_C$</td>
<td>$X^{(1)}_C + \cdots + X^{(n_C)}_C$</td>
<td>random variable of repeated trials of outcomes $X^{(i)}_C$. See Eq. 5.1</td>
</tr>
<tr>
<td>$Y_{\bar{C}}$</td>
<td>$X^{(1)}<em>{\bar{C}} + \cdots + X^{(n</em>{\bar{C}})}_{\bar{C}}$</td>
<td>random variable of repeated trials of outcomes $X^{(i)}_{\bar{C}}$. See Eq. 5.5</td>
</tr>
<tr>
<td>Hits($Y_C$)</td>
<td>$\text{Hit}(X^{(1)}_C) + \cdots + \text{Hit}(X^{(n_C)}_C)$</td>
<td>Gain in relevance value from hits. See Eq. 5.2</td>
</tr>
<tr>
<td>Misses($Y_C$)</td>
<td>$\text{Miss}(X^{(1)}<em>C) + \cdots + \text{Miss}(X^{(n</em>{\bar{C}})}_{\bar{C}})$</td>
<td>Unrealized gain in relevance value from misses. See Eq. 5.6</td>
</tr>
<tr>
<td>NearMisses($Y_{\bar{C}}$)</td>
<td>$\text{Near}(X^{(1)}<em>{\bar{C}}) + \cdots + \text{Near}(X^{(n</em>{\bar{C}})}_{\bar{C}})$</td>
<td>Gain in relevance value from near-misses. See Eq. 5.7</td>
</tr>
<tr>
<td>$p(a; R)$</td>
<td>$1 - \prod_{j=1}^{k} (1 - \tilde{p}(a; t_j))$</td>
<td>The probability that a user navigates to sub-tree $a$ from one or more entries in $R$. See Eq. 3.18</td>
</tr>
<tr>
<td>$E[\text{Hits}; R, A]$</td>
<td>$\sum_{a \in D_{HTS}} \text{rel}(a) \times (1 - p(a; R_{m-1}))$</td>
<td>Expected gain from hits. See Eq. 5.8</td>
</tr>
<tr>
<td>$E[\text{Misses}; R, A]$</td>
<td>$\sum_{a \in D_{MISSES}} \text{rel}(a) \times (1 - p(a; R))$</td>
<td>Expected unrealized gain from misses. See Eq. 5.9</td>
</tr>
<tr>
<td>$E[\text{NearMisses}; R, A]$</td>
<td>$\sum_{a \in D_{MISSES}} \text{rel}(a) \times p(a; R)$</td>
<td>Expected gain from near-misses. See Eq. 5.10</td>
</tr>
</tbody>
</table>

Table 5.1: Summary of statistical model for ESR evaluation.
navigation (near-misses with $E[\text{NearMisses}; R, A]$ in Equation 5.10), or, not at all (misses with $E[\text{Misses}; R, A]$ in Equation 5.9).

We determine the ESR parameters of expected gain in relevance value from hits, misses and near-misses at rank cut-off $k = 1$ for System 1 in Table 3.7 (recall that $R_1 = e_1, e_3, e_4$). At rank 1, System 1 outputs element $e_1$. This is not a hit because $R_1 \cap A = \emptyset$ (Equation 5.3). The misses are $A \setminus R_1 = e_3, e_4$ (Equation 5.4). The probability that relevant element $e_3$ can be navigated to from retrieved element $e_1$ is $p(e_3; R_1) = 1 - (1 - \tilde{p}(e_3; e_1)) = \tilde{p}(e_3; e_1) = 0.16$ (Equation 3.18). Similarly, the probability that $e_4$ can be navigated to is $p(e_4; R_1) = 1 - (1 - \tilde{p}(e_4; e_1)) = 0.11$. Assume binary relevance.

The expected gain from the near-miss $e_3$ is $rel(e_3) \times p(e_3; R_1) = 0.16$ (Equation 5.10). The expected (unrealized) gain from the miss $e_3$ is $rel(e_3) \times (1 - p(e_3; R_1)) = 0.84$ (Equation 5.9). The expected gain from the near-miss $e_4$ is $rel(e_4) \times p(e_4; R_1) = 0.11$ (Equation 5.10) and miss $e_4$ is $rel(e_4) \times (1 - p(e_4; R_1)) = 0.89$ (Equation 5.9). The recall-base is $0.16 + 0.84 + 0.11 + 0.89 = 2$ (Equation 5.12). The expected gains, defined by binary relevance, are shown in Row 1 of Table 5.2. The values in Row 1 in parentheses show the expected gains if we define the relevance value as relevance by length.

The expected gains in relevance value from hits, misses, and near-misses across rank cut-offs for Systems 1, 2 and 3 are shown for binary relevance and relevance by length (in parentheses) in Table 5.2. These can be used to understand whether these systems retrieve information directly or whether the user must navigate to find relevant information. For instance, at $k = 3$ (Rows 7–9), we note that both Systems 1 and 3 retrieve relevant information, whereas in System 2 the system returns near-misses and hence the user will need to spend some effort to locate relevant information.

The expected value of the recall-base across rank cut-offs is shown in Table 5.3. At a given rank cut-off, the size of the recall-base changes inversely to the redundancy in the output up to the given rank, i.e., more redundancy in the output will reduce the size of the recall-base (and the expected gains from hits and near-misses). In our example,
we note that by using System 3 the user will experience the least redundancy (Row 3 in Table 5.3) with the greatest gain (from Row 9 in Table 5.2). Additionally, we note that users of System 2 experience the least overall gain (from Row 8 in Table 5.2). This corresponds to our earlier assertion that System 2 would have the worst performance and System 3 would have the best.

To summarize, the ESR framework is comprised of three related expected values; namely expected gain in relevance value from the user seeing hits in the output (Equation 5.8), expected unrealized gain in relevance value from (unseen) misses in the collection (Equation 5.9), and expected gain in relevance value from the user seeing near-misses in the collection (Equation 5.10). The sum of these three expectations (Equation 5.12) represents our recall-base in ESR. Next, we show how this framework is used to measure performance for several task-specific approaches in SDR.

5.2 ESR Evaluation Measures

In this section, we formulate SDR evaluation measures for SR, HiXEval, XCG, and PRUM (introduced in Chapter 2) within our ESR framework. The selected measures are expressed in terms of the expectations defined in the previous section. We start by describing the original measures. Note that each represents a family of measures, and we formulate only a selection in each.

We recall the notation used in previous sections. The output $R = t_1, t_2, \ldots, t_k$ is a ranked list of $k$ trees. $R_i$ is a sublist of $R$ up to rank $i$ such that if $i > k$ then $R_i = R$ and if $i \leq 0$ then $R_0 = \emptyset$. The assessments $A = a_1, a_2, a_3, \ldots, a_n$ is a set of the trees that have been judged for a given information need. A relevant tree $a \in A$ has a relevance value $rel(a) > 0$, and $rel(a) = 0$ otherwise. Our expected gains in relevance value in ESR are $E[\text{Hits}; R, A]$ in Equation 5.8, $E[\text{Misses}; R, A]$ in Equation 5.9, $E[\text{NearMisses}; R, A]$ in Equation 5.10, and $E[\text{RecallBase}; R, A]$ in Equation 5.12.
### Chapter 5. Extended Structural Relevance

#### Table 5.2: Expected gain in relevance value with binary relevance (with relevance by length) for hits, near-misses, and misses.

<table>
<thead>
<tr>
<th></th>
<th>$E[\text{Hits}]$</th>
<th>$E[\text{NearMisses}]$</th>
<th>$E[\text{Misses}]$</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>$e_3$</td>
<td>$e_4$</td>
<td>$e_3$</td>
</tr>
<tr>
<td>Sys. 1</td>
<td>0.16 (4.8)</td>
<td>0.11 (2.2)</td>
<td>0.84 (25.2)</td>
</tr>
<tr>
<td>Sys. 2</td>
<td>0.16 (4.8)</td>
<td>0.11 (2.2)</td>
<td>0.84 (25.2)</td>
</tr>
<tr>
<td>Sys. 3</td>
<td>1 (30)</td>
<td>0 (0)</td>
<td>–</td>
</tr>
</tbody>
</table>

Output size $k = 2$

<table>
<thead>
<tr>
<th></th>
<th>$E[\text{Hits}]$</th>
<th>$E[\text{NearMisses}]$</th>
<th>$E[\text{Misses}]$</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>$e_3$</td>
<td>$e_4$</td>
<td>$e_3$</td>
</tr>
<tr>
<td>Sys. 1</td>
<td>0.84 (25.2)</td>
<td>–</td>
<td>0.11 (2.2)</td>
</tr>
<tr>
<td>Sys. 2</td>
<td>–</td>
<td>0.16 (4.8)</td>
<td>0.23 (4.56)</td>
</tr>
<tr>
<td>Sys. 3</td>
<td>1 (30)</td>
<td>–</td>
<td>0.11 (2.2)</td>
</tr>
</tbody>
</table>

Output size $k = 3$

<table>
<thead>
<tr>
<th></th>
<th>$E[\text{Hits}]$</th>
<th>$E[\text{NearMisses}]$</th>
<th>$E[\text{Misses}]$</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>$e_3$</td>
<td>$e_4$</td>
<td>$e_3$</td>
</tr>
<tr>
<td>Sys. 1</td>
<td>0.84 (25.2)</td>
<td>0.89 (17.8)</td>
<td>–</td>
</tr>
<tr>
<td>Sys. 2</td>
<td>–</td>
<td>0.16 (4.8)</td>
<td>0.23 (4.56)</td>
</tr>
<tr>
<td>Sys. 3</td>
<td>1 (30)</td>
<td>0.89 (17.8)</td>
<td>–</td>
</tr>
</tbody>
</table>

#### Table 5.3: Recall-base using binary relevance (relevance by length).

<table>
<thead>
<tr>
<th></th>
<th>$E[\text{RecallBase}]$</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>$k = 1$</td>
</tr>
<tr>
<td>Sys. 1</td>
<td>2 (50)</td>
</tr>
<tr>
<td>Sys. 2</td>
<td>2 (50)</td>
</tr>
<tr>
<td>Sys. 3</td>
<td>2 (50)</td>
</tr>
</tbody>
</table>
5.2.1 Structural Relevance

Structural relevance (SR) in Chapter 4 (originally published in Ali, Consens, Kazai & Lalmas [7]) is a measure of the user’s expected gain in relevant information given that the information may be redundant. The calculation of SR depends on the user’s browsing history, which is the set of trees that a user has viewed previously. We calculate SR as the expected relevance value of a ranked list given that the user does not find the results redundant,

\[
SR(R) = \sum_{i=1}^{k} rel_{\text{tree}}(t_i) \times (1 - p(t_i; R_{i-1})),
\]

(5.13)

where the system output \( R = t_1, t_2, \ldots, t_k \) is a ranked list of \( k \) subtrees, the browsing history \( R_{i-1} \) is the set of subtrees that are ranked higher than the \( i \)-th subtree \( t_i \in R \), \( rel_{\text{tree}}(t_i) \in [0, 1] \) is the inferred relevance value of subtree \( t_i \) (Equation 4.2), and the redundancy \( p(t_i; R_{i-1}) \), as given in Equation 3.18, is the probability that the nodes in tree \( t_i \) have been seen one or more times given the browsing history \( R_{i-1} \).

SR is calculated by summing the expected gain in inferred relevance value for the trees in the output. SR in Precision (SRP), which is \( SRP = SR(R)/k \), is used to measure precision of tree retrieval systems, and mean average precision across rank cut-offs is used to rank systems.

To represent SRP in ESR, we replace the expected gain in inferred relevance value \( SR(R) \) with the expected gain from hits in ESR. That is, we substitute \( SR(R) \) from Equation 5.13 with \( E[\text{Hits}; R, A] \) from Equation 5.8 giving

\[
ESRP(R, A) = E[\text{Hits}; R, A]/k.
\]

(5.14)

Note that if the inferred relevance value is equal to the assessed relevance value then \( SR(R) \) and \( E[\text{Hits}; R, A] \) are equivalent. For instance, in element retrieval, systems retrieve singletons and the inferred relevance in SR is exactly the judged relevance value in ESR, thus explaining the above equivalence.

A key limitation of inferred relevance value \( rel_{\text{tree}}(t_i) \), as originally proposed in SR,
is that it is not possible to calculate recall. The denominator in a recall calculation, i.e. the recall-base, is traditionally derived by summing the hits and misses. However, our expected gain in inferred relevance value $rel_{tree}(t_i)$ only accounts for gain from hits (and hits that are navigated to from higher-ranked trees, i.e. shrinkage in Section 3.2). Misses (corresponding unrealized gain) cannot be accounted for, thus recall cannot be defined. By formulating SR in our framework, as shown next, this limitation is overcome.

Indeed, we recall that the user gains relevant information from hits (Equations 5.8) and near-misses (Equations 5.10). The recall-base in Equation 5.12 represents an upper-limit to the user’s possible gain. We obtain a measure of recall by dividing the sum of the gain from hits and near-misses by the recall-base. We refer to this measure as Structural Relevance in Recall (ESRR), given by

$$ESRR(R, A) = \frac{E[\text{Hits}; R, A] + E[\text{NearMisses}; R, A]}{E[\text{RecallBase}; R, A]}.$$ (5.15)

Classical precision for document retrieval is $n/k$ (the number of hits $n$ divided by the length of the output $k$). Classical recall for document retrieval is $n/(n+m)$ (the number of hits divided by the size of the recall-base which is the number of hits $n$ plus the number of misses $m$). ESRP and ESRR reduce to classical precision and recall, respectively. First, we assume binary relevance such that $rel(a) = 1$ for all hits and misses. Second, we assume that there is no navigation $\bar{p}(e; e) = 1$ (Equation 3.12) and therefore redundancy $p(a; R_{m-1}) = 0$ for hits (Equation 3.8), and $p(a; R) = 0$ for near-misses (Equation 3.10) and misses (Equation 3.9), respectively. Let $n_H$ denote the expected gain from hits in Equation 5.8 which reduces to the number of relevant, retrieved sub-trees. Let $n_M$ denote the expected gain (unrealized) from misses in Equation 5.9 which reduces to the number of relevant, not retrieved sub-trees. We note that our expected gain from near-misses in Equation 5.10 is zero. Our measures reduce to $ESRP(R, A) = n_H/k$ and $ESRR(R, A) = n_H/(n_H+n_M)$, which are classical precision and recall, respectively, albeit where the retrieval units are sub-trees instead of documents.
5.2.2 Highlighting XML Retrieval Evaluation

Highlighting XML evaluation (HiXEval) proposed in Pehcevski & Thom [89], and further finalized in Kamps et al. [56], was developed to evaluate the performance of systems that retrieve (or can be modelled as retrieving) passages, where a passage is a block of text, delineated or not with XML tags.

HiXEval exploits the relevance assessment methodology used at INEX since 2005 (see Piwowarski, Trotman & Lalmas [96]), where human judges highlight the relevant passages in retrieved (pooled) documents. With this methodology, for a given information need, the relevant parts in documents are those that have been highlighted by the human judges (and described in Piwowarski, Trotman & Lalmas [96]). HiXEval measures precision and recall based on the amount of relevant information retrieved; the amount of relevant information in the collection; and the overlap of the relevant text in retrieved passages. The “amount of information” is measured using the character length of passages.

In HiXEval, for a given information need the total relevance value of the information contained across all documents in the collection is given by the number of highlighted characters in the whole collection. Let $T_{rel}$ denote the number of characters in the relevant (highlighted) text in the collection. The relevance value of a retrieved passage in HiXEval is the character length of the relevant text in the passage. If the relevant text overlaps with another retrieved passage, then the overlapped text is relevant to the user with probability $\alpha \in [0, 1]$, where $\alpha$ refers to the user tolerance to overlap as defined by de Vries, Kazai & Lalmas [36]. HiXEval assumes that user navigation does not extend beyond the boundaries of retrieved passages. Thus, HiXEval considers redundancy as only occurring between adjacent retrieved text passages overlapping each other.

For a retrieved passage $e$, the user gain in relevant information is given by $rsize(e)$, which is defined as follow. Let $rel(e)$ denote the number of characters in the relevant text in the passage. Let $rov(e)$ denote the number of characters in the relevant text that is overlapped with a higher-ranked passage in the output s.t. $rov(e) \leq rel(e)$. The gain
for retrieved passage $e$ is stated as follows:

$$rsize(e) = rel(e) - (1 - \alpha) \times rov(e)$$  \hspace{1cm} (5.16)$$

where $\alpha$ refers to the user tolerance to overlap defined in de Vries, Kazai & Lalmas [36].

Based on the above, numerous measures can be obtained for measuring precision and recall in passage retrieval. In this section, we consider two HiXEval measures; namely interpolated precision (iP) and interpolated recall (iR).\(^3\) Interpolated precision is the user gain in relevant information divided by the number of characters retrieved (Equation 5.17). Interpolated recall is the user gain in relevant information divided by the total relevance value in the collection (Equation 5.18).

$$iP@r = \frac{\sum_{i=1}^{r} rsize(e_i)}{\sum_{i=1}^{r} size(e_i)}$$ \hspace{1cm} (5.17)$$

$$iR@r = \frac{\sum_{i=1}^{r} rsize(e_i)}{T_{rel}}$$ \hspace{1cm} (5.18)$$

where $R = e_1, e_2, \ldots, e_k$ is a ranked list of $k$ passages, $e_i$ is a passage, $size(e)$ is the number of characters in passage $e$, $rsize(e)$ is shown in Equation 5.16, $T_{rel}$ is the number of characters in the relevant text in the collection, and $r \in [1, k]$ is a rank. Systems are ranked using mean average precision across either rank cut-offs or recall points.

We formulate now HiXEval, i.e. iP and iR, in the ESR framework. First, we define the relevance value $rel(a)$ as the number of characters in the relevant text in the nodes of the tree $a$. Second, we note that overlap in tree retrieval is a specific case of redundancy where trees in the output share nodes in common, and that this can be accounted for in ESR using an appropriate user navigation model. Third, let $T_{rel}$ be the number of characters in the relevant text in the collection and $size(t)$ denote the number of characters in the nodes of tree $t$.

\(^3\)At INEX, as defined in Kamps et al. [56], generalized precision (gP) and generalized recall (gR) are currently the official (HiXEval-based) measures for INEX. Interpolated precision (iP) and interpolated recall (iR) (also defined in Kamps et al. [56]) have been used in the past as official measures at INEX. In Section 6.3, we validate ESR measures of iP, iR, and gP.
We replace the user gain in relevant information $rsize()$ (Equation 5.16) with the sum of the gain from hits (Equation 5.8). We limit gain to hits because HiXEval measures limit consideration of user navigation to within retrieved elements. This is fully accounted for in ESR with hits. Indeed, as stated in Section 3.2, near-misses in HiXEval (and XCG) are defined differently than in ESR.\footnote{Near-misses are defined in XCG by Lalmas & Kazai\cite{59} as retrieved sub-documents that, may or may not be relevant, but which can be navigated from by the user to see non-retrieved, relevant information. Whereas, in ESR, we reverse this definition and define near-misses as relevant sub-documents that are navigated to from a retrieved sub-document.} We obtain the following ESR measures for interpolated precision ($SRiP$) and recall ($SRiR$), stated without derivation,

$$SRiP(R, A) = \frac{E[\text{Hits}; R, A]}{\sum_{i=1}^{k} \text{size}(t_i)}$$

(5.19)

$$SRiR(R, A) = \frac{E[\text{Hits}; R, A]}{T_{rel}}$$

(5.20)

where $\text{size}(t_i)$ is the number of characters in tree $t_i$, $T_{rel} = \sum_{a \in A} rel(a)$ is the number of characters in the relevant text in the collection, and $rel(a)$ is the number of characters in the relevant text of tree $a \in A$.

The key differences between $iP/iR$ and $SRiP/\text{SRiR}$ are that $SRiP/\text{SRiR}$ are based on tree retrieval and consider a broader notion of redundancy than overlap $\alpha$ in Equation 5.16 (which we consider as a special case of redundancy already considered in our framework). The $\text{SRiP}/\text{SRiR}$ measures above can be applied to any search task that can be modelled using tree retrieval. This demonstrates an important advantage when using ESR in that an evaluation approach like HiXEval can be applied to tasks that go beyond the search paradigm, here passage retrieval, for which it was originally proposed.

### 5.2.3 Extended Cumulated Gain

Extended cumulated gain (XCG), defined by Kazai & Lalmas\cite{59}, is a family of measures that evaluate the user gain in relevant information from an actual system compared to
the gain possible from an ideal system (see Section 2.2 for details on ideality). One of the XCG measures is the normalized extended cumulated gain \((NXCG)\), which we formulate now within our ESR framework.

\(NXCG\) is the ratio of the user cumulated gain in relevant information from an actual system compared to the cumulated gain from an ideal system. The cumulated gain \(xCG[k]\) (Equation [5.21]) is the user gain after consulting \(k\) ranks from the actual system. The ideal cumulated gain \(xCI[k]\) (shown in Equation [5.22]) is the user gain after consulting \(k\) ranks from the ideal system. \(NXCG\) is defined as their ratio (Equation [5.23]).

\[
xCG[k] = \sum_{i=1}^{k} xG[i] \quad (5.21)
\]

\[
xCI[k] = \sum_{i=1}^{k} xI[i] \quad (5.22)
\]

\[
NXCG[k] = \frac{xCG[k]}{xCI[k]} \quad (5.23)
\]

where \(xG[i]\) is the gain from the \(i\)-th element in the actual output \(R = e_1, e_2, \ldots, e_k\), and \(xI[i]\) is the gain from the \(i\)-th element in the ideal output \(I = ideal_1, ideal_2, \ldots, ideal_n\). XCG has been developed for measuring element retrieval systems,\(^5\) and thus \(e_i\) is an XML element in the output, and \(ideal_i\) is an element in the set of assessed ideal elements. Averaged \(NXCG\) at a given rank cut-off is used to rank systems. We present below how \(xG[i]\) and \(xI[i]\) in Equations [5.22] and [5.21], respectively, can be calculated.

Relevance in XCG is considered as follows. At each rank consulted, the user gains relevant information depending on whether the consulted element contains relevant text and whether its text overlaps with other retrieved elements. There are numerous ways in XCG to calculate the user gain in relevant information, depending on how the relevance of elements has been determined (which has changed over the years at INEX – see Piwowarski, Trotman & Lalmas [96]). For illustrative purposes, we use the same approach

\(^5\) Although this does mean that XCG cannot be extended to evaluate passage retrieval.
above in HiXEval as described in Section 5.2.2 which is based on assessors highlighting relevant text to judge relevance.

Let $size(e)$ denote the number of characters in a retrieved element. Let $rsize(e)$ denote the number of characters in the text of a retrieved element that are relevant to the user. The calculation of $rsize(e)$ is shown above in Section 5.2.2 in Equation 5.16. The actual gain in Equation 5.21 is then $xG[i] = rsize(e_i)/size(e_i)$. The ideal gain in Equation 5.22 is then $xI[i] = rsize(ideal_i)/size(ideal_i)$. Our calculation of gain in XCG is based on INEX 2006 evaluation measures defined by Lalmas et al. [70], however, in general, gain in XCG is not limited to normalized values $[0, 1]$.

We now formulate NXCG using ESR. As for HiXEval, we limit gain in XCG to hits. For this, we first consider $xCG$ and $xCI$ within our ESR framework. We start with $xCG$ (Equation 5.21). The user’s total expected gain in relevance value is the sum of hits (Equation 5.8). We refer to this as the cumulated gain $CG$ and show this below in Equation 5.24.

We now discuss $xCI$ (Equation 5.22). In this work, we propose an alternative that mitigates the instability caused by ideality cited by Kazai, de Vries & Lalmas [60]. We propose that ideal cumulated gain be replaced with a desired cumulated gain which is the cumulated gain that a user expects to experience at each consulted rank. Similar approaches to measuring effort-gain relationships can be found in expected search length (see Cooper [35]) and precall (see Raghavan, Bollman & Jung [98]). We model the user’s desired cumulated gain as follows.

Let $l \in (0, 1]$ be the proportion of the recall-base in Equation 5.12 needed to satisfy the user’s the information need. We call $l$ the desired recall. Let $m$ denote the desired effort spent (by number of ranks) to satisfy the user’s information need. Now, we say that for the system to meet the desires of our user then the user must gain $l/m$ of the recall-base (Equation 5.12) at each rank consulted, $(l/m) \times E[RECALL\!BASE; R_i, A]$. We call this our desired gain per rank. Let $i$ denote a rank cut-off. At rank $i$, the cumulated
desired gain $CD[i]$ is our desired gain per rank times our rank cut-off. This is shown below in Equation 5.25.

Thus, we can now express $NXCG$ within ESR. We divide the user gain from hits $CG[i]$ by the desired cumulated gain $CD[i]$. We call this measure normalized extended cumulated gain in ESR ($NSRCG$), shown in Equation 5.26.

\[
CG[i] = E[\text{Hits}; R_i, A] \quad (5.24)
\]

\[
CD[i] = i \times l \times E[\text{RecallBase}; R_i, A]/m \quad (5.25)
\]

\[
NSRCG[i] = \frac{CG[i]}{CD[i]} \quad (5.26)
\]

where $i \in [1, k]$ is the number of ranks consulted, $l \in (0, 1]$ is the desired recall, and $m$ is the desired effort in number of ranks.

$NSRCG$ does not require an ideality assumption. Ideality means that we must have human judges identify what is ideal. Desired recall has no such implication. Analogous to precision-recall analysis described in Chapter 1.2, we analyse systems by scoring their performance with $NSRCG$ across a range of desired values to determine how well the system serves a range of users with different desires (e.g. an interpretation of precision-recall analysis is the determination of whether a system with precision score $X$ can meet the needs of users’ who desire recall $Y$ to satisfy their information need). $NSRCG$ has the added benefit that it allows $NXCG$ (and other measures in the XCG family) to be applied to any SDR search task that can be modelled as tree retrieval.

### 5.2.4 Precision-Recall with User Modeling (PRUM)

Precision-Recall with User Modelling (PRUM) defined in Piwowarski, Gallinari & Dupret is a measure of whether a user sees a desired number of ideal elements. It is the ratio of the expected number of rank positions where the user gains relevant information by
seeing ideal elements compared to the expected number of rank positions that the user consults to satisfy their information need. It is calculated in Piwowarski & Dupret [92], given an information need, desired recall, and output, as follows,

\[
PRUM = \frac{E[\# \text{ of rank positions user sees ideal}]}{E[\# \text{ of rank positions consulted}]}
= \frac{E[CL(i)]}{E[CR(i)]}
\] (5.27)

where \( i \) is the user’s desired recall in number of ideal elements, \( CL(i) \) is the number of rank positions where the user gains relevant information, \( CR(i) \) is the number of ranked positions that the user must consult to see \( i \) unique, ideal elements. Systems are ranked using \( PRUM \) at the user desired recall-level averaged across topics.

In \( PRUM \), to see elements, a user either consults the system output or navigates from a retrieved element. In ESR, the user must visit the element to see it. Indeed, this is a key difference to our model in ESR. There is no concept of “visiting the element” in \( PRUM \). Moreover, in \( PRUM \), the user seeks to see a set of ideal elements that must be identified a priori to evaluation. In ESR, there is no concept of an ideal element. To appreciate the implications of the difference between these two approaches, the interested reader is encouraged to review the full derivation of \( PRUM \) in Piwowarski & Dupret [92].

We limit our presentation to demonstrating how \( PRUM \) can be calculated. We do not further elaborate on the work in Piwowarski & Dupret [92] that justifies the selection of ideal elements nor how to determine navigational probabilities in \( PRUM \). We next demonstrate how the expectations for \( CR(i) \) and \( CL(i) \), respectively, in Equation 5.27 can be calculated.

\( PRUM \) is calculated by enumerating all of the possible scenarios of consultations and navigations that result in the user seeing the desired number of ideal elements. A scenario is a set of navigations constrained by consultations to the output that result in the user seeing at least their desired number of ideal elements. Let \( i \) denote the desired number
### Table 5.4: Example of PRUM for $P(e_1 \leadsto e_4) = 0.2$.

The probability of scenarios $A$ and $B$ are calculated as $P(A) = P(e_1 \leadsto e_4) = 0.2$ and $P(B) = 1 - P(e_1 \leadsto e_4) = 0.8$.

<table>
<thead>
<tr>
<th>Scenarios</th>
<th>Rank</th>
<th>$A$</th>
<th>$B$</th>
</tr>
</thead>
<tbody>
<tr>
<td>1 ($e_3$)</td>
<td>$e_3$</td>
<td>$e_3$</td>
<td></td>
</tr>
<tr>
<td>2 ($e_1$)</td>
<td>$e_4$</td>
<td>-</td>
<td></td>
</tr>
<tr>
<td>3 ($e_4$)</td>
<td>-</td>
<td>$e_4$</td>
<td></td>
</tr>
<tr>
<td>$P(S)$</td>
<td>0.2</td>
<td>0.8</td>
<td>$E[]$</td>
</tr>
<tr>
<td>$CL(2)$</td>
<td>2</td>
<td>2</td>
<td>2</td>
</tr>
<tr>
<td>$CR(2)$</td>
<td>2</td>
<td>3</td>
<td>2.8</td>
</tr>
</tbody>
</table>

We illustrate with an example. Consider calculating $PRUM$ for a system that outputs $R = e_3, e_1, e_4$ for the document shown in Figure 3.2(a). The calculation of $PRUM$ is as follows, and summarized in Table 5.4. Let the ideal elements be $e_3$ and $e_4$, which are outputs in rank positions 1 and 3, respectively. Let the desired recall-level be two ideal elements $i = 2$. Assume that the only possible navigation is from element $e_1$ to element $e_4$. Given this navigation model, there are two possible scenarios:

(A) the user sees $e_3$ by consulting the ranked list and sees $e_4$ by navigating from $e_1$, so the user consults the ranked list two times ($CR(2) = 2$) and there are two ranks (1 and 2) that lead to seeing unique, ideal elements ($CL(2) = 2$);

(B) the user sees $e_3$ and $e_4$ by consulting the ranked list and does not navigate to $e_4$. 

The probability $P(S)$ of each scenario $S$ occurring can be calculated based on the taken (and not taken) navigations. The expected ranks are calculated by conditioning $CR(i)$ and $CL(i)$, respectively, on the probability $P(S)$ such that $PRUM = E[CL(i)]/E[CR(i)]$. 

We illustrate with an example. Consider calculating $PRUM$ for a system that outputs $R = e_3, e_1, e_4$ for the document shown in Figure 3.2(a). The calculation of $PRUM$ is as follows, and summarized in Table 5.4. Let the ideal elements be $e_3$ and $e_4$, which are outputs in rank positions 1 and 3, respectively. Let the desired recall-level be two ideal elements $i = 2$. Assume that the only possible navigation is from element $e_1$ to element $e_4$. Given this navigation model, there are two possible scenarios:
from $e_1$, so the user consults the ranked list three times ($C_R(2) = 3$) and there are
two ranks (1 and 3) that lead to seeing unique, ideal elements ($CL(2) = 2$);

The probability of each scenario occurring is determined as follows. Let $P(f \leadsto e)$
denote user navigation as the probability\(^6\) that the user has seen element $e$ given that
the user has seen element $f$. Assume user navigation is $P(e_1 \leadsto e_4) = 0.2$. For scenario
A, the user navigates to $e_4$ from $e_1$, i.e., $P(A) = P(e_1 \leadsto e_4)$. For scenario B, the user
does not navigate to $e_4$ from $e_1$, i.e., $P(B) = 1 - P(e_1 \leadsto e_4)$. Table 5.4 summarizes the
scenarios $A$ and $B$, where the row $P(S)$ shows the probability of each scenario occurring,
the column $E[\cdot]$ shows the expected values of $CL(2)$ and $C_R(2)$, and the column $PRUM$
shows that the $PRUM$ precision for this example is 0.714.

Next, we use ESR to formulate $PRUM$. For this, we need to express $CL$ and $C_R$
within ESR. The user desire is to see non-redundant, relevant information at each rank
position consulted. Note that $PRUM$, as defined in Piwowarski, Gallinari & Dupret \[94\],
does not consider graded assessments. Thus, let us assume binary relevance values. User
navigation in ESR and $PRUM$ is similar (they differ in their probabilistic interpretation,
but, in general, both consider navigation between pairs of nodes). Thus, unlike HiXEval
and XCG, $PRUM$ considers near-misses in ESR and the user gain in $PRUM$ is the sum of
hits (Equation 5.8) and near-misses (Equation 5.10). The desired number of consultations
of the output is equal to the gain because each relevant tree contributes up to 1 to the
gain. Thus, the desired number of ranks is stated as:

$$CL(i) = E[\text{Hits}; R_i, A] + E[\text{NearMisses}; R_i, A]$$  (5.28)

where $i \in [1, k]$ is a rank cut-off, and $rel(a) = 1$ for relevant trees $a \in A$, and $rel(a) = 0$
otherwise.

\(^6\) $P(f \leadsto e)$ in $PRUM$ is different from $\tilde{p}(e; f)$ in ESR. $P(e \leadsto f)$ is the probability that a user who
has seen element $f$ has also seen element $e$. Whereas, $\tilde{p}(e; f)$ is the probability that a user sees element $e$
by navigating to element $e$ given a navigation from element $f$. The difference lies in how user navigation
is assessed in e.g. user studies. $P(f \leadsto e)$ is determined by asking the reader post-assessment whether
specific ideal elements were seen or not. In contrast, $\tilde{p}(e; f)$ is determined by tracking the reader’s
attention and assuming that navigation is independent of relevance.
The number of ranks that the user consults to satisfy a given information need is obtained by calculating the rank cut-off for a given recall level. Let \( r \) be the user desired recall level. Let \( m \) be the minimum rank cut-off where the user desired recall level is achieved. This cut-off is calculated using \( \text{ESRR}(R, A) \) (Equation 5.15) by evaluating ESRR across rank cut-offs \( m \in [1, k] \) until \( \text{ESRR}(R_m, A) \) is greater than or equal to the desired recall \( r \):

\[
C_R = \min_{m \in [1, k]} m, \text{ where } \text{ESRR}(R_m, A) \geq r
\]  

(5.29)

Precision in \( PRUM \) using ESR (\( SRPRUM \)) is the ratio between the desired number of ranks to achieve a given recall-level \( CL(C) \) and the rank cut-off \( C_R \) where a given recall-level is achieved, which is

\[
SRPRUM = \frac{CL(C_R)}{C_R}
\]  

(5.30)

where \( CL(C_R) \) (Equation 5.28) is the desired number of consultations of the output to achieve recall \( r \), and \( C_R \) (Equation 5.29) is the actual number of consultations to achieve recall \( r \).

ESR does not rely on ideality, and so neither does \( SRPRUM \). Similarly, assuming binary relevance of judged trees, \( SRPRUM \) can be applied to any (SDR and beyond) search task that can be modelled as tree retrieval.

### 5.3 Calculating ESR Measures

Let us now continue our illustrative example of ESR evaluation. In Section 3.5, we introduced a toy collection (Figure 3.2(a)), a set of relevance judgments for both binary relevance and relevance by length of nodes \( e_3 \) and \( e_4 \) (Table 3.5), a navigation model for the collection (Table 3.6), and three example outputs R1, R2, and R3 (Table 3.7) where System 3 (\( R_3 \)) is the best system, System 1 (\( R_1 \)) is the second-best system, and System 2 (\( R_2 \)) is the worst system. In Section 5.1, we calculated the ESR expected gains from
hits, misses and near-misses across the rank positions of each example system output (as summarized in Table 5.3). In this section, we demonstrate how we use our ESR expectations to calculate the ESR measures proposed in this section.

The first step in this demonstration is calculating SR, HiXEval, XCG and PRUM for System 3 ($R3 = e_3, e_1, e_4$) at rank cut-off $k = 2$. We begin by calculating structural relevance in precision (ESRP) and structural relevance in recall (ESRR), assuming binary relevance. Referring to the expected gains for $R3_2$ found in Row 6 of Table 5.2, the sum of the hits is

$$E[\text{Hits}; R3_2, A] = 1 + 0 = 1;$$

sum of the near-misses is

$$E[\text{NearMisses}; R3_2, A] = 0 + 0.11 = 0.11;$$

and the sum of the misses is

$$E[\text{Misses}; R3_2, A] = 0 + 0.89 = 0.89.$$

From Row 3 of Table 5.3, the recall-base is

$$E[\text{RecallBase}; R3_2, A] = 2.$$

Thus, using Equation 5.14, the precision is

$$ESRP(R3, A)@2 = E[\text{Hits}; R3_2, A]/2 = 0.5.$$

Furthermore, using Equation 5.15, the recall is

$$ESRR(R3, A)@2 = (1 + 0.11)/2 = 0.555.$$

Table 5.5 tabulates ESRP and ESRR across rank cut-offs for all systems.

Next, we calculate interpolated precision (SRiP) and recall (SRiR) in HiXEval, assuming relevance by length. From Row 6 of Table 5.2, the sum of hits for $R3_2$ is

$$E[\text{Hits}; R3_2, A] = 30 + 0 = 30.$$
Table 5.5: Structural Relevance in Precision (ESRP) and Recall (ESRR).

From Row 3 of Table 5.3, the recall-base is $E[\text{RECALL BASE}; R3_2, A] = 50$. Interpolated precision is $SRiP@2 = 30/130 = 0.231$ (Equation 5.19). Interpolated recall is $SRiR@2 = 30/50 = 0.6$ (Equation 5.20). Table 5.6 tabulates SRiP and SRiR across rank cut-offs for all systems.

Next, we use normalized cumulated gain in ESR (NSRCG) to calculate XCG (NXCG). Again, assume relevance by length. Again, we refer to expected gains for $R3_2$ shown in Row 6 of Table 5.2 and the recall-base in Row 3 of Table 5.3. Let the desired recall be $l = 100\%$ and let the desired effort be $m = 2$ ranks. The desired cumulated gain is $CD[2] = 2 \times 100\% \times 50/2 = 50$ (Equation 5.25). Similarly, the user’s expected gain is $CG[2] = 30$ (Equation 5.24). Normalized cumulated gain is $NSRCG[2] = CG[2]/CD[2] = 30/50 = 0.6$ (Equation 5.26). Table 5.7 tabulates $CD$, $CG$, and $NSRCG$ across rank cut-offs for all systems.

Finally, we calculate PRUM using precision-recall with user modelling in ESR ($SRPRUM$). Assume binary relevance values using expected gains for $R3_2$ found in Row 6 of Table 5.2 and the recall-base in Row 3 of Table 5.3. Let the required number of ranks to achieve the desired recall be $C_R = 2$ (Equation 5.29), which corresponds to a desired recall of $l \geq 0.555$ using ESRR in Table 5.5. The expected number of rank positions where the user gains relevant information at $C_R = 2$ is $CL(2) = 1 + 0.11 = 1.11$ (Equation 5.28). Precision-recall with user modelling in ESR is $SRPRUM = 1.11/2 = 0.555$ (Equation 5.30). Table 5.8 tabulates $SRPRUM$ at a desired recall of $l = 100\%$ for all systems.
Table 5.6: Precision and recall for HiXEval using ESR.

<table>
<thead>
<tr>
<th></th>
<th>SRiP@1(SRiR@1)</th>
<th>SRiP@2(SRiR@2)</th>
<th>SRiP@3(SRiR@3)</th>
</tr>
</thead>
<tbody>
<tr>
<td>System 1</td>
<td>0 (0)</td>
<td>0.194 (0.558)</td>
<td>0.287 (1)</td>
</tr>
<tr>
<td>System 2</td>
<td>0 (0)</td>
<td>0 (0)</td>
<td>0 (0)</td>
</tr>
<tr>
<td>System 3</td>
<td>1 (0.6)</td>
<td>0.231 (0.6)</td>
<td>0.319 (1)</td>
</tr>
</tbody>
</table>

Table 5.7: XCG using ESR.

\[
\frac{CG[k]}{CD[k]} = NSRCG[k]
\]

<table>
<thead>
<tr>
<th></th>
<th>k=1</th>
<th>k=2</th>
<th>k=3</th>
</tr>
</thead>
<tbody>
<tr>
<td>System 1</td>
<td>0/25 = 0</td>
<td>25.2/45.2 = 0.56</td>
<td>25.2/64.5 = 0.39</td>
</tr>
<tr>
<td>System 2</td>
<td>0/25 = 0</td>
<td>0/50 = 0</td>
<td>0/75 = 0</td>
</tr>
<tr>
<td>System 3</td>
<td>30/25 = 1.2</td>
<td>30/50 = 0.6</td>
<td>30/71.7 = 0.42</td>
</tr>
</tbody>
</table>

Table 5.8: PRUM using ESR.

<table>
<thead>
<tr>
<th></th>
<th>System 1</th>
<th>System 2</th>
<th>System 3</th>
</tr>
</thead>
<tbody>
<tr>
<td>SRPRUM</td>
<td>0.577</td>
<td>0.129</td>
<td>0.63</td>
</tr>
</tbody>
</table>
Tables 5.5, 5.6, 5.7, and 5.8 show our results for ESRP/ESRR, SRiP/SRiR, NSRCG, and SRPRUM, respectively, for all systems. We can observe the following. System 2 does not retrieve hits. At rank cut-off 3 (which is the maximum recall for all systems), for SR, HiXEval and PRUM, the system ranking is \( R_3 \succ R_1 \succ R_2 \), where \( \succ \) denotes the left-hand system performing better than the right-hand system. Using XCG, System 1 and 3 are tied and the system ranking is \( (R_3, R_1) \succ R_2 \). As expected, in Section 3.5 both Systems 1 and 3 outperformed System 2 for all measures. Similarly, in terms of precision, System 3 outperforms System 1 and is the best system. Using XCG, System 3 outperforms System 1. Overall, we obtain the predicted ranking of systems.

### 5.4 Additional Measures

For comparative purposes, we include additional measures. Our ESR measures NSRCG, SRiP, and SRiR do not consider near-miss gain. We propose additional measures that account for near-miss gain. Below, we re-formulate \( iP \) (Equation 5.17) and \( iR \) (Equation 5.18) in HiXEval to ESR measures that include near-misses. For interpolated precision \( (iP) \) and recall \( (iR) \), respectively, we propose the following:

\[
SRiP_2(R, A) = \frac{E[\text{Hits}; R, A] + E[\text{NearMisses}; R, A]}{\sum_{i=1}^{k} \text{size}(t_i)} \tag{5.31}
\]

\[
SRiR_2(R, A) = \frac{E[\text{Hits}; R, A] + E[\text{NearMisses}; R, A]}{T_{rel}} \tag{5.32}
\]

where \( \text{size}(t_i) \) is the number of characters in tree \( t_i \), \( T_{rel} = \sum_{a \in A} \text{rel}(a) \) is the number of characters in the relevant text in the collection, and \( \text{rel}(a) \) is the number of characters in the relevant text of tree \( a \in A \).

Normalized extended cumulated gain (\( NXCG \) in Equation 5.23) including near-misses (\( NSRCG_2 \)) is proposed as follows:
\[
CG[i] = E[\text{Hits}; R_i, A] + E[\text{NearMisses}; R, A] \tag{5.33}
\]
\[
CD[i] = i \times l \times E[\text{Recall Base}; R_i, A]/m \tag{5.34}
\]
\[
NSRCG2[i] = \frac{CG[i]}{CD[i]} \tag{5.35}
\]

where \(i \in [1, k]\) is the number of ranks consulted, \(l \in (0, 1]\) is the desired recall, and \(m\) is the desired effort.

Our additional measures are akin to generalized measures (defined in HiXEval in Section 2.1) because near-misses in ESR account for gain that users derive from navigating in retrieved documents. We summarize our proposed ESR measures in Table 5.9.

### 5.5 Summary

In this chapter, we have demonstrated how current SDR measures can be formulated in ESR. Table 5.9 summarizes the original measures (as proposed at INEX) and the corresponding ESR measures. When formulated within ESR, the resulting measures model SDR as tree retrieval and thereby may differ from the original measures. We have however shown that calculating the expected gain from hits using the probabilistic approach for navigation in Chapter 3, upon which ESR is based, is a reliable performance measure with respect to HiXEval (published in Ali, Consens, Kazai & Lalmas [7] with the experimental results shown in this work in Section 4.4). In Chapter 4 we showed that our pillars of SDR evaluation was stable with respect to XCG. The goal of this work is to provide a framework in which new measures for SDR evaluation can be developed. Our intention in this section was to show how current SDR measures could be (directly) expressed within our ESR framework.

As discussed at the outset of this chapter, our motivation was to overcome the inconsistencies that arose from the development of task-specific measures in SDR. Without
ESR, we are limited to comparing SDR measures largely in terms of system ranking (comparative system performance). However, our ESR measures enable us to compare measures in more granular terms of how users experience relevance gain, how users experience redundancy, and how users prefer to navigate to relevant information. This is significant given the challenge evaluating the broad range of SDR search tasks and the high cost to validate stable measures.

The benefits of ESR stem from SDR measures that are inherently comparable because they rely on the same set of parameters, namely $E[\text{Hits}; R, A]$, $E[\text{Misses}; R, A]$, $E[\text{NearMisses}; R, A]$, and $E[\text{RecallBase}; R, A]$ (Equations 5.8, 5.9, 5.10, and 5.12 respectively). In addition, ESR provides a convenient way for measures to share common models of relevance value and user navigation; provides a means to apply evaluation approaches developed for different paradigms to tree retrieval; and, system performance can be compared in both general terms of how users gain relevant information (via hits, misses and near-misses) and how a system fulfills a specific search task (via task-specific measures). We believe that the flexibility to support such varied measures in a single framework is an important advancement for the development and evaluation of complex search tasks. In the next chapter, we will complete our study by validating our ESR measures and, more importantly, showing how compatibility of measures across tasks can be tested using low-cost simulation (in lieu of costly user studies).
## Structural Relevance (SR)

### INEX

**Structural Relevance in Precision (SRP)**

\[
SRP(R) = \sum_{i=1}^{k} rel(t_i) \times (1 - p(t_i; R_{i-1}))
\]

Eq. 4.3

### ESR

**Precision (ESRP) & Recall (ESRR)**

\[
ESRP = \frac{E[\text{Hits}; R, A]}{k}
\]

Eq. 5.14

\[
ESRR = \frac{E[\text{Hits} \cap \text{NearMisses}; R, A]}{E[\text{RecallBase}; R, A]}
\]

Eq. 5.15

## Highlighting XML Retrieval Evaluation (HiXEval)

### INEX

**Interpolated Precision (iP@r) and Recall (iR@r)**

\[
iP@r = \sum_{i=1}^{r} \text{size}(e_i)
\]

\[
iR@r = \sum_{i=1}^{r} \text{size}(e_i)
\]

Eq. 5.17

Eq. 5.18

### ESR

**Precision (SRiP†, SRiP2) & Recall (SRiR†, SRiR2)**

\[
SRiP = \frac{E[\text{Hits}; R, A]}{\sum_{i=1}^{k} \text{size}(t_i)}
\]

Eq. 5.19

\[
SRiP2 = \frac{E[\text{Hits}; R, A] + E[\text{NearMisses}; R, A]}{\sum_{i=1}^{k} \text{size}(t_i)}
\]

Eq. 5.31

\[
SRiR = \frac{E[\text{Hits}; R, A]}{\text{TR}_{rel}}
\]

Eq. 5.20

\[
SRiR2 = \frac{E[\text{Hits}; R, A] + E[\text{NearMisses}; R, A]}{\text{TR}_{rel}}
\]

Eq. 5.32

## eXtended Cumulated Gain (XCG)

### INEX

**Normalized Extended Cumulated Gain (NXCG)**

\[
xCG[k] = \sum_{i=1}^{k} xG[i]
\]

Eq. 5.21

\[
xCI[k] = \sum_{i=1}^{k} xI[i]
\]

Eq. 5.22

\[
NXCG[k] = \frac{xCG[k]}{xCI[k]}
\]

Eq. 5.23

### ESR

**Desired Cumulated Gain (NSRCG† & NSRCG2)**

\[
CG[i] = E[\text{Hits}; R_i, A]
\]

Eq. 5.24

\[
CD[i] = i \times l \times E[\text{RecallBase}; R_i, A]/m
\]

Eq. 5.25

\[
NSRCG[i] = \frac{CG[i]}{CD[i]}
\]

Eq. 5.26

\[
NSRCG2[i] = \frac{E[\text{Hits}; R_i, A] + E[\text{NearMisses}; R_i, A]}{CD[i]}
\]

Eq. 5.35

## Precision Recall User Modeling (PRUM)

### INEX

**Precision Recall User Modeling (PRUM)**

\[
PRUM = \frac{E[\# \text{ of rank pos. user sees ideal}]}{E[\# \text{ of rank pos. consulted}]}
\]

Eq. 5.27

### ESR

**PRUM With Desired Recall (SRPRUM)**

\[
CL(i) = E[\text{Hits} \cap \text{NearMisses}; R_i, A]
\]

Eq. 5.28

\[
C_R = m, \text{ where } ESRR(R_m, A) \geq r
\]

Eq. 5.29

\[
SRPRUM = CL(C_R)/C_R
\]

Eq. 5.30

---

1 These measures only account for hits.

Table 5.9: Summary of SDR measures
Chapter 6

Compatible Measures

Compatible measures are stable for the same set of related search tasks. However, establishing compatibility can be costly because, for each task, a user study must be undertaken to validate stability (as described in our experimental method in Section 1.3). In this chapter, we address this issue by proposing a low-cost simulation-based stability test called Scenario Stability Analysis that can be used in lieu of user studies.

The outline of this chapter is as follows. In Section 6.1, we present Scenario Stability Analysis. In Section 6.2, using scenario stability analysis, we predict the stability of the ESR measures proposed earlier in Chapter 5. Finally, in Section 6.3, we verify that our stability predictions match experimental validation using data from INEX 2006 and 2007.

6.1 Scenario Stability Analysis

Scenario Stability Analysis is a test for predicting the stability of an SDR measure. Using simulated SDR outputs, we compare whether an experimental measure is similar to a known stable measure. We determine similarity based on the four properties of bias, range, relative performance and overall performance. We combine our properties to make an overall determination of whether one measure is similar to another. If a measure
is stable and our measure is similar to it, then we predict that our measure is likewise stable.

In Section 6.1.1, we introduce our properties of bias, range, relative performance and overall performance. In Section 6.1.2, we show how we represent SDR search tasks in our simulation. Finally, in Section 6.1.3 we present our simulation.

6.1.1 Properties of SDR Measures

We test stability by showing that a measure is “similar” to a known stable measure. We denote similarity as being either strong, weak, or not similar. *Strong similarity* implies that our experimental measure captures performance similarly to our stable measure. *Weak similarity* implies that our experimental measure captures performance similarly to our stable measure however only under certain conditions. *No similarity* implies that our experimental measure does not capture performance similarly to our stable measure.

We propose four properties by which to compare our SDR measures: bias, range, relative performance and overall performance. There are many other IR measure properties in the classical literature that can be considered in describing how two measures are similar, including reflexivity, transitivity, numeric equivalence, monotonicity [20], Archimedean property [20], order equivalence [73], system ranking similarity [21], the Thomsen condition [119], independent user preferences [49, 107], and additive conjoint structures [119]. We did not choose classical properties for this work because they do not consider the user’s willingness to spend effort vis à vis navigation and redundancy.

Our properties exploit how most SDR measures predictably increase (and decrease) with changes in the model parameters that represent the user’s willingness to spend effort.

**Definition 6.1.1 (Upper-bounded measure)**  *Let an upper-bounded measure denote a measure parameterized to model a user that prefers to spend maximal effort, e.g. navigates to all accessible relevant information.*
Definition 6.1.2 (Lower-bounded measure) Let a lower-bounded measure denote a measure parameterized to model a user that prefers to spend minimal effort, e.g. no navigation.

An upper-bounded (lower-bounded) measure indicates that the user is willing to spend maximal (minimal) effort, and not that the measure score is upper-bounded (lower-bounded). For instance, in Section 5.2, we show how current measures (XCG, PRUM and HiXEval) and our ESR framework differ in terms of accounting for gain from hits (where users do not spend effort navigating to experience gain) and near-misses (where users spend effort navigating to experience gain).

Let us now define the notation for a measure in our simulation.

Definition 6.1.3 (Measure in Simulation) Let $M$ denote the score from a given IR measure where $M \geq 0$. A greater measure score indicates higher performance, and vice versa. For a given output $R$, let $U$ and $L$ denote, respectively, the upper- and lower-bounded scores for our measure where $\min(U,L) \leq M \leq \max(U,L)$.

We note that our definition of a measure for simulation uses a simple function of effort. Our simulation is not applicable for measures with more complex functions of effort. However, our definition above is sufficient for the measures considered in this work.

We now introduce our four measure properties of bias, range, relative performance and overall performance. For each property, we show how our proposed property relates to SDR performance; how we determine similarity between measures by exploiting upper- and lower-bounded measures; and, how similarity is akin to classical notions of measure equivalence.

Property 6.1.4 (Bias) The bias of a measure specifies whether the measure’s score is greater by either upper-bounding or lower-bounding effort. A measure can be upper biased
(scores higher with upper-bound measure), lower biased (scores higher with lower-bound measure), and not biased (otherwise).

We determine bias for a measure $M$ by taking the difference of its upper- and lower-bounded scores ($U$ and $L$, respectively):

\[ \text{bias} = U - L. \tag{6.1} \]

We identify the bias direction as being upper bias (1), lower bias ($-1$) or no bias (0) by taking the sign of the bias defined above in Equation \ref{bias}. We write

\[ b = \text{sgn} \text{bias}. \tag{6.2} \]

If the biases of two measures, which we shall refer to as the reference measure and the current measure, respectively, are in the same direction then the orientation of the upper-bounded and lower-bounded measures are the same for the reference and the current measure. If the biases are not in the same direction then the stability of our current measure will be sensitive to changes in effort parameterization in the reference measure. If the reference measure has no bias, then the stability of the current measure is not sensitive.

Let $A$ and $B$ be measures as defined in Definition \ref{measures} where $A$ denotes the current measure and $B$ the reference measure. Let $\bar{b}_A$ and $\bar{b}_B$ be the bias direction in Equation \ref{bias} averaged over a set of outputs. We write the discrimination function $\beta^*$ to determine whether the bias of our current measure is generally in the same direction as our reference measure, as follows

\[ \beta^* = \begin{cases} 
    \bar{b}_A & \text{if } \bar{b}_B = 0, \text{ [no bias in reference]} \\
    \bar{b}_B & \text{if } \bar{b}_A = 0, \text{ [no bias in current]} \\
    ||\bar{b}_B|| & \text{if } \bar{b}_A \times \bar{b}_B > 0, \text{ [same direction]} \\
    -||\bar{b}_B|| & \text{if } \bar{b}_A \times \bar{b}_B < 0. \text{ [opposite direction]} 
\end{cases} \tag{6.3} \]
where $\|\|\|$ denotes absolute value.

Our bias discrimination indicates the agreement in the directions of the bias of our current and reference measures where values about zero indicate that our measures have no bias. We define bias similarity based on the discrimination, as follows,

**Definition 6.1.5 (Bias Similarity)** Let $\beta \in [0, 1]$ denote our bias discrimination threshold. If $\beta^* > \beta$ then our measures have strong similarity. If $\beta^* \in [-\beta, \beta]$ then our measures have weak similarity. Else, $\beta^* < -\beta$ and our measures are not similar.

We consider measures with the same bias akin to having order-based equivalence [73], in that, the measures agree in how they model gain given the user’s preference for spending effort. Next, we define the range property.

**Property 6.1.6 (Range)** The range of a measure $M$ (see Definition 6.1.3) is the normalized distance $d$ between the scores $U$ and $L$. We write

$$d = \frac{\|U - L\|}{\max(U, L)}.$$  \hspace{1cm} (6.4)

We identify two types of range: wide and narrow.

**Definition 6.1.7 (Wide and Narrow Range)** Let $\omega$ be an arbitrary threshold that defines a wide range. If the range is greater than $\omega$ then the range is wide. Otherwise, we say that the range is narrow.

A measure with wider range has greater fidelity to account for changes in effort than a measure with narrow range. Range is not a commutative property because a measure with a wide range can be constrained to obtain a narrow range, whereas, vice versa, this is not possible.

We determine similarity of range given a set of ranked list outputs by calculating the average range (in Equation 6.4) across outputs.
Definition 6.1.8 (Range Similarity) Let \( \bar{d}_A \) and \( \bar{d}_B \) denote the average range (Equation 6.4) of measures \( A \) and \( B \), respectively, for a given set of outputs. Let \( \omega \) be our threshold in Definition 6.1.7. If \( \bar{d}_A, \bar{d}_B > \omega \) then our measures have strong similarity with each other. If \( \bar{d}_A, \bar{d}_B \leq \omega \), then our measures have weak similarity. If \( \bar{d}_A > \omega \) and \( \bar{d}_B \leq \omega \) then measure \( A \) has strong similarity with \( B \) but not vice versa. Finally, if \( \bar{d}_A \leq \omega \) and \( \bar{d}_B > \omega \), then \( A \) is not similar to \( B \).

Range similarity is akin to numerical equivalence, in that, measures with similar range can be constrained to score within the same normalized numerical range. Now, let us define relative performance.

Property 6.1.9 (Relative Performance) The relative performance of a measure is the human judgment of performance indicated by the measure score for a given output.

To compare relative performance across a set of outputs, we encode scores into a vector as follows.

Definition 6.1.10 (Relative Performance Vector) For a measure defined in Definition 6.1.3, let \( M(R) \) denote the score of our measure on ranked list \( R \). For outputs \( R_1, R_2, R_3, \ldots \), let the relative performance vector be \( \Theta(M) = (M(R_1), M(R_2), M(R_3), \ldots) \).

To compare relative performance vectors, we use cosine similarity which is the cosine of the angle between two vectors.

Definition 6.1.11 (Relative Performance Similarity) Let \( \Theta(A) \) and \( \Theta(B) \) be the relative performance vectors defined above in Definition 6.1.10 for measures \( A \) and \( B \), respectively. If the cosine \( \cos(\Theta(A), \Theta(B)) > 0.9 \) then our measures share strong similarity. Otherwise, the measures are not similar.
We do not define weak similarity because cosine similarity does not capture the magnitude of performance. For instance, consider two measures with the same ordering of a set of systems. The relative performance would indicate that the measures are similar. However, one measure may score the systems as high-performance and the other may score the systems as low-performance, i.e. weak similarity. We leave the definition of weak similarity for relative performance for future work. Relative performance is akin to system ranking similarity [21]. We also note that relative performance similarity is a commutative property. Finally, let us define overall performance.

**Property 6.1.12 (Overall Performance)** The overall performance of a measure for a given output is its change in score across fixed rank cut-offs.

We evaluate overall performance, in our work, using two fixed rank cut-offs. We encode the overall performance into a vector which we now define.

**Definition 6.1.13 (Overall Performance Vector)** Let \( M(R_i) \) denote the score at rank \( i \) calculated with measure \( M \) as defined in Definition [6.1.3] for ranked list \( R \). Let \( a \) and \( b \) denote two fixed rank cut-offs. Let \( M'(R) = M(R_b) - M(R_a) \) denote the score difference between rank cut-offs \( a \) and \( b \). For outputs \( R_1, R_2, R_3, \ldots \), let our overall performance vector be denoted as \( \Delta(M) = (M'(R_1), M'(R_2), M'(R_3), \ldots) \).

As in relative performance, we use cosine similarity to compare overall performance vectors. However, we use cosine in \([0.707, 0.9]\) to denote weak similarity. A cosine of 0.707 corresponds to vectors that are within 45° of each other, i.e. within the same quadrant of a vector space. Let us now define overall performance similarity.

**Definition 6.1.14 (Overall Performance Similarity)** Let \( \Delta(A) \) and \( \Delta(B) \) denote overall performance vectors (per Definition [6.1.13] above) for measures \( A \) and \( B \), respectively. If \( \cos(\Delta(A), \Delta(B)) > 0.9 \) then measures \( A \) and \( B \) have strong similarity. If \( \cos(\Delta(A), \Delta(B)) \) \( \in [0.707, 0.9] \) then the vectors are in the same general direction and
we say that measures $A$ and $B$ have weak similarity. Otherwise, our measures are not similar.

To compare overall performance across a range of rank cut-offs, we would need to choose our fixed pairs of rank cut-offs and compare each as a separate property in our analysis. For instance, to compare two measures for overall performance across rank cut-offs $k = 5$, $k = 10$, and $k = 50$, we would evaluate overall performance similarity for the three pairs $(a, b) : (1, 5), (1, 10), (1, 50)$. Overall performance is akin to testing for monotonicity [20] which can be used to differentiate between types of measures such as recall-oriented (monotonic) and precision-oriented (non-monotonic) measures. We note that overall performance is a commutative property in our work.

This completes our properties and we next present the model for our simulation.

### 6.1.2 Scenarios in SDR

We simulate SDR by defining a priori a set of retrieval scenarios that allow us to generate random outputs for each scenario. Scenarios were first introduced by Piwowarski, Gallinari & Dupret in PRUM [94] to model, given an output, how the user may navigate and consult to satisfy their information need. Like PRUM, our scenarios limit user-system interaction to consultation and navigation, which is sufficient to model our pillars in Chapter 3 of relevance, navigation and redundancy.

Each scenario is specified in terms of gain states (how users experience gain in SDR) and dependency relations (how users experience redundancy in SDR). Let us now begin by describing gain states.

**Definition 6.1.15 (Gain State)** A gain state is a property of an output sub-document that specifies whether, by consulting the output and thereby retrieving the sub-document, the user gains from a hit ($h$), near-miss ($n$), hit plus near-miss ($h + n$), or miss ($m$). Hits, misses and near-misses are explained in Section 3.2.
Let us calculate the gain states for a ranked list \( R = t_1, t_2, \ldots, t_k \) of \( k \) sub-documents. For each sub-document \( t_i \) in the output \( R \), there are 4 possible gain states and therefore there exists \( 4^k \) combinations of gain states for the output. For instance, if \( k = 2 \) then there are \( 4^2 = 16 \) pairs of gain states which may be written as follows: \((h, h), (h, n), (h, h+n), (h, m), (n, h), (n, n), (n, h+n), (n, m), (h + n, h), (h + n, n), (h + n, h + n), (h + n, m), (m, h), (m, n), (m, h + n), (m, m)\).

**Definition 6.1.16 (Dependency Relation)** A dependency relation is a binary property of a relevant sub-document and a set of sub-documents. By either consulting the set or by navigating from the sub-documents in the set, if users can see the relevant sub-document then we say that the set is a dependent set. If there does not exist the same relevant sub-document that can be seen via navigation, or otherwise, from all sub-documents in the set, then we say that the set is an independent set.

Let us now combine our dependency relations with gain states to define our scenarios.

**Definition 6.1.17 (Scenario)** A scenario is the combination of a gain state (Definition 6.1.15) with a dependency relation (Definition 6.1.16). Let \((a, b)\) denote a gain state. Let \( S \) be a scenario. We use the notation \( \perp \) to denote the scenario that our gain state applies to an independent set of sub-documents, e.g. \( S = a \perp b \). Conversely, we use the notation \( \rightarrow \) to denote that our gain state applies to a dependent set, e.g. \( S = a \rightarrow b \).

We can use our scenarios to describe how users experience gain. For example, scenario \( S = h \rightarrow h \) represents two dependent hits. Consider an output that fulfills this scenario. The output would consist of hits in ranks 1 and 2. However, because the user does not experience near-miss gain at either rank position, the dependency must be because the retrieved sub-documents overlap. Hence, this scenario describes a system outputting two relevant sub-documents that overlap.
Another example, consider scenario \( S = h \bot h \) which represents two independent hits. The output that fulfills this scenario must contain hits in ranks 1 and 2 however the retrieved sub-documents are distinct and inaccessible by navigation from each other.

Above, we noted that there are 16 combinations of gain states for an output \( R \) of length \( k = 2 \). Overall, we would obtain \( 16 \times 2 = 32 \) possible scenarios. However, in this work, we ignore the following seven dependent scenarios: \( h \rightarrow m, n \rightarrow m, h + n \rightarrow m, m \rightarrow h, m \rightarrow n, m \rightarrow h + n, m \rightarrow m \). We explain this as follows. First, a user cannot distinguish between experiencing no gain or redundancy in scenarios of both dependent misses \( S = m \rightarrow m \) and independent misses \( S = m \bot m \). As a result, we ignore the dependent case. Second, we ignore any dependent set containing a miss because they cannot happen. Consider the scenario \( h \rightarrow m \) (where the first rank position contains a hit and the second rank position contains a miss). We have a contradiction because the user can see the miss from the hit, which, in ESR, means that the miss must be a near-miss. A similar contradiction occurs if we replace the hit with either a near-miss or a hit plus near-miss. Thus, we obtain the 25 scenarios summarized below in Table 6.1 where \( \text{ID} \) refers to our label for each scenario and \( \text{Scenario} \) shows the scenario. For instance, scenario \( S3 \) (at Row 4 in Table 6.1) is \( h + n \rightarrow h \) and it denotes a dependent set consisting of a hit plus near miss in rank 1 and a hit in rank 2. We will explain columns \( \text{Sub-docs} \) and \( \text{Docs} \) in Table 6.1 in Section 6.1.3 where we define our model collection.

To apply dependency relations to outputs of length \( k > 2 \), we partition the output into combinations of sets of 1 or more sub-documents. For each partition, we calculate our gain states and dependency relations. However, in a scenario, there may be multiple dependent sets but only a single independent set because, to the user, the experience of gain is indistinguishable whether the same sub-documents appear in a single independent set or multiple independent sets. Now, dependent sets contain 2 or more sub-documents because a user can only experience redundancy by navigating from two or more retrieved sub-documents. This leads us to the two following simplifications. First, a partition that
### Table 6.1: Proposed 25 retrieval scenarios for modelling navigation.

<table>
<thead>
<tr>
<th>ID</th>
<th>Scenario</th>
<th>Sub-docs</th>
<th>Docs</th>
</tr>
</thead>
<tbody>
<tr>
<td>S1</td>
<td>$h + n \rightarrow h + n$</td>
<td>e1, e3</td>
<td>d1</td>
</tr>
<tr>
<td>S2</td>
<td>$h + n \perp h + n$</td>
<td>e3, e6</td>
<td>d1, d2</td>
</tr>
<tr>
<td>S3</td>
<td>$h + n \rightarrow h$</td>
<td>e3, e1</td>
<td>d1</td>
</tr>
<tr>
<td>S4</td>
<td>$h + n \perp h$</td>
<td>e4, e7</td>
<td>d2, d3</td>
</tr>
<tr>
<td>S5</td>
<td>$h + n \rightarrow n$</td>
<td>e1, e2</td>
<td>d1</td>
</tr>
<tr>
<td>S6</td>
<td>$h + n \perp n$</td>
<td>e6, e8</td>
<td>d2, d3</td>
</tr>
<tr>
<td>S7</td>
<td>$h \rightarrow h$</td>
<td>e7, e7</td>
<td>d3</td>
</tr>
<tr>
<td>S8</td>
<td>$h \perp h$</td>
<td>e7, e9</td>
<td>d3, d4</td>
</tr>
<tr>
<td>S9</td>
<td>$h \rightarrow n$</td>
<td>e9, e10</td>
<td>d4</td>
</tr>
<tr>
<td>S10</td>
<td>$h \perp n$</td>
<td>e7, e10</td>
<td>d3, d4</td>
</tr>
<tr>
<td>S11</td>
<td>$h \rightarrow h + n$</td>
<td>e4, e6</td>
<td>d2</td>
</tr>
<tr>
<td>S12</td>
<td>$h \perp h + n$</td>
<td>e7, e4</td>
<td>d1, d2</td>
</tr>
<tr>
<td>S13</td>
<td>$n \rightarrow h$</td>
<td>e10, e9</td>
<td>d3</td>
</tr>
<tr>
<td>S14</td>
<td>$n \perp h$</td>
<td>e5, e7</td>
<td>d2, d3</td>
</tr>
<tr>
<td>S15</td>
<td>$n \rightarrow n$</td>
<td>e8, e8</td>
<td>d3</td>
</tr>
<tr>
<td>S16</td>
<td>$n \perp n$</td>
<td>e2, e5</td>
<td>d1, d2</td>
</tr>
<tr>
<td>S17</td>
<td>$n \rightarrow h + n$</td>
<td>e2, e1</td>
<td>d1</td>
</tr>
<tr>
<td>S18</td>
<td>$n \perp h + n$</td>
<td>e8, e6</td>
<td>d3, d2</td>
</tr>
<tr>
<td>S19</td>
<td>$m \perp h$</td>
<td>e11, e9</td>
<td>d4, d5</td>
</tr>
<tr>
<td>S20</td>
<td>$m \perp n$</td>
<td>e11, e10</td>
<td>d4, d5</td>
</tr>
<tr>
<td>S21</td>
<td>$m \perp h + n$</td>
<td>e11, e1</td>
<td>d1, d5</td>
</tr>
<tr>
<td>S22</td>
<td>$h \perp m$</td>
<td>e9, e11</td>
<td>d4, d5</td>
</tr>
<tr>
<td>S23</td>
<td>$n \perp m$</td>
<td>e10, e11</td>
<td>d4, d5</td>
</tr>
<tr>
<td>S24</td>
<td>$h + n \perp m$</td>
<td>e1, e11</td>
<td>d1, d5</td>
</tr>
<tr>
<td>S25</td>
<td>$m \perp m$</td>
<td>e11, e11</td>
<td>d5, d5</td>
</tr>
</tbody>
</table>
contains a set with a single sub-document represents the scenario where the single sub-document set is independent and all other sets are dependent. Second, for a partition of \( m \) sets of sub-documents where all sets contain 2 or more sub-documents, we would obtain \( m+1 \) scenarios where we have \( m \) scenarios with one set independent and 1 scenario where none of the sets are independent.

Let us complete our discussion with a short example to illustrate how we calculate scenarios for an output of length \( k > 2 \). Consider the ranked list \( R = t_1, t_2, t_3, t_4 \). For convenience let 1, 2, 3, 4 denote our four sub-documents in the output. For simulation, we have 8 possible partitions of this output as follows: 

- \( P_1 = (1, 2, 3, 4) \)
- \( P_2 = (1)(2, 3, 4) \)
- \( P_3 = (2)(1, 3, 4) \)
- \( P_4 = (3)(1, 3, 4) \)
- \( P_5 = (4)(1, 3, 4) \)
- \( P_6 = (1, 2)(3, 4) \)
- \( P_7 = (1, 3)(2, 4) \)
- \( P_8 = (1, 4)(2, 3) \)

We recall from Section 6.1.2 that an independent relation has 4 gain states whereas a dependent relation has only 3 gain states. For \( P_1 \), we obtain 2 dependency relations where \( P_1 \) is independent (\( 4^4 = 128 \) gain states) or dependent (\( 4^3 = 64 \) gain states) and which represents \( 128 + 64 = 172 \) scenarios. For \( P_2 \), we obtain 1 dependency relation because the set of (1) must be independent and (2, 3, 4) dependent which represents \( 4^1 + 3^3 = 31 \) scenarios. \( P_3, P_4 \) and \( P_5 \), like \( P_2 \), also each represent 31 scenarios. Let us now calculate the scenarios for the partition \( P_6 = (1, 2), (3, 4) \). Our selected partition consists of two sets; the left-hand set (1, 3) and the right-hand set (2, 4). There are three possible dependency relations for our partition: (i) both sets are dependent (\( 3^2 \times 3^2 = 81 \) gain states); (ii) the left-hand set is dependent and the right-hand set independent (\( 3^2 \times 4^2 = 144 \) gain states); and, (iii) the left-hand set is independent and the right-hand set dependent (\( 4^2 \times 3^2 = 144 \) gain states). \( P_6 \) represents \( 81 + 144 + 144 = 369 \) scenarios. \( P_7 \) and \( P_8 \), like \( P_6 \), also each represent 369 scenarios. Therefore, overall our simulation would consist of a total of \( 1 \times 172 + 4 \times 31 + 3 \times 369 = 1403 \) scenarios. This completes our example and our presentation of calculating scenarios.

We limit our simulation tests to outputs of length \( k = 2 \). Scenarios of length \( k = 2 \) are sufficient to represent redundancy in INEX measures in terms of navigation in PRUM [94].
Chapter 6. Compatible Measures

and overlapping elements in XCG [59] and HiXEval [56]. We also note that our simulation can be adapted to handle specific navigation models where users see only some of the same relevant information from each sub-document (such as in hierarchical navigation in PRUM in Footnote 5 in Section 3.5) by adjusting how navigation is calculated in the simulation.

In the next section, we propose our model collection and show how to use our scenarios to generate randomized outputs to simulate SDR.

6.1.3 Model Collection and Randomized Outputs

A model collection allows us to simulate SDR. A model collection consists of a set of scenarios, a collection of sub-documents, relevance scores, and proposed outputs. In practice, based on a set of scenarios, our model collection could be automatically generated. However, in this work, we chose to determine our model collection by hand. We first present our hand-crafted model collection. Then, we present how we simulate SDR by randomizing our collection, relevance scores, and proposed outputs.

Figure 6.1 shows a model collection based on the scenarios in Table 6.1. The collection consists of 5 documents (labelled \(d_1, d_2, d_3, d_4,\) and \(d_5\) respectively). These documents are sub-divided into 11 sub-documents (labelled \(e_1, e_2, \ldots, e_{11}\) respectively). There are 6 relevant sub-documents in the collection: \(e_1, e_3, e_4, e_6, e_7\) and \(e_9\) appearing in the 4

![Figure 6.1: Proposed model collection.](image-url)
documents \(d_1, d_2, d_3\) and \(d_4\).

In simulating SDR, we randomize the collection, the relevance scores, and the outputs. The collection is randomized as follows. Collection documents are assigned a random length. Collection sub-documents are then assigned random lengths and offsets such that they are constrained to the length of their containing documents and do not overlap. The relevance scores of sub-documents are randomized according to a \textit{scale} to simulate different types of relevance. At scale 0, we simulate binary relevance using relevance scores 0 or 1. At scale 1, we simulate graded relevance using relevance scores between 0 and 10. At scale 2 and above, we simulate relevant text length using relevance scores between 0 and \(10^\text{scale}\). The relevance score of a sub-document is always less than or equal to the length of the sub-document. Non-relevant sub-documents always have a relevance score of 0.

To simulate our outputs, we model a retrieved sub-document starting at the same offsets as a collection sub-document. However, to account for overlap in INEX, the length of a relevant, retrieved sub-document is randomized to be able to span adjacent sub-documents in the collection. The relevance score of an output sub-document is calculated by summing the relevance scores of the sub-documents that it spans.

In Figure 6.1 we show the proposed outputs to capture our scenarios in Table 6.1. We use arrows to denote our retrieved sub-documents. The tail of the arrow indicates the sub-document retrieved at rank 1, and the head of the arrow indicates the sub-document retrieved at rank 2. Each arrow is annotated with its associated scenario in Table 6.1. For instance, arrow 5 denotes a proposed output for scenario \(S_5\) where we retrieve relevant sub-document \(e_1\) at rank 1 and non-relevant sub-document \(e_2\) at rank 2. In Table 6.1 column \textbf{Sub-docs} shows the proposed outputs from our model collection in Figure 6.1. Column \textbf{Docs} shows the documents that contain the sub-documents in our proposed outputs.

Consider simulating SDR for scenario \(S_5\) \((h + n \rightarrow n)\) in Table 6.1 based on the
Let $R = t_1, t_2$ denote our simulated ranked list output. Sub-document $e_1$ is relevant. We simulate $t_1$ by generating a sub-document that contains all or part of $e_1$ and possibly part of the lower-adjacent sub-document $e_2$. Sub-document $e_2$ is non-relevant. We simulate $t_2$ by generating a sub-document that contains all or part of $e_2$. This completes our simulation of SDR. In the next section, we use our model collection to simulate SDR and test our ESR measures.

### 6.2 Comparing ESR to INEX Measures

Our simulation captures INEX overlap \cite{63}, redundancy in ESR (described in Section 3.4), different types of relevance (such as binary relevance in element retrieval for PRUM and XCG and relevant text length in HiXEval), and an exhaustive set of 25 scenarios shown in Table 6.1 for outputs of length $k = 2$ that represent different cases of users experiencing gain and redundancy at INEX. Our goal is to simulate SDR to test the stability of our ESR measures for XCG, PRUM, and HiXEval with respect to INEX measures.

Our tests results are calculated based on 1000 iterations across relevance value scales of 0 (binary relevance) to 5 (randomized relevance scores up to $10^5$). We used the ESREval software package presented in Appendix C to both run our simulation and calculate our scenario stability analysis results.

#### 6.2.1 Experimental Measures

We compare our proposed ESR measures presented in Chapter 5 to stable INEX measures. In this study, we also compare $PRUM$ to $ESRP$ (Equation 5.14) which is a measure of precision that does not account for near-misses. Table 6.2 summarizes the measures we consider in our study. For our measures that require ideality, we calculated the ideal ranking based on the ordering of the randomized relevance scores. An ideal element is
### Highlighting XML Retrieval Evaluation (HiXEval)

#### INEX

**Interpolated Precision** \( (iP@r) \) and **Recall** \( (iR@r) \)

\[
iP@r = \frac{\sum_{r=1}^{R} r \text{size}(e_i)}{\sum_{i=1}^{R} \text{size}(e_i)} \quad \text{Eq. 5.17}
\]

\[
iR@r = \frac{\sum_{r=1}^{R} r \text{size}(e_i)}{T_{rel}} \quad \text{Eq. 5.18}
\]

#### ESR

**Precision** \( (SRiP^\dagger, SRiP2) \) and **Recall** \( (SRiR^\dagger, SRiR2) \)

\[
SRiP = \frac{E[\text{Hits}, R, A]}{\sum_{i=1}^{k} \text{size}(t_i)} \quad \text{Eq. 5.19}
\]

\[
SRiP2 = \frac{E[\text{Hits}, R, A] + E[\text{NearMisses}, R, A]}{\sum_{i=1}^{k} \text{size}(t_i)} \quad \text{Eq. 5.31}
\]

\[
SRiR = \frac{E[\text{Hits}, R, A]}{T_{rel}} \quad \text{Eq. 5.20}
\]

\[
SRiR2 = \frac{E[\text{Hits}, R, A] + E[\text{NearMisses}, R, A]}{T_{rel}} \quad \text{Eq. 5.32}
\]

### eXtended Cumulated Gain (XCG)

#### INEX

**Normalized Extended Cumulated Gain** \( (NXCG) \)

\[
xCG[k] = \sum_{i=1}^{K} xG[i] \quad \text{Eq. 5.21}
\]

\[
xCI[k] = \sum_{i=1}^{K} xI[i] \quad \text{Eq. 5.22}
\]

\[
NXCG[k] = \frac{xCG[k]}{xCI[k]} \quad \text{Eq. 5.23}
\]

#### ESR

**Desired Cumulated Gain** \( (NSRCG^\dagger & NSRCG2) \)

\[
CD[i] = i \times l \times E[\text{RecallBase}, R_i, A]/m \quad \text{Eq. 5.25}
\]

\[
NSRCG[i] = \frac{E[\text{Hits}, R_i, A]}{CD[i]} \quad \text{Eq. 5.26}
\]

\[
NSRCG2[i] = \frac{E[\text{Hits}, R_i, A] + E[\text{NearMisses}, R_i, A]}{CD[i]} \quad \text{Eq. 5.35}
\]

### Precision Recall User Modeling (PRUM)

#### INEX

**Precision Recall User Modeling** \( (PRUM) \)

\[
PRUM = \frac{E[\# \text{ of rank pos. user sees ideal}]}{E[\# \text{ of rank pos. consulted}]} \quad \text{Eq. 5.27}
\]

\[
C_R = m, \text{ where } ESRR(R_m, A) \geq r \quad \text{Eq. 5.29}
\]

#### ESR

**PRUM With Desired Recall** \( (SRPRUM) \)

\[
SRPRUM = CL(C_R)/C_R \quad \text{Eq. 5.30}
\]

\[
ESRP = E[\text{Hits}, R_i, A]/k \quad \text{Eq. 5.14}
\]

\[\dagger\] These measures only account for hits.

\[\dagger\] We compare \( ESRP \) to \( PRUM \) although it is not actually derived from \( PRUM \).

---

**Table 6.2: Summary of Comparisons of INEX and SDR Measures**
always relevant and, in our simulation, we assume that relevant sub-documents are also ideal elements.

Measures that do not meet the requirements in Definition 6.1.3 are not applicable for this test. Our proposed measures, presented in Section 5.2, met these conditions. For the INEX measures HiXEval and XCG, we bounded user effort by varying the tolerance-to-irrelavance $\alpha$ parameter (see Equation 5.16). For PRUM and all of our ESR measures, we varied navigation (see Equations 5.27 and 3.12) such that expected gain from near-misses would be either near-zero, i.e. no navigation, or near full relevance value, i.e. user is near certain to navigate.

In our experiments, we used the following settings for our properties. We set our bias discrimination threshold in Definition 6.1.5 to $\beta = 0.1$. For the range threshold in Definition 6.1.7, we use $\omega = 0$ because our INEX measures XCG and HiXEval exhibit zero range in outputs that do not contain overlapping sub-documents which appears in a significant number of our simulated outputs. If we were comparing ESR measures to ESR measures, then $\omega = 0.1$ would probably be more appropriate. We evaluated relative and overall performance similarity for both upper- and lower-bounded measures. The outcome of our experiments are thus that two measures may have both their upper- and lower-bounded measures similar, only their upper-bounded measures similar, only their lower-bounded measures similar, or the measures are not similar.

6.2.2 Results of Bias Comparison

We compare the bias (Property 6.1.4 in Section 6.1.1) of measures using the methodology in Definition 6.1.5. We report the number of scenarios where average bias shows strong, weak or no similarity. We also include our bias discrimination $\beta^*$ in Equation 6.3 to show how our overall similarity was determined for each pair of measures considered.

We summarize our bias comparison results in Table 6.3. Overall, we found strong similarity to INEX measures for the ESR measures NSRCG2, SRiR2 and SRPRUM
Table 6.3: Summary of bias analysis for ESR measures

<table>
<thead>
<tr>
<th>Stable</th>
<th>Experimental</th>
<th>Not Similar</th>
<th>Weak</th>
<th>Strong</th>
<th>Overall ($\beta^*$)</th>
</tr>
</thead>
<tbody>
<tr>
<td>$NXCG$</td>
<td>$NSRCG$</td>
<td>5</td>
<td>13</td>
<td>7</td>
<td>Not similar ($-0.20$)</td>
</tr>
<tr>
<td>$NXCG$</td>
<td>$NSRCG2$</td>
<td>3</td>
<td>7</td>
<td>15</td>
<td>Strong ($0.32$)</td>
</tr>
<tr>
<td>$iP$</td>
<td>$SRiP$</td>
<td>5</td>
<td>7</td>
<td>13</td>
<td>Weak ($-0.031$)</td>
</tr>
<tr>
<td>$iP$</td>
<td>$SRiP2$</td>
<td>3</td>
<td>19</td>
<td>3</td>
<td>Weak ($-0.078$)</td>
</tr>
<tr>
<td>$iR$</td>
<td>$SRiR$</td>
<td>5</td>
<td>19</td>
<td>1</td>
<td>Weak ($-0.066$)</td>
</tr>
<tr>
<td>$iR$</td>
<td>$SRiR2$</td>
<td>3</td>
<td>7</td>
<td>15</td>
<td>Strong ($0.42$)</td>
</tr>
<tr>
<td>$PRUM$</td>
<td>$SRPRUM$</td>
<td>0</td>
<td>6</td>
<td>19</td>
<td>Strong ($0.67$)</td>
</tr>
<tr>
<td>$PRUM$</td>
<td>$ESRP$</td>
<td>8</td>
<td>12</td>
<td>5</td>
<td>Not similar ($-0.27$)</td>
</tr>
</tbody>
</table>

except in cases where users may experience overlap (such as scenario $S7 = h \rightarrow h$). Our measures $SRiP$, $SRiP2$ and $SRiR$ had many weak scenarios and overall display weak similarity. Our measures $NSRCG$ and $ESRP$ were not similar to INEX measures.

We theorize that there are three main differences between ESR measures and INEX measures. First, unlike in XCG and HiXEval where overlap is explicitly penalized, we consider overlap as a special case of redundancy in ESR. Second, ESR and PRUM both account for navigation to near-misses whereas in XCG and HiXEval near-misses are only counted if the near-miss is overlapped with text in the output. Third, desired gain in ESR (presented in Section 5.2.3) is different from ideal gain in PRUM and XCG because, unlike ideal gain, desired gain is sensitive to how user navigation is modelled.

6.2.3 Results of Range Comparison

We compare the range of measures (Property 6.1.6) as described in Section 6.1.1. We averaged the normalized distance in each of our 25 scenarios in Table 6.1 across our iterations to determine strong similarity (ESR measure has wide range), weak similarity
Table 6.4: Summary of range analysis for ESR measures

(both of our measures have narrow range) and no similarity (ESR measure has narrow range and INEX measure has a wide range). INEX measures do not exhibit range in many scenarios. For instance, \textit{NXCG}, \textit{iP}, and \textit{iR} have 6 scenarios where range is greater than 0. \textit{PRUM} has 11 scenarios where range is greater than 0. We summarize our results of our range comparison in Table 6.4. We calculated our overall results in Table 6.4 based on the range determination (strong, weak or no similarity) that contained the most scenarios. Overall, we found strong similarity to INEX measures for the ESR measures \textit{NSRCG2}, \textit{SRiP2}, \textit{SRiR2} and \textit{SRPRUM}. Our measures \textit{NSRCG}, \textit{SRiP}, and \textit{SRiR} have very few scenarios with strong similarity to INEX measures, and overall these ESR measures have weak similarity to INEX measures. \textit{ESRP} was not similar to \textit{PRUM} in 11 scenarios and we conclude that \textit{ESRP} is not similar to \textit{PRUM}.

6.2.4 Results of Relative Performance Comparison

We now compare the relative performance (Property 6.1.9) of our measures. Table 6.5 shows the results of comparing the upper- and lower-bounded measures for INEX measures (on the vertical) compared to ESR measures (on the horizontal). Overall, the
Chapter 6. Compatible Measures

LOWER BOUND SIMILARITY

<table>
<thead>
<tr>
<th></th>
<th>NSRCG</th>
<th>NSRCG2</th>
<th>SRiP</th>
<th>SRiP2</th>
<th>SRiR</th>
<th>SRiR2</th>
<th>ESRP</th>
<th>SRPRUM</th>
</tr>
</thead>
<tbody>
<tr>
<td>PRUM</td>
<td>0.935</td>
<td>0.935</td>
<td>0.95</td>
<td>0.838</td>
<td>0.935</td>
<td>0.935</td>
<td><strong>0.935</strong></td>
<td><strong>0.967</strong></td>
</tr>
<tr>
<td>NXCG</td>
<td>1</td>
<td>1</td>
<td>0.933</td>
<td>0.787</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>0.948</td>
</tr>
<tr>
<td>iP</td>
<td>0.947</td>
<td>0.947</td>
<td><strong>0.988</strong></td>
<td><strong>0.863</strong></td>
<td>0.947</td>
<td>0.947</td>
<td>0.947</td>
<td>1</td>
</tr>
<tr>
<td>iR</td>
<td>1</td>
<td>1</td>
<td>0.933</td>
<td>0.787</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>0.948</td>
</tr>
</tbody>
</table>

UPPER BOUND SIMILARITY

<table>
<thead>
<tr>
<th></th>
<th>NSRCG</th>
<th>NSRCG2</th>
<th>SRiP</th>
<th>SRiP2</th>
<th>SRiR</th>
<th>SRiR2</th>
<th>ESRP</th>
<th>SRPRUM</th>
</tr>
</thead>
<tbody>
<tr>
<td>PRUM</td>
<td>0.935</td>
<td>0.935</td>
<td>0.845</td>
<td>0.848</td>
<td>0.828</td>
<td>0.933</td>
<td><strong>0.832</strong></td>
<td><strong>0.908</strong></td>
</tr>
<tr>
<td>NXCG</td>
<td>1</td>
<td>1</td>
<td>0.935</td>
<td>0.933</td>
<td>0.98</td>
<td>0.907</td>
<td>0.977</td>
<td>0.775</td>
</tr>
<tr>
<td>iP</td>
<td>0.947</td>
<td>0.947</td>
<td><strong>0.987</strong></td>
<td><strong>0.983</strong></td>
<td>0.927</td>
<td>0.877</td>
<td>0.925</td>
<td>0.848</td>
</tr>
<tr>
<td>iR</td>
<td>1</td>
<td>1</td>
<td>0.935</td>
<td>0.933</td>
<td><strong>0.98</strong></td>
<td><strong>0.907</strong></td>
<td>0.977</td>
<td>0.775</td>
</tr>
</tbody>
</table>

Table 6.5: Relative performance comparison between INEX and ESR.

proposed ESR measures show strong similarity to their counterparts in INEX except $SRiP2$ for $iP$ (see Lower Bound Similarity, row $iP$, column $SRiP2$ in Table 6.5), and $ESRP$ for $PRUM$ (see Upper Bound Similarity, row $PRUM$, column $ESRP$ in Table 6.5).

We conclude that the ESR proposals $NSRCG$, $NSRCG2$, $SRiP$, $SRiR$, $SRiR2$, and $SRPRUM$ have strong similarity with both upper- and lower-bounded INEX measures. However, upper-bounded $ESRP$ is not similar to upper-bounded $PRUM$ and lower-bounded $SRiP2$ is not similar to lower-bounded $iP$.

6.2.5 Results of Overall Performance Comparison

We compare the overall performance (Property 6.1.12) of our measures. Table 6.5 shows the results of comparing the upper- and lower-bounded measures for INEX measures (on the vertical) compared to ESR measures (on the horizontal). We observe the following results. Upper-bounded $NSRCG2$ is not similar to upper-bounded $NXCG$ (see Upper
### LOWER BOUND SIMILARITY

<table>
<thead>
<tr>
<th></th>
<th>NSRCG</th>
<th>NSRCG2</th>
<th>SRiP</th>
<th>SRiP2</th>
<th>SRiR</th>
<th>SRiR2</th>
<th>ESRP</th>
<th>SRPRUM</th>
</tr>
</thead>
<tbody>
<tr>
<td>PRUM</td>
<td>0.785</td>
<td>0.785</td>
<td>0.865</td>
<td>0.343</td>
<td>0.392</td>
<td>0.392</td>
<td><strong>0.787</strong></td>
<td><strong>0.887</strong></td>
</tr>
<tr>
<td>NXCG</td>
<td><strong>0.940</strong></td>
<td><strong>0.940</strong></td>
<td>0.718</td>
<td>0.267</td>
<td>0.232</td>
<td>0.228</td>
<td>0.940</td>
<td>0.720</td>
</tr>
<tr>
<td>iP</td>
<td>0.767</td>
<td>0.767</td>
<td><strong>0.953</strong></td>
<td><strong>0.332</strong></td>
<td>0.407</td>
<td>0.405</td>
<td>0.767</td>
<td>0.983</td>
</tr>
<tr>
<td>iR</td>
<td>0.502</td>
<td>0.493</td>
<td>0.267</td>
<td>-0.207</td>
<td>1</td>
<td>1</td>
<td>0.497</td>
<td>0.395</td>
</tr>
</tbody>
</table>

### UPPER BOUND SIMILARITY

<table>
<thead>
<tr>
<th></th>
<th>NSRCG</th>
<th>NSRCG2</th>
<th>SRiP</th>
<th>SRiP2</th>
<th>SRiR</th>
<th>SRiR2</th>
<th>ESRP</th>
<th>SRPRUM</th>
</tr>
</thead>
<tbody>
<tr>
<td>PRUM</td>
<td>0.463</td>
<td>0.762</td>
<td>0.343</td>
<td>0.360</td>
<td>-0.062</td>
<td>0.110</td>
<td><strong>0.493</strong></td>
<td><strong>0.613</strong></td>
</tr>
<tr>
<td>NXCG</td>
<td><strong>0.925</strong></td>
<td><strong>0.705</strong></td>
<td>0.698</td>
<td>0.692</td>
<td>0.235</td>
<td>0.127</td>
<td>0.922</td>
<td>0.243</td>
</tr>
<tr>
<td>iP</td>
<td>0.697</td>
<td>0.420</td>
<td><strong>0.955</strong></td>
<td><strong>0.942</strong></td>
<td>0.412</td>
<td>0.260</td>
<td>0.685</td>
<td>0.295</td>
</tr>
<tr>
<td>iR</td>
<td>0.285</td>
<td>-0.070</td>
<td>0.315</td>
<td>0.297</td>
<td><strong>0.935</strong></td>
<td><strong>0.748</strong></td>
<td>0.245</td>
<td>-0.247</td>
</tr>
</tbody>
</table>

Table 6.6: Overall performance comparison between INEX and ESR.

Bound Similarity, row NXCG, column NSRCG2 in Table 6.6. Lower-bounded SRiP2 is not similar to lower-bounded iP (see Lower Bound Similarity, row iP, column SRiP2 in Table 6.6). Neither upper-bounded ESRP nor upper-bounded SRPRUM are similar to upper-bounded PRUM (see Upper Bound Similarity, row PRUM, columns ESRP and SRPRUM, respectively, in Table 6.6).

We conclude that the ESR proposals NSRCG, SRiP, SRiR, and SRiR2 have strong similarity to their INEX counterparts. The ESR proposals NSRCG2, ESRP and SRPRUM are lower-bound similar to their INEX counterparts. SRiP2 is upper-bound similar to its INEX counterpart.

#### 6.2.6 Summary

Table 6.7 summarizes our observations and Table 6.8 shows our predictions for our ESR measures. Our conclusions are described as follows. NSRCG is not a stable measure for NXCG based on its bias, likely stemming from its range, which together indicates
Table 6.7: Summary of similarity per feature between INEX and ESR measures.

<table>
<thead>
<tr>
<th>Stable</th>
<th>Exper.</th>
<th>Bias</th>
<th>Range</th>
<th>Lower Relative</th>
<th>Upper Relative</th>
<th>Lower Overall</th>
<th>Upper Overall</th>
</tr>
</thead>
<tbody>
<tr>
<td>NXCG</td>
<td>NSRCG2</td>
<td>Not similar</td>
<td>Weak</td>
<td>Strong</td>
<td>Strong</td>
<td>Strong</td>
<td>Strong</td>
</tr>
<tr>
<td>NXCG</td>
<td>NSRCG</td>
<td>Strong</td>
<td>Strong</td>
<td>Strong</td>
<td>Strong</td>
<td>Strong</td>
<td>Not similar</td>
</tr>
<tr>
<td>iP</td>
<td>SRiP</td>
<td>Weak</td>
<td>Strong</td>
<td>Strong</td>
<td>Strong</td>
<td>Strong</td>
<td>Strong</td>
</tr>
<tr>
<td>iP</td>
<td>SRiP2</td>
<td>Weak</td>
<td>Strong</td>
<td>Weak</td>
<td>Strong</td>
<td>Not similar</td>
<td>Strong</td>
</tr>
<tr>
<td>iR</td>
<td>SRiR</td>
<td>Weak</td>
<td>Strong</td>
<td>Strong</td>
<td>Strong</td>
<td>Strong</td>
<td>Strong</td>
</tr>
<tr>
<td>iR</td>
<td>SRiR2</td>
<td>Strong</td>
<td>Strong</td>
<td>Strong</td>
<td>Strong</td>
<td>Strong</td>
<td>Weak</td>
</tr>
<tr>
<td>PRUM</td>
<td>SRPRUM</td>
<td>Strong</td>
<td>Strong</td>
<td>Strong</td>
<td>Strong</td>
<td>Weak</td>
<td>Not similar</td>
</tr>
<tr>
<td>PRUM</td>
<td>ESRP</td>
<td>Not similar</td>
<td>Not similar</td>
<td>Not similar</td>
<td>Strong</td>
<td>Weak</td>
<td>Not similar</td>
</tr>
</tbody>
</table>

problems with capturing overlap and ideality. NSRCG2 is a stable measure for NXCG however its upper-bounded overall performance is not similar due to how it includes near-miss gain and overlap. SRiP is a stable measure for iP however its range indicates problems with capturing overlap. SRiP2 is a stable measure for iP however we note that it will have problems with bias, lower-bounded relative performance, and lower-bounded overall performance. It is likely not a good measure of iP. SRiR is not a stable measure for iR due to its bias and range which suggest problems with capturing overlap. SRiR2 is a stable measure for iR however it is apt to be non-monotonic (based on overall performance). SRPRUM is a stable measure for PRUM however it is apt to be non-monotonic (based on overall performance) although we theorize that this weakness may also be due to differences between desired recall in ESR and ideal elements in PRUM. ESRP is not a stable measure for PRUM due to problems in bias, range, and relative performance. This completes our analysis of INEX and ESR measures using scenario stability analysis. In the next section, we compare our predictions with measures validation results using data from INEX 2006 and 2007.
### 6.3 INEX System Rankings Using ESR Measures

We test the stability of our ESR measures in Table 6.2 by comparing them with INEX measures in evaluating a range of ad-hoc retrieval tasks at the INEX including INEX 2006 Focused Task, INEX 2006 Best In Context Task, INEX 2007 Focused Task, and INEX 2007 Relevant In Context Task.

#### 6.3.1 Experimental Setup

For each ESR measure tested, we determined stability by comparing the system rankings from ESR to the official INEX results using Spearman’s Rho. Spearman’s Rho ($\rho$) indicates whether two separate rankings are positively ($\rho > 0$) or negatively ($\rho < 0$) ordered. If $\rho = 1$, then the ranked lists are in exactly the same order. If $\rho = -1$, then the ranked lists are in exactly opposite order. The p-value is the probability that the compared rankings are not correlated. If a p-value is less than 0.05 then two measures are correlated in terms of how they order systems. So, in comparing system rankings for a current measure versus a reference measure, the rankings will be either positively

<table>
<thead>
<tr>
<th>Stable</th>
<th>Experimental</th>
<th>Prediction</th>
<th>Problems</th>
</tr>
</thead>
<tbody>
<tr>
<td>$NXCG$</td>
<td>$NSRCG$</td>
<td>Unstable</td>
<td>Overlap</td>
</tr>
<tr>
<td>$NXCG$</td>
<td>$NSRCG2$</td>
<td>Stable</td>
<td>Overlap, Sensitive to navigation</td>
</tr>
<tr>
<td>$iP$</td>
<td>$SRiP$</td>
<td>Stable</td>
<td>Overlap</td>
</tr>
<tr>
<td>$iP$</td>
<td>$SRiP2$</td>
<td>Stable</td>
<td>Overlap, Sensitive to navigation</td>
</tr>
<tr>
<td>$iR$</td>
<td>$SRiR$</td>
<td>Unstable</td>
<td>Overlap, Sensitive to navigation</td>
</tr>
<tr>
<td>$iR$</td>
<td>$SRiR2$</td>
<td>Stable</td>
<td>Sensitive to navigation</td>
</tr>
<tr>
<td>$PRUM$</td>
<td>$SRPRUM$</td>
<td>Stable</td>
<td>Ideality, Sensitive to navigation</td>
</tr>
<tr>
<td>$PRUM$</td>
<td>$ESRP$</td>
<td>Unstable</td>
<td></td>
</tr>
</tbody>
</table>

Table 6.8: Summary of stability predictions for ESR measures.
correlated, negatively correlated, or not correlated.

To calculate recall-points for ESR measures $SRiP$ (Equation 5.19) and $SRiP2$ (Equation 5.31), we selected $SRiR2$ (Equation 5.32). To calculate recall-points for ESR measure $ESRP$ (Equation 5.14), we selected $ESRR$ (Equation 5.15).

Our ESR measures use a common navigation model. We selected the depth-weighted summary navigation model shown in Table 6.9 in Section 4.3.2 and originally published in Ali, Consens & Larsen [10] where we validated that this model captures user effort in terms of how users were observed to spend time reading and navigating among elements in Wikipedia documents. Our model was validated with observations from the user tracking study in Hammer-Aebi et al. [47] which was conducted at the INEX 2006 Interactive Track [80] to assess navigation in the INEX Wikipedia collection\footnote{http://www-connex.lip6.fr/~denoyer/wikipediaXML/} (the same collection used in our current experiments). The navigation graph consists of five partitions (namely, article, section, ss1, ss2, other) where each partition is weighted inversely to the average tag depth of the nodes included in the partition. We used steady-state probabilities for navigation. The ESREval software package presented in Appendix C was used to calculate all of our ESR measures.

### 6.3.2 INEX 2006 Focused Task - nXCG and iP

In the focused task, the system searches for relevant elements or passages such that the returned answers do not contain overlapped text.

We evaluated 43 participating systems across 107 topics in the INEX 2006 Ad-hoc
Chapter 6. Compatible Measures

Focused Track for \( k = 1000 \) using the ESR measures \( NSRCG \) (Equation 5.26 at rank cut-offs), \( NSRCG2 \) (Equation 5.35 at rank cut-offs), \( MASriP \) (\( SRiP \) in Equation 5.19 mean-averaged over recall points calculated using \( SRiR \) in Equation 5.20), and \( MASriP2 \) (\( SRiP2 \) in Equation 5.31 mean-averaged over recall points calculated using \( SRiR2 \) in Equation 5.32). The reported INEX measures for this task are \( MAiP \) (\( iP \) in Equation 5.17 mean-averaged over recall points), \( NXCG \) at rank cut-offs (shown in Equation 5.23) with overlap ON and OFF, respectively. For \( MAiP \), we evaluated systems up to the rank cut-off of \( k = 1000 \). For \( NXCG \), we compared it to ESR measures at rank cut-off \( k = 10 \).

The systems evaluated included the top-30 officially best systems (as determined using \( NXCG \)) and 13 randomly selected systems.

Table 6.10 shows the system ranking comparison results between the original (INEX) and ESR measures using Spearman’s Rho and p-value (in parentheses). The system rankings from \( NSRCG \) are negatively ordered \((\rho < 0)\) with \( NXCG \) OFF (where user tolerates overlaps). With overlap ON, system rankings via \( NSRCG \) are not correlated \((p\text{-value greater than 0})\). System rankings via \( NSRCG2 \) are negatively ordered \((\rho < 0)\) with \( NXCG \) OFF and not correlated \((p\text{-value greater than 0.05})\). However, with overlap ON, \( NSRCG2 \) is positively ordered \((\rho > 0)\) with \( NXCG \) and correlated \((p\text{-value less than 0.05})\).

<table>
<thead>
<tr>
<th></th>
<th>( MASriP )</th>
<th>( MASriP2 )</th>
</tr>
</thead>
<tbody>
<tr>
<td>( MAiP )</td>
<td>0.43(0.005)</td>
<td>0.69(0.00)</td>
</tr>
<tr>
<td>( NSRCG[10] )</td>
<td>( NSRCG2[10] )</td>
<td></td>
</tr>
<tr>
<td>( NXCG[10] ) OFF</td>
<td>-0.37(0.016)</td>
<td>-0.005(0.973)</td>
</tr>
<tr>
<td>( NSRCG[10] )</td>
<td>( NSRCG2[10] )</td>
<td></td>
</tr>
<tr>
<td>( NXCG[10] ) ON</td>
<td>-0.114(0.463)</td>
<td>0.34(0.028)</td>
</tr>
</tbody>
</table>

Table 6.10: INEX 2006 Focused Task, 107 topics, 43 Systems, \( k=1000 \).
The system rankings via \( \text{MASRiP} \) and \( \text{MASRiP2} \) are positively ordered (\( \rho > 0 \)) and positively correlated (\( \rho > 0 \), p-value less than 0.05) with those via \( \text{MAiP} \).

Thus, in the focused task, \( \text{NSRCG} \) and \( \text{NSRCG2} \) are not stable measures of \( \text{NXCG} \) OFF. But, \( \text{NSRCG2} \) is with respect to \( \text{NXCG} \) ON, whereas \( \text{NSRCG} \) is not. In addition, both \( \text{MASRiP} \) and \( \text{MASRiP2} \) are both stable measures of \( \text{iP} \) with overlap ON. This agrees with our predictions in Table 6.8 in Section 6.2 where \( \text{NSRCG2}, \text{SRiP} \) and \( \text{SRiP2} \) are found to be stable, although in our simulation \( \text{SRiP} \) is the better measure. Based on the negative ordering of ESR measures for \( \text{NXCG} \) with overlap OFF, we theorize that ESR, as defined thus far, does not capture users who tolerate seeing overlapped information as defined at INEX. This is likely because we have limited our consideration of user navigation, in this work, to between nodes and not within the same node as described in Section 3.3.4.

### 6.3.3 INEX 2006 Best In Context Task - EPRUM

In the best-in-context task, the system searches for the elements or passages that represent the best point(s) from which to navigate to all of the relevant information in retrieved documents.

We evaluated 64 participating systems across 107 topics in the INEX 2006 Best In Context Track for \( k = 1000 \) using \( \text{ESRR} \) (Equation 5.15) for calculating recall points. The official INEX measure for this task is \( \text{EPRUM} \) (shown in Equation 2.3 in Section 2.1) which is a simplified version of the \( \text{PRUM} \) measure. Navigation in \( \text{EPRUM} \) uses a proximity measure based on the scalar parameter \( A \), where \( A \) represents the distance, in the document, that a user will navigate from a given entry point to locate relevant information. For instance, \( A = 0.1 \) refers to a user willing to navigate to information very close to the entry point, whereas \( A = 1000 \) refers to a user willing to navigate to information much further away to the entry point.

Table 6.11 shows the system ranking comparison results between INEX and ESR.
measures using Spearman’s Rho and p-value (in parentheses). The system rankings from SRPRUM are positively ordered ($\rho > 0$) and positively correlated ($\rho > 0$, p-value less than 0.05) to \textit{EPRUM} for all values of $A$. The system rankings from \textit{ESRP} are positively ordered ($\rho > 0$) but uncorrelated (p-value greater than 0.05) to \textit{EPRUM} for all values of $A$. We conclude that \textit{SRPRUM} is a stable measure of \textit{EPRUM} and \textit{ESRP} is not for the best in context task. This agrees with our predictions in Table 6.8 where \textit{SRPRUM} is stable and \textit{ESRP} is unstable.

### 6.3.4 INEX 2007 Focused Task - iP

We evaluated 77 participating systems across 102 topics in the INEX 2007 Ad-hoc Focused Track for $k = 1000$ using the ESR measures \textit{MASRiP} (\textit{SRiP} in Equation 5.19 mean-averaged across recall points calculated using \textit{SRiR2} in Equation 5.32 up to rank $k = 1000$); \textit{SRiP} at recall points calculated using \textit{SRiR2}; \textit{MASRiP2} (\textit{SRiP2} in Equation 5.31 mean-averaged across recall points calculated using \textit{SRiR2} in Equation 5.32 up to rank $k = 1000$); and, \textit{SRiP2} at recall points calculated using \textit{SRiR2}. The official measure for this task is \textit{iP} with overlap ON (Equation 5.17) at the recall point 0.01 calculated using \textit{iR} (Equation 5.18).

Table 6.12 shows the system ranking comparison results between INEX and ESR measures using Spearman’s Rho and p-value (in parentheses). \textit{MASRiP}, \textit{MASRiP2}, \textit{SRiP}, and \textit{SRiP2} are positively ordered ($\rho > 0$) and positively correlated ($\rho > 0$, p-value
Chapter 6. Compatible Measures

<table>
<thead>
<tr>
<th>SRiP_0.01</th>
<th>MASRiP</th>
<th>SRiP_2_0.01</th>
<th>MASRiP_2</th>
</tr>
</thead>
<tbody>
<tr>
<td>iP_0.01</td>
<td>0.53(0.00)</td>
<td>0.72(0.00)</td>
<td>0.27(0.022)</td>
</tr>
</tbody>
</table>

Table 6.12: INEX 2007 Focused Task, 102 topics, 77 Systems, k=1000.

less than 0.05) to iP. The mean-averaged ESR measures (MASRiP and MASRiP\_2) have better rank correlation (\(\rho\) is higher) than their corresponding rank cut-off measures (SRiP and SRiP\_2, respectively). These results agree with our validation results above in Section 6.3.2 where we concluded that SRiP and SRiP\_2 are stable measures of iP for the focused task. This agrees with our predictions in Table 6.8 where both SRiP and SRiP\_2 are found to be stable.

6.3.5 INEX 2007 Relevant In Context Task - MAgP

In the relevant-in-context task, for each relevant document, the system finds all of the relevant sub-documents that do not overlap and outputs them in the rank order of where they appear in the document.

We evaluated 77 participating systems across 102 topics in the INEX 2007 Relevant in Context Track for \(k = 1000\) using the ESR measures MASRiP (SRiP in Equation 5.19 mean-averaged over recall points calculated using SRiR\_2 in Equation 5.32 up to rank \(k = 1000\)), and MASRiP\_2 (SRiP\_2 in Equation 5.31 mean-averaged across recall points calculated using SRiR\_2 in Equation 5.32 up to rank \(k = 1000\)). The official measure for this task is MAgP which is generalized precision mean-averaged across recall points where recall is measured using generalized recall (gR). In HiXEval \[56\], to account for

<table>
<thead>
<tr>
<th>MASRiP</th>
<th>MASRiP_2</th>
</tr>
</thead>
<tbody>
<tr>
<td>MAgP</td>
<td>0.45(0.00)</td>
</tr>
</tbody>
</table>

Table 6.13: INEX 2007 Relevant In Context Task, 102 topics, 77 Systems, k=1000.
near-misses, generalized measures have been proposed. These measures extend interpolated measures by accounting for user gain at the document-level. For instance, if a system retrieves a passage with $x$ number of relevant characters from a document containing $y$ relevant characters, then, depending on how navigation is modelled, the user is modelled to see between $x$ and $y$ relevant characters from the given document. In this way, navigation is considered beyond retrieved text passages and is akin to near-misses as defined in ESR and PRUM.

Table 6.13 shows the system ranking comparison results between INEX and ESR measures using Spearman’s Rho and p-value (in parentheses). Both $MASRiP$ and $MASRiP2$ are positively ordered ($\rho > 0$) and positively correlated ($\rho > 0$, p-value less than 0.05) to $MAgP$. We conclude that $MASRiP$ and $MASRiP2$ are stable measures of $MAgP$ for the relevant in context task. We theorize that, in general, given our analysis of $SRiP$ and $SRiP2$ in Section 6.2 that $SRiP2$ is the more appropriate measure for this task because it includes near-misses in accounting for gain.
6.4 Conclusions

In this chapter, we presented scenario stability analysis as a way to predict the stability of our ESR measures using simulation to compare an experimental measure to a stable SDR measure across a range of tasks. We used our simulation to compare experimental ESR measures with stable INEX measures. Table 6.2 summarizes the INEX measures ($\text{NXCG}$, $iP$, $iR$ and $\text{PRUM}$) and ESR measures ($\text{NSRCG}$, $\text{NSRCG2}$, $\text{SRiP}$, $\text{SRiP2}$, $\text{SRiR}$, $\text{SRiR2}$, $\text{SRPRUM}$, and $\text{ESRP}$) that we compared. We formally validated the stability of our ESR measures with data from INEX 2006 and 2007. We then compared our predictions using our low-cost scenario stability analysis with our costly validation results using INEX data.

We summarize our predictions and validation results in Table 6.14. Our validation results for $\text{NSRCG}$, $\text{NSRCG2}$, $\text{SRiP}$, $\text{SRiP2}$, $\text{SRPRUM}$ and $\text{ESRP}$ agreed with our predictions, however we were unable to directly compare our recall measures $\text{SRiR}$ and $\text{SRiR2}$. In the validation study, we were limited to testing $\text{SRiR2}$ indirectly by calculating the recall-points for $\text{SRiP}$ and $\text{SRiP2}$. Thus, further testing of our approach may be needed to show that our approach works, in general, for recall-oriented measures. However, we argue that the strong agreement between our predictions and our validation results, in the measures that we did test, are sufficient to demonstrate the effectiveness of replacing validation studies with our simulation-based test, Scenario Stability Analysis.

The main problems we observe with ESR is in how it accounts for users that tolerate overlapped text. We leave this problem for future work, but it could likely be resolved by introducing a further partition in ESR where relevant text in the collection that has been retrieved, i.e., $E[\text{Hits}; R, A]$ in Equation 5.8 is further sub-divided into overlapped and non-overlapped partitions. This would allow measures to account for overlapped text and possibly penalize performance using a tolerance-to-irrelevance parameter $\mu$.

Simulation is not as definitive for validating a measure as a user study. However, given our need in SDR for validating stability across many tasks, we believe that simulation
offers a more feasible solution than formal user studies for researchers to improve SDR evaluation in a cost-effective and timely manner.

Overall, we conclude that scenario stability analysis provides a low-cost and effective way to test the stability of measures across a range of related tasks. This completes our goal of developing stable, consistent, and compatible SDR measures.
Chapter 7

Conclusions and Future Work

Our work considers the problem of developing a common set of stable IR measures to evaluate performance across the range of tasks studied in SDR research. A measure is stable if it reliably evaluates system performance the same as a human judge in terms of how well the system serves the user to satisfy the user’s information need. This is an active area of research. Currently, each task in SDR is evaluated using a measure that has been specifically developed and validated as stable for the task. However, each measure captures differently how users judge the performance of an IR system. As a result, it is difficult to compare scores from different measures for the same SDR system.

Our solution is the Extended Structural Relevance (ESR) framework which defines a common basis upon which to evaluate any ad-hoc SDR search task; i.e. tasks where a user seeks verbatim answers from a fixed collection. We show how stable SDR measures can be derived from the ESR framework by proposing measures in ESR that match our current SDR measures. The key benefit is that our ESR measures use a common model of how users judge performance, a common set of measure parameters, and a common data set, called a test collection, upon which to evaluate systems.

In Chapter 3, we proposed a common basis for SDR evaluation that does not rely on ideality and addresses the problem of stability that we identified in Section 2.2. The
two main contributions of our work in this chapter were to define our pillars of SDR evaluation (relevance, navigation and redundancy) and to show how tree retrieval can be used to model a broad range of tasks in SDR. We provide experimental validation of our approach in Chapter 4 where we tested the stability of the measure SR (based on our pillars, albeit limited, to evaluating precision) across different ad-hoc search tasks modelled as tree retrieval.

In Chapter 5, we proposed the Extended Structural Relevance (ESR) framework to address our problem of consistent measures identified in Section 2.3. For SDR evaluation, ESR provides a common basis to calculate the user’s expected gain from hits, misses and near-misses based on relevance, navigation and redundancy. ESR is the first SDR framework in the literature for developing IR measures that are stable across such a broad range of tasks. The main benefit of ESR is that it introduces a consistent way to formulate SDR measures. There are still two outstanding methodological issues in ESR. First, work is needed on how to assess the relevance of trees. Second, earlier research, e.g. Losee [75], Van Rijsbergen [119], and Bollman [20], defines equivalence between measures based solely on relevance, thus, we must revisit how measures are deemed to be equivalent because the structural relevance assumption in Assumption 3.2.3 introduces the additional factors of navigation and redundancy. Our main results show how selected (inconsistent) measures of PRUM, HiXEval and XCG can be formulated in ESR.

Finally, in Chapter 6, we address the problem of compatibility identified in Section 2.4. We propose scenario stability analysis which is a low-cost simulation of SDR to predict the stability of measures across a range of sub-tasks. Its main contribution is to offer a low-cost alternative to costly validation studies. Validation has been a significant problem in SDR where numerous systems and search tasks in the literature have not been evaluated because of both a lack of validated stable measures and the high cost associated with validating a measure for a new task. We used scenario stability analysis to predict the stability of our ESR measures proposed earlier in Section 5.2. Then,
we formally validated our ESR measures using data from INEX 2006 and 2007. The ESReval package (described in Appendix C), written in Java, implements all of the measures presented in this work and is available upon request from the author. Our main result confirmed that our predictions using low-cost simulation agreed closely with costly validation results from INEX 2006 and 2007, and, most importantly, demonstrated the effectiveness of our proposal to replace costly validation with low-cost simulation.

In our dissertation, we have identified numerous areas of future work such as proposing ESR measures (Section 5.2), refining relevance assessments (Section 5.5), developing different navigation models (Section 5.5), and refining ESR to account for overlap (Section 6.2.6) by proposing an additional partition in ESR for overlapped text in hits that incorporates the overlap parameter $\alpha$ in the $rsize$ function in HiXEval and XCG (in Section 5.2.2).

We believe that relevance, user navigation and redundancy are also of concern to search tasks outside of SDR. For instance, within the context of semantic web search such as presented by Guha, McCool & Miller [45] where systems search collections of RDF documents, we are investigating how ESR can be applied to evaluate systems where navigation plays an important role to search collections that do not contain structured documents but, instead, structured information (such as semantic associations and ontologies). Comparing the relative effectiveness of semantic web search systems using classical precision and recall is a well-known challenge (see Aleman-Meza et al. [3], Aleman-Meza et al. [4], and Fernandez et al. [40]). Our belief is that our ESR framework can serve as a basis to define measures for evaluating search tasks across SDR, semantic web search of RDF collections, and many other areas of information access.

A challenge for ESR will be to model user navigation at the scale of the semantic web (as described by Aleman-Meza et al. [4]) or linked data repositories (as described by Fernandez et al. [40]). To these ends, in Section 2.5 we identified a strong motivation for investigating a unification between diversity tasks (see Clarke et al. [30]) and SDR given
the similar need for compatibility in both research areas. Another promising avenue of research in evaluation at the scale of the web is incomplete assessments as described by Baillie, Azzopardi & Ruthven [16]. A key future goal for ESR will be to represent recent proposals of measures for incomplete assessments such as binary preference (see Buckley & Voorhees [22]), rank-biased precision (Moffatt & Zobel [82]), and others (see Sakai [105]; Sakai [104]; and Yilmaz & Aslam [122]), which have not been considered within INEX.

Finally, ESR will need to account for end user behaviour on the web. For instance, consider abandonment behaviour as described by Radlinski, Kurup & Joachims [97] which refers to why (or when) a user navigates away from a given web page. In future work, we need to consider whether issues like abandonment might affect our user model. For instance, it is reasonable that if a user retrieves no relevant information in a sub-document then the user will not likely spend more effort to navigate from it to locate relevant information, whereas, given a highly relevant result then the user will spend maximally to locate more relevant information in the parts of the document not retrieved.
Appendix A

Weak Ordering

Chapter 3 shows how expected relevance value is calculated for strictly ordered lists. Ranked lists can also be modelled as being weakly ordered. Weak ordering refers to the case where users randomly consult components that are tied in rank. In our work Ali, Consens & Lalmas [9], we show how the redundancy of elements, i.e. singletons, can be obtained for the weakly ordered ranked list $R$. In this section, we show how redundancy can be obtained for weakly ordered ranked lists of more general sub-documents. We first review the redundancy of weakly ordered elements. Next, we show how it can be obtained for sub-documents.

From Ali, Consens & Lalmas [9], the probability of element $e$ being seen once by the user consulting the weakly ordered ranked list $R$ can be written as follows:

$$q(e; R) = q(e; \Omega) = \frac{1}{\ell} \cdot \sum_{R' \in \Omega} q(e; R'[e])$$  \hfill (A.1)

where $\ell = |\Omega| = \prod_{i=1}^{l} l_i!$, $l$ is the number of ranks in $R$, and $l_i$ is the number of components in rank $i$ of $R$. $\Omega$ contains all of the possible browsing histories that would satisfy the information need of the user, $R'[e]$ is sublist of $R'$ containing the elements higher-ranked than $e$, and $q(e; R'[e])$ is the probability that element $e$ being seen once by the user given their (strictly ordered) browsing history $R' \in \Omega$. If $\Omega$ contains a single
browsing history (the case where the ranked list is not weakly ordered), then Equation \ref{A.1} reduces to Equation \ref{3.12}.

Prior to showing how this extends to sub-documents, consider the following examples to demonstrate weak ordering.

**Example A.0.1 Consulting a Weakly Ordered Ranked List.** Consider the following ranked list of components, \( R = \{ x \mid y \ z \} \), where there are 2 ranks with the first rank containing component \( x \) and the second rank containing \( y \) and \( z \). If a user required all three component to satisfy their information need then there would be two possible scenarios in which the components in the list would be consulted. These scenarios (shown below as \( R'_1 \) and \( R'_2 \)) represent a user seeking relevant information by consulting the list one component at a time.

\[
R'_1 = \{ x \mid y \mid z \} \\
R'_2 = \{ x \mid z \mid y \}
\]

\( \Omega \) denotes the set of all possible consultation scenarios where the user satisfies their information need by retrieving tied results randomly and retrieves each result in the output once. Let \( \ell \) denote the number of scenarios in \( \Omega \). For a ranked list with \( l \) ranks where \( l_i \) denotes the number of components in rank \( i \) of the list, the total number of consultation scenarios is obtained by \( \ell = \prod_{i=1}^{l} l_i! \). The next example demonstrates our expansion of consultation scenarios for weakly ordered ranked lists.

**Example A.0.2 Expansion of Consultation Scenarios.** Consider the following ranked list \( R = \{ e_1 f g \mid e_2 e_3 e_4 \} \) containing elements \( (e_1, e_2, e_3, e_4) \) where the user may navigate between them. What are the possible scenarios with which the user will consult the ranked list \( R \)?

\textit{Ans.} The first rank in \( R \) is a tied rank with 3 components, the second rank is tied and has 2 components, and the third rank is not tied and contains 1 component. In this
case, there are \( \ell = 3!2!1! = 12 \) possible scenarios for consulting all components in \( R \). Specifically, these are

\[ \Omega = \left\{ [e_1, g, f, e_2, e_4], [e_1, f, g, e_2, e_4], [g, e_1, f, e_2, e_4], [g, f, e_1, e_2, e_3, e_4], [f, e_1, g, e_2, e_3, e_4], [f, g, e_1, e_2, e_3, e_4], [e_1, g, f, e_2, e_3, e_4], [e_1, f, g, e_2, e_3, e_4], [g, e_1, f, e_3, e_2, e_4], [g, f, e_1, e_3, e_2, e_4], [f, e_1, g, e_3, e_2, e_4], [f, g, e_1, e_3, e_2, e_4] \right\} \]

Next, using a brief example, we demonstrate the calculation of the probability of seeing an element once in a weakly ordered ranked list.

**Example A.0.3** Probability of Seeing an Element Once By Consulting a Weakly Ordered List. Consider the system output \( R \) and the scenarios \( \Omega \) from the previous example. Let the probability of seeing content specified in component \( e_3 \) from \( e_1 \) be 0.8, from \( e_2 \) be 0.4 and from \( e_4 \) be 0.6. What is the overall probability that \( e_3 \) is never redundant in the output?

**Ans.** Applying Equation [A.1] to our example we get

\[
q(e; R) = \frac{1}{12} \cdot \left[ 6 \cdot (1 - \tilde{p}(e_3; e_1)) \cdot (1 - \tilde{p}(e_3; e_2)) + 6 \cdot (1 - \tilde{p}(e_3; e_1)) \right]
\]

\[
= 0.16.
\]

Above, we have demonstrated the relationship between calculating redundancy and the weak ordering of elements. We now consider weakly ordered sub-documents.

Let \( \bar{e} = \{ u_1, u_2, \ldots, u_\ell \} \) denote a sub-document as a set of nodes where \( u_\ell \) is an element. Consider navigation as shown in Equation [3.17] in Chapter 3. Let us replace trees with sub-documents. This is shown below in Equation [A.2]. To calculate redundancy, we replace elements in Equation [A.1] with sub-documents, as given in Equation [A.3] below. We substitute redundancy given a strict order \( q(\bar{e}, R'[\bar{e}]) \) with redundancy given user tolerance in Equation [B.3] in Appendix B. Navigation \( \tilde{p}(\bar{e}; \bar{f}) \) in Equation [A.3] is...
calculated using Equation A.2. Equation A.3 is composed of the terms $1 - \tilde{p}(\bar{e}; \bar{f})$ and $\tilde{p}(\bar{e}; \bar{f})$. We replace these with product series $Q^t$ (Equation A.5) and $P^{r-t}$ (Equation A.6), respectively. This completes our formulation of redundancy for weakly ordered sub-documents, and is shown in Equation A.4.

$$\tilde{p}(\bar{e}; \bar{f}) = \sum_{v \in \bar{f}} \sum_{u \in \bar{e}} \tilde{p}(u; v) \frac{1}{|\bar{e}| \cdot |\bar{f}|}$$ (A.2)

$$q(\bar{e}; R[\bar{e}]) = q(\bar{e}; \Omega) = \frac{1}{\ell} \cdot \sum_{R'[\bar{e}] \in \Omega} q(\bar{e}; R'[\bar{e}])$$ (A.3)

$$q(\bar{e}; R[\bar{e}]) = \sum_{t=0}^{\min(r, \tau)} \sum_{f_i \in R'[\bar{e}]} \cdots \sum_{f_t \in R'[\bar{e}]} Q^t \times P^{r-t}$$ (A.4)

$$Q^t = Q(\bar{e}, t, f_1, \ldots, f_t) = \prod_{j=1}^{t} 1 - \tilde{p}(\bar{e}; \bar{f}_j)$$ (A.5)

$$P^{r-t} = P(\bar{e}, t, R'[\bar{e}], f_1, \ldots, f_t) = \prod_{f_j \in R'[\bar{e}]} \tilde{p}(\bar{e}; \bar{f}_j) \prod_{f_j \notin \bigcup_{j=1}^{t} \bar{f}_j}$$ (A.6)

where $\bar{e} = \{u_1, u_2, \ldots, u_{\ell}\}$, and $u_i$ is an element, $|\bar{e}|$ is the size of $\bar{e}$, $R[\bar{e}]$ is a ranked list $R$ up to the rank of $\bar{e}$, $\Omega$ is the set of consultation scenarios for the weakly ordered ranked list $R[\bar{e}]$, $\ell = \prod_{i=1}^{l} l_i!$ is the size of $\Omega$, $l$ is the number of ranks in $R[\bar{e}]$, $l_i$ is the size of the $i$-th rank, $r$ is the number of dependent sub-documents to $\bar{e}$ in output $R[\bar{e}]$, the constant $\tau$ is the user tolerance (see Appendix B), $\tilde{p}(\bar{e}; \bar{f})$ is the probability that sub-document $\bar{e}$ will be seen once given that the user visited sub-document $\bar{f}$, and $R - \bar{f}$ is the ranked list $R$ minus sub-document $\bar{f}$. 

Copyright © 2023 by John Wiley & Sons, Inc.
Appendix B

Tolerance-to-Irrelevance in SR

We refer to user tolerance, based on tolerance-to-irrelevance [36], as the number of times a user sees the same content before the effort to see the content is considered wasted. In this work, we assume (unless explicitly stated) that a user tolerates seeing content once. Redundancy occurs because a user sees the same information more times than they tolerate [36]. In this section, we derive the general case of calculating redundancy in SR where the user tolerates seeing content more than once.

In this work, we assume that users do not tolerate seeing content more than once (unless explicitly stated). Let $\tau$ denote the user’s tolerance where $\tau + 1$ is the number of times a user tolerates seeing content. Thus, we usually assume $\tau = 0$.

We begin by reviewing the case where the user tolerates seeing content once ($\tau = 0$). Content that is seen within the user’s tolerance is not redundant, i.e., the user will see the content $\tau + 1$ or fewer times.

Let $R_i$ denote the ranked list $R$ up to the rank of node $e_i$. We will refer to $R_i$ as the user’s browsing history. Let $q(e; R_{i-1})$ be the probability that node $e$ is not redundant to the user given the browsing history $R_{i-1}$, i.e., it has been seen once.

Consider the ranked list of nodes $R = e_1, e_2, e_3$ where the user may navigate between them. In using this list to seek relevant information, the user will consult each node in
Appendix B. Tolerance-to-Irrelevance in SR

In descending order and thereby may see them more than once. Prior to visiting node $e_3$, the user will have a browsing history of having visited $e_1$ and $e_2$. On each visit, there was a probability that the user saw node $e_3$. The probability that node $e_2$ was not seen while navigating from node $e_1$ is $1 - \tilde{p}(e_2; e_1)$ using Equation\[3.12\] (in Section 3.3.2). The probability that the node $e_3$ was not seen while navigating from node $e_1$ is $1 - \tilde{p}(e_3; e_1)$. Similarly, the probability that node $e_3$ was not seen while navigating from node $e_2$ is $1 - \tilde{p}(e_3; e_2)$. For $e_2$ to not be seen redundantly given the browsing history $R_1 = \{e_1\}$ is $q(e_2; R_1) = 1 - \tilde{p}(e_2; e_1)$. For $e_3$ to not be seen redundantly given the browsing history $R_2 = \{e_1, e_2\}$ is $q(e_3; R_2) = (1 - \tilde{p}(e_3; e_1)) \cdot (1 - \tilde{p}(e_3; e_2))$.

The probability that node $e_i$ given browsing history $R_i$ is not seen redundantly by the user can be written, as follows:

$$q(e_i; R_{i-1}) = \prod_{f \in R_{i-1}} 1 - \tilde{p}(e_i; f) \quad (B.1)$$

where $q(e_i; R_{i-1})$ is the probability that $e_i$ is not seen given the browsing history $R_{i-1}$, $R$ is a ranked list of nodes, $R_i$ is the ranked list up to the rank of the $i$-th node, and $\tilde{p}(e; f)$ is Equation\[3.12\].

Next, we consider the case where the user tolerates seeing nodes one or more times ($\tau > 0$). We write the general probability that a node is seen less times than the user tolerates, as follows:

$$P(\text{User tolerates } e_i \vert \text{Browsing history}) = P(e_i, N \leq \tau; R_{i-1}) \quad (B.2)$$

where $P$ is the probability that a user tolerates $e_i$ as being relevant, $e_i$ is the node in which the user seeks relevant information, $N$ represents the number of times that $e_i$ has been seen, $\tau$ is the number of times that the user tolerates seeing a node, and $R_{i-1}$ is the browsing history.

Let us define the event of a user seeing a node as being either (1) retrieving the node from output, or (2) navigating to it from a node in the output. Let $e_i$ be the node at
rank \( i \) in ranked list \( R \). The probability of \( e_i \) being seen from \( \tau \) nodes in the browsing history \( R_{i-1} \) has a binomial distribution, as follows:

\[
q(e_i; R_{i-1}) = P(\text{User tolerates } | \text{Browsing history})
\]

\[
= P(e_i, N \leq \tau; R_{i-1})
\]

\[
= \sum_{t=0}^{\min(r, \tau)} \sum_{g_1 \in R_{i-1}} \cdots \sum_{g_t \in R_{i-1}} \left[ \prod_{j=1}^{t} 1 - \tilde{p}(e_i; g_j) \prod_{f \in R_{i-1}} \tilde{p}(e_i; f) \right]
\]

where \( e_i \) is a node, \( R_{i-1} \) is the browsing history of the user, \( r \) is the size of browsing history \( R_{i-1} \), the constant \( \tau \) is the user tolerance, \( f \) is a node in \( R_{i-1} \), \( g_j \) is a node in \( R_{i-1} \), \( \tilde{p}(e_i; f) \) is the probability that node \( e_i \) will be seen once given that the user visited node \( f \), and \( R - g_j \) is the browsing history \( R_{i-1} \) minus the node \( g_j \).

Equation \( \text{B.3} \) above is the general solution that we seek. If we set \( \tau = 0 \), then Equation \( \text{B.3} \) reduces to Equation \( \text{3.12} \) At \( \tau = \infty \), the user’s tolerance is infinite and the user cannot experience redundancy. Taking the limit of Equation \( \text{B.3} \) as \( \tau \to \infty \), we get \( P(e_i, N \leq \infty; R_{i-1}) = 1 \). The calculation for redundancy in cases where \( \tau \in [1, \infty) \) can be expensive. For instance, at \( \tau = 1 \), using Equation \( \text{B.3} \) we get the following:

\[
q_{\tau=1}(e_i; R_{i-1}) = \prod_{f \in R_{i-1}} \tilde{p}(e; f) + \sum_{g_1 \in R_{i-1}} \left[ (1 - \tilde{p}(e; g_1)) \prod_{f \in R_{i-1}} \tilde{p}(e; f) \right]
\]

We have included this appendix, in this work, for completeness and leave for future work the task of improving the expense of calculation of this generalization.
Appendix C

ESREval Software Package

ESREval is a Java-based lab workbench for evaluating SDR based on the methods, measures and results presented in this work. The package requires Java 1.4 or higher. It implements the streaming XML parser StAX\(^1\) to parse XML files. Persistent storage is implemented using an embedded HSQLDB\(^2\) database. The experimental results reported in this work have been run on an Intel Pentium dual-core machine running Windows XP with 4GB RAM. ESREval is used for both evaluating system performance (as described in Chapter 5) and for simulating SDR performance (as described in Chapter 6).

C.1 Using ESREval to Evaluate Performance

Evaluation of performance in ESREval is described in Chapter 5. To evaluate a system, ESREval is configured and run as follows:

1. Convert navigation models into summary graphs and property files.

2. Load navigation model(s) into ESREval database.

3. Convert relevance assessments and submission files into a modified file-offset-length

\(^1\)http://stax.codehaus.org \\
\(^2\)http://hsqldb.org
Appendix C. ESREval Software Package

(FOL) format where structural paths can be mapped to summary graphs.

4. Load relevance assessments in modified FOL format into ESREval database.

5. Set the desired ESR measures to be calculated with ESREval.

6. Calculate measures for each submission (in modified FOL format) and store results in ESREval database.

7. Output results from ESREval database using either JDBC queries, CSV files, or GNUPlot scripts.

Navigation in ESR is described in Section 3.3. We represent navigation graphs using the XML summary format output by the DescribeX package in Ali, Consens, Khatchadourian & Rizzolo [8]. A summary navigation graph can be directly generated from a collection by summarizing the collection using DescribeX. User navigation graphs are similarly represented in ESREval, but must be crafted by hand into the DescribeX summary file format. The summary partition extents are used to weight the graph edges and navigation is calculated as steady-state probabilities. These probabilities are stored in a key-value property file where each key is a regular expression (that corresponds to exactly one partition in the summary) and the value is the navigational probability. The property files (referred to as MCH files) can be auto-generated by ESREval directly from a DescribeX XML summary file.

Other experimental (and unpublished) navigation strategies include sibling navigation, hyperlink navigation, and semantic navigation. Sibling navigation uses the Gambler’s Ruin Problem [103] (p. 215) to model users who navigate between similar documents components (such as users navigating chapter-to-chapter in a book). Hyperlink navigation using a simple binomial probability to model users who traverse links between documents. Semantic navigation, which models seeking information within RDF documents, is calculated by weighting a usage summary graph for RDF data as defined
in Consens & Khatchadourian [33]. In ESREval, multiple strategies can be used if we assume that the user uses multiple independent strategies simultaneously. After a navigation model is expressed into a key-value property file, the property file is then loaded into the ESREval database.

To input assessments and submissions into ESREval, original IR files (such as INEX assessments and submissions) must be converted using ESREval into our modified file-offset-length (FOL) format that appends the structural path (corresponding to partitions in the user navigation graph) and specifying tree-based outputs over a set of adjacent lines where each lines denotes a node in an induced tree. Assessments are automatically represented as trees where assessed nodes from the same document are combined into a single tree in our ESR assessments. Submission files must be specially processed in ESR in order to represent output trees. The assessments must then be loaded into persistent storage in the ESREval database.

Measures in ESREval are represented as plain-text arithmetic expressions stored in key-value property files. These expressions are parsed and calculated using a simple open-source Java-based interpreter for parsing arithmetic expressions. These arithmetic expressions are composed of pre-defined variables that are calculated by ESREval. There are approximately 60 different pre-defined variables such as ranked list variables (e.g. rank, list position, length, number of relevant rank positions); ESREval relevance gain from hits (Equation 3.8), near-misses (Equation 3.10) and misses (Equation 3.9) where relevance value can be binary relevance, graded relevance, or relevant text length; cumulated gain from hits and near-misses; total relevance value in the collection; and many others. Each equation is specified as follows: (1) how it is calculated at a rank position for a given output, (2) whether it is calculated against a reference measure (such as precision in precision-recall), and (3) the aggregate operator (which is set to be the average score) that is used to combine evaluations across topics at either a given rank or reference

\[ \text{http://www.cs.bris.ac.uk/Teaching/Resources/COMS30122/java/calc/index.html} \]
measure cut-off (depending on whether a reference measure has been specified).

Next, we run ESREval to evaluate our set of submissions in our modified FOL format. During evaluation, each hit in the output is used to re-structure the trees in the assessments by moving retrieved nodes into a tree of hits and leaving unretrieved assessed nodes in separate trees. This is done because it is faster to compute ESR across multi-node trees as compared to singleton trees, hits in ESR are based on exact matches (as specified in Section 3.2), and this approach does not affect how navigation between trees is calculated. Our measures are calculated and the final results across rank positions (or reference measure cut-offs) are stored in persistent storage in our ESREval database. After all of the submissions have been evaluated, the overall results can be obtained by either integrating a utility application through JDBC to our HSQLDB backend, outputting results as comma-delimited files using ESREval built-in functions, or using ESREval built-in functions to output GnuPlot\textsuperscript{4} scripts.

\section*{C.2 Using ESREval to Simulate SDR Measures}

ESREval includes a simulation of SDR as described in Chapter 6. A modified FOL format is used to represent the collection (see above in Section C.1). The simulation uses a driver-based implementation, whose class is dynamically loaded by specifying it in the ESREval configuration files, to represent our desired scenarios. For each tested measure, an interface must be implemented that accepts a scenario as an argument and outputs evaluation scores for both the upper-bounded and lower-bounded measure for each rank position considered in our simulation. In this work, we have limited our results to ranked lists of length $k = 2$. However, for larger simulations, the existing coded measures would need to be refactored. ESREval randomizes the relevance values of the simulated collection across relevance value scales, calculates the averaged scores

\footnote{http://www.gnuplot.info}
across iterations, and terminates simulation after a given number of iterations has been reached. In addition, ESREval calculates our measure properties of bias (Property 6.1.4), range (Property 6.1.6), relative performance (Property 6.1.9), and overall performance (Property 6.1.12). All results are held in memory and are output to stdout at termination of the simulation.
Bibliography


[65] Gabriella Kazai, Sherezad Masood, and Mounia Lalmas. A study of the assessment of relevance for the INEX ’02 test collection. In ECIR ’04: Advances in Information


