A Robust Retrieval Engine for Proximal and Structural Search

Katsuya Masuda† **Takashi Ninomiya**†‡ **Yusuke Miyao**† **Tomoko Ohta**†‡ **Jun'ichi Tsujii**†‡
† Department of Computer Science, Graduate School of Information Science and Technology,
University of Tokyo, Hongo 7-3-1, Bunkyo-ku, Tokyo 113-0033, Japan
‡ CREST, JST (Japan Science and Technology Corporation)
Honcho 4-1-8, Kawaguchi-shi, Saitama 332-0012, Japan
{kmasuda,ninomi,yusuke,okap,tsujii}@is.s.u-tokyo.ac.jp

1 Introduction

In the text retrieval area including XML and Region Algebra, many researchers pursued models for specifying what kinds of information should appear in specified structural positions and linear positions (Chinenyanga and Kushmerick, 2001; Wolff et al., 1999; Theobald and Weilkum, 2000; Clarke et al., 1995). The models attracted many researchers because they are considered to be basic frameworks for retrieving or extracting complex information like events. However, unlike IR by keyword-based search, their models are not robust, that is, they support only exact matching of queries, while we would like to know to what degree the contents in specified structural positions are relevant to those in the query even when the structure does not exactly match the query.

This paper describes a new ranked retrieval model that enables proximal and structural search for structured texts. We extend the model proposed in Region Algebra to be robust by i) incorporating the idea of rankedness in keyword-based search, and ii) expanding queries. While in ordinary ranked retrieval models relevance measures are computed in terms of words, our model assumes that they are defined in more general structural fragments, i.e., *extents* (continuous fragments in a text) proposed in Region Algebra. We decompose queries into subqueries to allow the system not only to retrieve exactly matched extents but also to retrieve partially matched ones. Our model is robust like keyword-based search, and also enables us to specify the structural and linear positions in texts as done by Region Algebra.

The significance of this work is not in the development of a new relevance measure nor in showing superiority of structure-based search over keyword-based search, but in the proposal of a framework for integrating proximal and structural ranking models. Since the model treats all types of structures in texts, not only ordinary text structures like "title," "abstract," "authors," etc., but also semantic tags corresponding to recognized named entities or events can also be used for indexing text fragments and contribute to the relevance measure. Since extents are treated similarly to keywords in traditional models, our model will be integrated with any ranking and scalability techniques used by keyword-based models.

We have implemented the ranking model in our retrieval engine, and had preliminary experiments to evaluate our model. Unfortunately, we used a rather small corpus for the experiments. This is mainly because there is no test collection of the structured query and tag-annotated text. Instead, we used the GENIA corpus (Ohta et al., 2002) as structured texts, which was an XML document annotated with semantics tags in the filed of biomedical science. The experiments show that our model succeeded in retrieving the relevant answers that an exact-matching model fails to retrieve because of lack of robustness, and the relevant answers that a non-structured model fails because of lack of structural specification.

2 A Ranking Model for Structured Queries and Texts

This section describes the definition of the relevance between a document and a structured query represented by the region algebra. The key idea is that a structured query is decomposed into subqueries, and the relevance of the whole query is represented as a vector of relevance measures of subqueries.

The region algebra (Clarke et al., 1995) is a set of operators, which represent the relation between the *extents* (i.e. regions in texts). In this paper, we suppose the region algebra has seven operators; four containment operators ($\rhd$, $\lhd$, $\not\rhd$, $\not\lhd$) representing the containment relation between the extents, two combination operators ($\triangle$, $\bigtriangledown$) corresponding to "and" and "or" operator of the boolean model, and ordering operator ($\diamond$) representing the order of words or structures in the texts. For convenience of explanation, we represent a query as a tree structure as

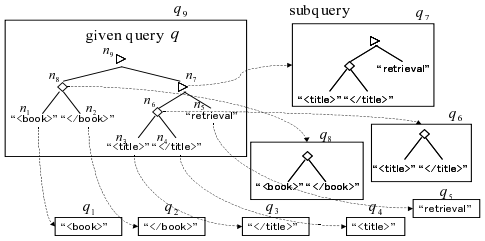

Figure 1: Subqueries of the query '[book] ▷ ([title] ▷ "retrieval")'

shown in Figure 1 [1] . This query represents 'Retrieve the books whose title has the word "retrieval." '

Our model assigns a relevance measure of the structured query as a vector of relevance measures of the subqueries. In other words, the relevance is defined by the number of portions matched with subqueries in a document. If an extent matches a subquery of query q, the extent will be somewhat relevant to q even when the extent does not exactly match q. Figure 1 shows an example of a query and its subqueries. In this example, even when an extent does not match the whole query exactly, if the extent matches "retrieval" or '[title]▷"retrieval"', the extent is considered to be relevant to the query. Subqueries are formally defined as following.

Definition 1 (Subquery) *Let q be a given query and $n_1, ..., n_m$ be the nodes of q. Subqueries $q_1, ..., q_m$ of q are the subtrees of q. Each q_i has node n_i as a root node.*

When a relevance $\sigma(q_i, d)$ between a subquery q_i and a document d is given, the relevance of the whole query is defined as following.

Definition 2 (Relevance of the whole query) *Let q be a given query, d be a document and $q_1, ..., q_m$ subqueries of q. The relevance vector $\Sigma(q, d)$ of d is defined as follows:*

$$\Sigma(q, d) = \langle \sigma(q_1, d), \sigma(q_2, d), ..., \sigma(q_m, d) \rangle$$

A relevance of a subquery should be defined similarly to that of keyword-based queries in the traditional ranked retrieval. For example, TFIDF, which is used in our experiments in Section 3, is the most simple and straightforward one, while other relevance measures recently proposed in (Robertson and Walker, 2000) can be applied. TF value is calculated using the number of extents matching the subquery, and IDF value is calculated using the number of documents including the extents matching the subquery.

While we have defined a relevance of the structured query as a vector, we need to sort the documents according to the relevance vectors. In this paper, we first map a vector into a scalar value, and then sort the documents according to this scalar measure. Three methods are introduced for the mapping from the relevance vector to the scalar measure. The first one simply works out the sum of the elements of the relevance vector.

Definition 3 (Simple Sum)

$$\rho_{sum}(q, d) = \sum_{i=1}^{m} \sigma(q_i, d)$$

The second represents the rareness of the structures. When the query is $A \triangleright B$ or $A \triangleleft B$, if the number of extents matching the query is close to the number of extents matching A, matching the query does not seem to be very important because it means that the extents that match A mostly match $A \triangleright B$ or $A \triangleleft B$. The case of the other operators is the same as with $\triangleright$ and $\triangleleft$.

Definition 4 (Structure Coefficient) *When the operator op is $\triangle$, $\bigtriangledown$ or $\diamond$, the structure coefficient of the query A op B is:*

$$sc_{AopB} = \frac{C(A) + C(B) - C(A\ op\ B)}{C(A) + C(B)}$$

and when the operator op is $\triangleright$ or $\triangleleft$, the structure coefficient of the query A op B is:

$$sc_{AopB} = \frac{C(A) - C(A\ op\ B)}{C(A)}$$

where A and B are the queries and $C(A)$ is the number of extents that match A in the document collection.

The scalar measure $\rho_{sc}(q_i, d)$ is then defined as

$$\rho_{sc}(q, d) = \sum_{i=1}^{m} sc_{q_i} \cdot \sigma(q_i, d)$$

The third is a combination of the measure of the query itself and the measure of the subqueries. Although we calculate the score of extents by subqueries instead of using only the whole query, the score of subqueries can not be compared with the score of other subqueries. We assume normalized weight of each subquery and interpolate the weight of parent node and children nodes.

Definition 5 (Interpolated Coefficient) *The interpolated coefficient of the query q_i is recursively defined as follows:*

$$\rho_{ic}(q_i, d) = \lambda \cdot \sigma(q_i, d) + (1 - \lambda) \frac{\sum_{c_i} \rho_{ic}(q_{c_i}, d)}{l}$$

where c_i is the child of node n_i, l is the number of children of node n_i, and $0 \leq \lambda \leq 1$.

This formula means that the weight of each node is defined by a weighted average of the weight of the query and its subqueries. When $\lambda = 1$, the weight of each query is normalized weight of the query. When $\lambda = 0$, the weight of each query is calculated from the weight of the subqueries, i.e. the weight is calculated by only the weight of the words used in the query.

[1] In this query, '[x]' is a syntax sugar of '⟨x⟩ ◇ ⟨/x⟩'.

1	‘([cons]▷([sem]▷“G#DNA_domain_or_region”))△(“in”◇([cons]▷([sem]▷(“G#tissue”▽“G#body_part”))))’
2	‘([event]▷([obj]▷“gene”))△(“in”◇([cons]▷([sem]▷(“G#tissue”▽“G#body_part”))))’
3	‘([event]▷([obj]◇([sem]▷“G#DNA_domain_or_region”)))△(“in”◇([cons]▷([sem]▷(“G#tissue”▽“G#body_part”))))’
4	‘([event]▷([dummy]▷“G#DNA_domain_or_region”))△(“in”◇([cons]▷([sem]▷(“G#tissue”▽“G#body_part”))))’

Table 1: Queries submitted in the experiments

3 Experiments

In this section, we show the results of our preliminary experiments of text retrieval using our model. Because there is no test collection of the structured query and tag-annotated text, we used the GENIA corpus (Ohta et al., 2002) as a structured text, which was an XML document composed of paper abstracts in the field of biomedical science. The corpus consisted of 1,990 articles, 873,087 words (including tags), and 16,391 sentences.

We compared three retrieval models, i) our model, ii) exact matching of the region algebra (*exact*), and iii) not-structured *flat* model. In the *flat* model, the query was submitted as a query composed of the words in the queries in Table 1 connected by the “and” operator (△).

The queries submitted to our system are shown in Table 1, and the document was “sentence” represented by “⟨sentence⟩” tags. Query 1, 2, and 3 are real queries made by an expert in the field of biomedicine. Query 4 is a toy query made by us to see the robustness compared with the *exact* model easily. The system output the ten results that had the highest relevance for each model[2].

Table 2 shows the number of the results that were judged relevant in the top ten results when the ranking was done using ρ_{sum}. The results show that our model was superior to the *exact* and *flat* models for Query 1, 2, and 3. Compared to the *exact* model, our model output more relevant documents, since our model allows the partial matching of the query, which shows the robustness of our model. In addition, our model outperforms the *flat* model, which means that the structural specification of the query was effective for finding the relevant documents. For Query 4, our model succeeded in finding the relevant results although the *exact* model failed to find results because Query 4 includes the tag not contained in the text (“⟨dummy⟩” tag). This result shows the robustness of our model.

Although we omit the results of using ρ_{sc} and ρ_{ic} because of the limit of the space, here we summarize the results of them. The number of relevant results using ρ_{sc} was the same as that of ρ_{sum}, but the rank of irrelevant results using ρ_{sc} was lower than that of ρ_{sum}. The results using ρ_{ic} varied between the results of the *flat* model and the results of ρ_{sum} depending on the value of λ.

[2]For the *exact* model, ten results were selected randomly from the exactly matched results if the total number of results was more than ten. After we had the results for each model, we shuffled these results randomly for each query, and the shuffled results were judged by an expert in the field of biomedicine whether they were relevant or not.

Query	*our model*	*exact*	*flat*
1	10/10	9/10	9/10
2	6/10	5/ 5	3/10
3	10/10	9/ 9	8/10
4	7/10	0/ 0	9/10

Table 2: (The number of relevant results) / (the number of all results) in top 10 results.

4 Conclusions

We proposed a ranked retrieval model for structured queries and texts by extending the region algebra to be ranked. Our model achieved robustness by extending the concept of words to extents and by matching with subqueries decomposed from a given query instead of matching the entire query or words.

References

T. Chinenyanga and N. Kushmerick. 2001. Expressive and efficient ranked querying of XML data. In *Proceedings of WebDB-2001*.

C. L. A. Clarke, G. V. Cormack, and F. J. Burkowski. 1995. An algebra for structured text search and a framework for its implementation. *The computer Journal*, 38(1):43–56.

T. Ohta, Y. Tateisi, H. Mima, and J. Tsujii. 2002. GENIA corpus: an annotated research abstract corpus in molecular biology domain. In *Proceedings of HLT 2002*.

S. E. Robertson and S. Walker. 2000. Okapi/Keenbow at TREC-8. In *TREC-8*, pages 151–161.

A. Theobald and G. Weilkum. 2000. Adding relevance to XML. In *Proceedings of WebDB'00*.

J. Wolff, H. Flörke, and A. Cremers. 1999. XPRES: a Ranking Approach to Retrieval on Structured Documents. Technical Report IAI-TR-99-12, University of Bonn.