Information Retrieval from Annotated Texts

Aviezri S. Fraenkel
Dept. of Applied Math. & CS
The Weizmann Institute of Science
Rehovot 76100, Israel
fraenkel@wisdom.weizmann.ac.il

Shmuel T. Klein
Dept. of Math. & CS
Bar Ilan University
Ramat-Gan 52900, Israel
tomi@bimacs.cs.biu.ac.il

http://www.wisdom.weizmann.ac.il/~fraenkel/fraenkel.html

September 8, 1998

Abstract

Methods for the correct and efficient handling of annotations in a full-text retrieval system are investigated. The problem with annotations is that they cannot be treated as regular text, since this would disrupt proximity searches, but on the other hand, they cannot be ignored, as they may carry important information. Moreover, in some cases, a user may wish to restrict a search to prespecified subsets of annotations. We suggest a new way of processing the database to overcome the above dilemma.

Keywords: Full-text information retrieval systems, annotations, access methods, inverted files, concordances, proximity searches.

Information Retrieval from Annotated Texts

Abstract: Methods for the correct and efficient handling of annotations in a full-text retrieval system are investigated. The problem with annotations is that they cannot be treated as regular text, since this would disrupt proximity searches, but on the other hand, they cannot be ignored, as they may carry important information. Moreover, in some cases, a user may wish to restrict a search to prespecified subsets of annotations. We suggest a new way of processing the database to overcome the above dilemma.

1. Introduction and Background

We consider a full text retrieval system based on a large data textual corpus written in some natural language. The special case investigated here does not restrict the text to consist of a linear sequence of words, but allows for what we shall call below layers of *annotations*: each annotation is an additional text block of varying size (from a few words to several pages), attached to a main text portion either by the author or by an independent editor, generally to elucidate some point in the main text. These annotations may consist of:

- short references to other texts, or to other passages in the same text, which are often included in the main text and labeled as an annotation either by special font or typesize or some form of brackets, e.g., cf. Hamlet, Scene II or *see above, page 15*;
- various types of footnotes; these tend to be short in general, but depending on the type of literature, their total amount may sometimes even exceed that of the original text, e.g., in many law related publications;

- commentaries to classical or widely accepted texts, such as *Bible* commentaries or annotations to *Alice in Wonderland*.

Note that with each word of the main text, several different annotations[1*] may be associated.

* or comments added in the margins by persons claiming that the margins are too small to accommodate their proofs

Our work has applications to a *hypertext* environment [13], in which the database consists of a large collection of texts that frequently refer to each other by means of a set of *hyperlinks*. The problem of retrieving information in a Hypertext environment has been studied by several authors. Guinan and Smeaton [11] suggest the usage of dynamically planned guided tours. Frei and Stieger [9] distinguish between referential hyperlinks (like the first example above) and semantic links (as in the second and third examples). They concentrate on the latter, and consider free text annotations to be part of the semantic link. Savoy [17] suggests an extended vector processing scheme to improve retrieval performance by extracting additional information from hyperlinks. Frisse [10] uses statistical information retrieval techniques to find good starting points in a browsing process. Croft and Turtle [4] show how to incorporate hyperlinks into a probabilistic retrieval model based on inference nets.

On the one hand, our approach is different from the usual hypertext approach [1, 5] and from the above in that we consider a main text, and look upon the annotations as having a somewhat different status, whereas the texts linked together in a hypertext system are judged to be of similar a priori importance. We also do not consider the problem of information retrieval in its broadest sense, but concentrate on the technical aspect of locating text passages which satisfy certain constraints expressed by a user query. On the other hand, we consider the hyperlinks *themselves* to be annotations, which turns them into searchable text, contrarily to the usual hypertext approach.

To give an example, suppose that we look for mathematical papers on *surgery* (in topology, manifold structures, differential geometry, homotopy, etc.). On March 17, 1995, we ran the search `surgery` on the WWW using the WebCrawler search engine at URL:

`http://webcrawler.cs.washington.edu/WebQuery.html`

[1] such as various comments written as footnotes...

which returned 374 WWW sites. As we might imagine, the bulk of them was on clinical surgery. Buried among them was `John Roe's Home Page` at URL:

`http://www.jesus.ox.ac.uk:80/~jroe/`

with information on a graduate course entitled *Surgery for amateurs* (and its links with *coarse geometry* and the classification of manifold structures). Though this was not exactly what we were looking for, it was a good lead to papers on this topic.

If the WWW would have searchable links, we could have constrained the search by the condition that `surgery` and (a link containing the word `math` or a link to `Mathematical Reviews`) appear at a distance of no more than one sentence apart. Such links do not appear in the text, only in the *html* background language. Presumably, such a modified search would lead us immediately to abstracts of papers on the requested subject, with much improved *recall* and *precision*.

It is true that a search for `surgery` AND `math` returned only 43 WWW sites. But also here the bulk was about mathematicians employed in departments of surgery; schools of medicine with mathematics departments; and universities having both schools of medicine and mathematics. The search engines on the WWW do not, at present, use metrical operators other than "document", to the best of our knowledge. We suggest that some of the robots prowling the Web might do well to add at least the sentence number in the document of the keyword they have hunted up, thus enabling some nontrivial proximity searches. This would also permit to display a KWIC (KeyWord-In-Context) of the returned documents rather than merely the documents' beginnings, as usually done now on the Web, thus helping one to decide whether clicking onto any given document is likely or not to return useful information.

For permitting the addition of links as components in proximity searches, we consider a hyperlink to be just another annotation. The main thrust of this paper is to solve the technical problem of locating information in an annotated text-environment; in particular, to enable running a proximity search where some of the words might be in the main text and others in an annotation. Thus, by reducing the hyperlink problem to that of properly handling selected subsets of annotations, any solution to the latter problem also solves the former. This would be useful, e.g., in *browsing* a main text with many annotations,

all of which are hidden from view in order not to obstruct the main text; but each can be selected by clicking on its link. We therefore have to address the problem of efficient information retrieval in an annotated text environment.

The motivation for this work stems from the *Responsa Retrieval Project* [8] (RRP), which consists of a corpus including, among others texts, the Bible, the Talmud, Maimonides' Code and other statutory law, and hundreds of additional Hebrew and Aramaic texts, mainly "case-law", written over the last seventeen centuries all over the world; and of sophisticated tools for searching in this full-text retrieval system. It is one of the few surviving original storage and retrieval systems in law which sprang up in the 1960's in the US and elsewhere. It is widely used, mainly by legal and Judaica experts. An extensive net of citations, references and cross-references is interwoven into most of this corpus, and there are many layers of citations. The utility of the system would be greatly enhanced if the citations would be transformed into hyperlinks, and if searches could be run on arbitrary preselected subsets of the main text and annotations. It would permit viewing the entire development of a subject, from the Bible until modern times, and trace its history and meaning.

The following example, run on the Hebrew Bible in the RRP corpus, shows the usefulness of supporting queries on both the main text and annotations. Suppose we seek information related to *war in the land of Babylon*. A natural query would thus be: `war` (1,10) `Babylon`, which means, as will be explained in more detail in the following section, that we are looking for an appearance of the word `war` followed by the word `Babylon` in the same sentence, up to ten words after the word `war`. When this query is run on the Bible without annotations, only one location is retrieved: "`And all the men of might` ... `that were strong and apt for war, even them the king of Babylon brought captive to Babylon`" (Kings 2, 24:16). It turns out that this verse is not relevant to the search topic, as the term `war` is here a part of the expression `apt for war`, describing the captives from a war instigated by Babylon, which, however, took place outside the land of Babylon, and is therefore not relevant to our search topic. But if the search is conducted on the Bible together with its annotations, the following verse is retrieved, in addition to the one mentioned above: "`A sound of war is in the land, and of great destruction`" (Jeremiah 50:22). The verse is chosen because there is an anno-

tation to the word `land`, written by David Altschuler (18th Century Bible exegete from Jaworow, Galicia), who wrote a book of commentaries to the Bible known as *Metzudat David*: "`in the land` — `is heard in the land of Babylon`". It would have been hard to get to this verse without the help of the annotation, since in the Bible, the term `land`, without further specification, generally refers to the land of Israel or Canaan, and other lands are mentioned explicitly as land of Egypt, land of Babylon, etc.. Altschuler's commentary in this special case is based on the context: the whole Chapter 50 of the book of Jeremiah is a prophecy about Babylon, a name which is therefore not explicitly mentioned in every verse. For this simple example, the extension of the search engine to dealing also with annotations has improved precision from 0 to 0.5, and recall from 0 to $1/r$, where $r \geq 1$ is the number of locations which are relevant to the given topic.

In the last example, we consider the Talmud. Its text has been compiled in the 6th Century and is mainly written in Aramaic. As the production of books was expensive, the text is extremely condensed and often almost impossible to understand without the help of the many layers of commentaries which have been added over the centuries. The most famous of these commentaries was written by Rabbi Shlomo Yitzhaki (died 1105 in France), widely known by his acronym *Rashi*. The main text and Rashi's commentary are tightly interwoven, the commentary adding one or more words to clarify the elliptical and often cryptic talmudic formulations. There are therefore many instances for which, at least from the point of view of the contemporary reader, the train of thought may well pass from the text to the commentary (annotation), and back to the text in a natural way. It thus makes sense to search for the occurrence of a term in the main text, following at a certain distance a term previously encountered in the commentary (annotation). Moreover, restricting a search to the main text only, which is printed apart, may result in poorer performance, both for recall and precision.

Suppose we are interested in passages relating to the topic of the age at which children should be sent to "children's teachers". (The terminology *elementary school* or *grammar school* is post-talmudic, though equivalents of *high-school, college, academy* appear in the Talmud.) A natural query would involve the expression `children's teacher(s)` to be found at some small distance from an occurrence of one of the numerals representing age: `two` or `three` or `four`, etc. When this query is run on the Talmud without

annotations, a single passage is retrieved, in Tractate *Baba Batra* of the Babylonian Talmud (page 21a): "...`he decreed that they would engage children's teachers in each province and town and bring [the children to them] at the age of about six or seven`...". This passage is clearly relevant, but it is not the only one. If the search is performed on the annotated text, an additional passage is found, in Tractate *Ktubot* (page 50a): "...`said to Rabbi Shmuel bar Shilat` — *Rashi: he was a children's teacher* — `[children who are] less than six [years old] are not to be accepted`..." With the embedded text, the passage would be retrieved, even though the first expression (`children's teacher`) is found in the annotation, and the second (the number `six`) in the main text, following the annotation. In this example, precision remains at level 1.0, but recall is improved.

In this paper we discuss the theoretical aspects, and develop the logical framework for the inclusion of annotations into the search process. In the next section, the general assumptions about the retrieval system we consider are formally defined. In Section 3, we suggest some ways to deal with the problem and discuss solutions and their implementation details.

2. Definition of the environment

The main problem caused by the presence of annotations is that they disrupt the flow of the main text. The designer of an IR system is then faced with the following dilemma. One cannot just ignore the annotations, since they may be a legitimate way chosen by the author to convey information, or, in the case the annotation has been added by somebody else, they may often add clarity or help the reader in some other way. But treating the annotations as regular text may confound proximity searches, as in the following example:

...`one of the well-known tautologies` $(2b \vee \neg(2b))$ `is due to Shakespeare`...

with the part in parentheses being an annotation; a search for `Shakespeare` and `tautologies` within a distance of up to five words would fail, even though the main text would qualify for it.

The question of how to treat annotations is independent of the method used to implement the search engine of the information retrieval system. The suggested solutions will, however, depend on these implementation details. We do not consider here retrieval systems based on the vector space model, in which query and document terms are assigned weights, and the output consists of a list of documents ranked by decreasing values of some similarity measure [15], yet even in such models, ranking by proximity of search terms may improve the quality of the results. In such vector models, the inferior status of annotated terms can be simulated by assigning them lower weights, though this won't do when the user is interested in searching the annotations rather than the main text itself. We restrict our attention to retrieval in full-text systems, in which those text passages are sought that precisely match the requests of a user query.

More formally, a query consists of an optional level-indicator, m keywords and $m-1$ binary distance constraints, as in

$$level:\ A_1\ (l_1, u_1)\ A_2\ (l_2, u_2)\ \cdots\ A_{m-1}\ (l_{m-1}, u_{m-1})\ A_m. \tag{1}$$

This is a conjunctive query, requiring all the keywords A_i to occur within the given metrical constraints specified by l_i, u_i, which are (positive or negative) integers satisfying $l_i \leq u_i$ for $1 \leq i < m$, with the couple (l_i, u_i) imposing a lower and upper limit on the distance from A_i to A_{i+1}. Negative distance means that A_{i+1} may appear before A_i in the text. The distance is measured in words (the default value), sentences or paragraphs, as prescribed by the level-indicator.

In a more general setting, one could also consider extended queries, consisting of several disjuncts, each having a form similar to (1). The requested set of locations to be retrieved is then simply the union of the sets of locations to be retrieved for each of the disjuncts. We may therefore restrict our attention to queries of the form (1).

In its simplest form, the keyword A_i is a single word or a (usually very small) set of words given explicitly by the user. In more complex cases a keyword A_i will represent a set of words $A_i = \bigcup_{j=1}^{n_i} A_{ij}$, all of which are considered synonymous in the context of the given query. For example, a variable-length-don't-care-character $*$ can be used, which stands for an arbitrary, possibly empty, string. This allows the use of prefix, suffix and infix truncation in the query. Thus A_i could be `comput*`, representing, among others, the

words `computer`, `computing`, `computerize`, etc.; or it could be `*mycin`, which retrieves a large class of antibiotics; infix truncation also can be useful for spelling foreign names, such as `Ba*tyar`, where $*$ could be matched by `h`, `k`, `kh`, `ch`, `sh`, `sch`, etc.

Another possibility for getting the variants of a keyword is from the use of a thesaurus (`month` representing `January`, `February`, etc.), or from some morphological processing (`do` representing `does`, `did`, `done`, etc.).

For every word W, let $\mathcal{C}(W)$ be the ordered list of the coordinates of all its occurrences in the text. The problem of processing a query (1) consists then, in its most general form, of finding all the m-tuples $(a_1, \ldots, a_m)$ of coordinates satisfying

$$\forall i \in \{1, \ldots, m\} \quad \exists j \in \{1, \ldots, n_i\} \qquad \text{with} \quad a_i \in \mathcal{C}(A_{ij})$$

and

$$l_i \leq d(a_i, a_{i+1}) \leq u_i \qquad \text{for } 1 \leq i < m,$$

where $d(x, y)$ denotes the distance from x to y on the given level. Every m-tuple satisfying these constraints is called a *solution* [3].

One approach to full-text retrieval, applicable mainly to rarely updated databases, is to use *inverted files*, that is, auxiliary files such as a *dictionary* and a *concordance* [16]. As used here, a dictionary is an easily searchable list of all the different words occurring in the text and usually contains for each word some statistical information such as the total number of times it occurs and the number of documents in which it appears, as well as a pointer into the concordance. A concordance contains for every word, W, the complete list of locations in the text where W appears. Depending on the underlying hierarchical structure of the text, these references may take various forms, for example, each reference could be represented by the 5-tuple (a, d, p, s, w), called a *coordinate*, where a is the author, d is the document number, p is the paragraph number (in the document), s is the sentence number (in the paragraph) and w is the word number (in the sentence); or, alternatively, a coordinate could consist of book number, page, line; or simply, when any other structure is lacking, the number of the physical block containing W and the offset within the block. The main key in the Responsa database is the *author*, since, like in a set of collected works, all the documents of a single author appear in one or more

consecutive volumes, and it is desirable to retain this property also in the computerized system. This structure has the advantage of easily supporting searches restricted to given authors.

The order of the coordinates in the concordance is induced by the order of the terms in the dictionary. Internally, for a given word, its references are lexicographically ordered according to the fields of the coordinate. Depending on the database, values in the field corresponding to the highest level (the level of authors in the first mentioned hierarchy above) are generally listed in chronological order. The retrieval process then consists of accessing the concordance for each keyword and collating the corresponding lists of coordinates.

Another approach to full-text retrieval, most suitable for medium size databases, is first to effectively reduce the size of the database by removing from consideration segments that cannot possibly satisfy the request, then to use pattern matching techniques to process the query, but only over the—hopefully small—remaining part of the database. The filtering process which reduces the amount of text to be scanned is based on assigning *signatures* to text fragments and to individual words. Signature schemes have been used in a variety of ways in information retrieval, in particular as an access method for text [6, 7, 2, 14].

Yet another approach to full-text retrieval, applicable mainly to dynamically changing databases, is to permit searches of the form (1) without constructing a concordance a priori. Rather, a superfast search engine scans the entire database sequentially, counting the word number in the current sentence, the sentence number in the current paragraph, etc., constructing a temporary concordance for each document on the fly.

In either case we see that searches of the form (1) are supported. In the sequel we thus concentrate on the inverted file method, but all the discussions apply to the three retrieval methods.

In the presence of annotations, there are three main possibilities:

1. *Treat annotations as non-retrievable text.* This is the easiest solution from the point of view of the overhead, and this is the standard way of treating the links in hypertext. Of course, important information may be lost, so that recall and/or

precision are often reduced. On the other hand, the annotations are only ignored for retrieval purposes, that is, they are not pointed to from the concordance, but they are not discarded on presentation if they happen to appear in proximity of text to be displayed.

2. *Embed annotations into the regular text.* This is the easiest solution from the point of view of the processing, and it is the standard way of treating short remarks in parentheses. This may cause retrieval errors, due to the disruption of the main text, and may thus lower both recall and precision.

3. *Treat annotations as special text.* This is the approach advocated in this paper, though it is more expensive both in terms of additional storage and of processing time. It permits to include or exclude, for any search, prespecified subfamilies of annotations — which is not possible in the previous two alternatives.

We are thus looking for data structures and methods that will permit standard full-text retrieval as explained above, regardless of the presence or absence of annotations in the text. In particular: proximity searches should be processed correctly, i.e., the numbering of the words, sentences and paragraphs of the main text should not be altered, even if an annotation appears close to the required keywords; the search for solutions to a given query should, however, also include the text of any prespecified subset of annotations, possibly excluding the main text itself; finally, annotations should be considered as alternative, yet legitimate, extensions of the main text, so that under conditions specified below, text portions should be retrieved even if one keyword appears in the main text, and another keyword appears in the annotation, within the required range.

3. Methods of embedding annotations

The algorithmic solutions to the first two possibilities above are straightforward. If annotations are considered non-retrievable, they are just ignored when inverting the files. If they are embedded in the regular text, their coordinates appear in the concordance just as do those of the main text. The problem with the third possibility, discussed below, is that

for each coordinate, additional information must be kept to indicate if the corresponding term belongs to an annotation or not.

3.1 Theoretical aspects

To simplify the following discussion, we restrict our examples to the *words* level-indicator, i.e., we constrain all the keywords to lie in a single sentence (see (1)). But the discussion applies similarly to any other level-indicator. Let us first investigate how the coordinates should be numbered. Consider, for example, the sentence

$$a_1 \quad a_2 \quad a_3 \quad b_1 \quad b_2 \quad b_3 \quad b_4 \quad a_4 \quad a_5 \quad a_6,$$

where a_i are considered words of the main text, and b_i words in annotations.[2] Since the presence of the annotation should not disrupt the numbering of the words in the main text, the internal number (within the sentence) of a_i should be i. But the word b_1 is, in a certain sense, also the fourth word in the sentence, the same as a_4. It is thus not enough to have a separate numbering for annotation words; rather, their coordinates must somehow refer to the word in the main text to which they are annotated. A possible solution is to define the internal numbering of coordinates of annotating words as continuing the sequence of the numbering of the main text, which, for the above example, would mean that the internal numbering of the words b_i should be $i + 3$. Therefore, some indicator flag is needed to differentiate between terms with identical internal number.[4]

A one-bit flag would be too restrictive in this case. First, there might be more than one kind of annotation, e.g., comments written by various editors, and several annotations may well apply to a single location in the main text, as in

$$a_1 \quad a_2 \quad a_3 \quad b_1 \quad b_2 \quad c_1 \quad c_2 \quad a_4 \quad a_5 \quad a_6, \tag{2}$$

where both b_1 b_2 and c_1 c_2 are annotations to the main text at a_3. Here, b_1, c_1 and a_4 should all have index 4. This could still be dealt with by considering the union of all the

[2] In this preliminary investigation, nested annotations[3] are not considered.

[3] i.e., annotations to annotations

[4] The question of semantic and metrical relationships between b_4 and a_4 will be discussed below.

annotations to the same point in the main text as constituting a single annotation. But this would be against our philosophy of letting the user decide which kinds of annotations he wants to search, and which he wishes to disregard.

Secondly, even if all the annotations are of the same kind and point to different locations, they might be long enough to have the numbering of their words overlap. Consider, for example, the following sentence:

$$a_1 \quad a_2 \quad a_3 \quad b_1 \quad \ldots \quad b_{13} \quad a_4 \quad a_5 \quad \ldots \quad a_{10} \quad c_1 \quad \ldots \quad c_{10} \quad a_{11} \quad a_{12} \quad \ldots \quad a_{16},$$

where b_i and c_i are annotation words. In this example, the indices of the words b_{7+j}, c_j and a_{10+j} would all be identical to $10+j$, for $1 \leq j \leq 6$, so a one-bit flag would not suffice to resolve this ambiguity.

A possible solution would thus be to define an extended coordinate, including also a *tag* field indicating whether the coordinate points to a word in the main text or to an annotation, and in the latter case, giving also information about the type of annotation and its index. In the construction phase of the concordance, all the annotations are first grouped together by type (e.g., all footnotes, then all the comments by editor A, then all comments by editor B, etc.), and are then numbered sequentially. A small table T can be used to identify annotations of a certain type with the corresponding range in the enumeration.

The general form of the extended coordinate could thus be (a, d, p, s, w, tag), with $tag = 0$ for words of the main text[5]. For the example in (2), the (w, tag) part of the coordinates of the words from left to right would be: $(1,0)$, $(2,0)$, $(3,0)$, $(4,1)$, $(5,1)$, $(4,2)$, $(5,2)$, $(4,0)$, $(5,0)$, $(6,0)$. The tag easily permits restricting the search to the main text only or to annotations only.

Adding the tag, however, solves only part of the problem. Consider for example again the sentence in (2), and suppose the query to be processed is $c_2 \ \ (1,1) \ \ a_4$. As a matter of fact, the words c_2 and a_4 are adjacent if annotations are embedded in the text, and

[5]The reason for appending the tag-field at the right, rather than at the left end of the coordinate is that the concordance is ordered lexicographically: having the tag at the left end would, for each word, group all its occurrences in annotations at the end of the list; with the tag at the right end, coordinates of annotation words appear close to coordinates of the words they might annotate.

thus the sentence should be retrieved under possibility 2. above. However, the relative index of c_2 is 5, and that of a_4 is 4, so that this sentence does not qualify for retrieval. Indeed, c_2 appears at the end of an annotation to word a_3, and a_4 just happens to follow c_2 of the embedded annotation. It may therefore be quite possible that there is no real connection between c_2 and a_4. This is just a typical example of the asymmetric aspect of the problem: annotations are text portions that are *appended to* some word in the text, and the annotation itself or some pointer to it generally appears immediately *following* the annotated word, rather than *preceding* it. There is thus a much stronger affinity between an annotated word in the main text and the first words of the annotation, than between the last words of the annotation and the following words of the main text, especially if the annotation is sufficiently long.

The real question is therefore: How should we measure the distance between a word in an embedded annotation and a word that occurs later in the main text? Refer again to the example in (2) and consider the distance from b_2 to a_4. The following three possibilities could be considered:

(a) The different annotations to a single word w are not ordered, but embedded in the text so as to form a partially ordered set. In this case, each annotation to w is individually embedded in the main text at w. The distance between any two words in two distinct annotations to w is defined as ∞. The distance between b_2 and a_4, however, is 1, which is also the distance between c_2 and a_4; the distance between b_1 (or c_1) and a_4 is 2.

(b) After embedding the annotations as in **(a)**, the distance is computed by referring to a common anchor in the main text, say a_3: b_2 being at distance 2 from a_3 and a_4 at distance 1, this would imply a distance of $d(b_2, a_4) = d(a_3, a_4) - d(a_3, b_2) = -1$ from b_2 to a_4. Note that this distance is independent of the choice of anchor.

(c) The same as **(a)**, except that if the annotation is "long", then the distance between a word in the annotation at w and a word in the main text following w is defined as ∞. Note, however, that the distance between a word preceding w and a word in the annotation at w is defined as in **(a)**.

The first and second alternatives are symmetric in the sense that for **(a)**, the distance from b_2 to a_4 depends on the length of the annotation, i.e., on the number of annotation words following b_2, but not on the number of words preceding b_2 in the same annotation, and conversely for **(b)**. This seems to make **(b)** a less attractive alternative: we measure distance "backwards" in the annotation and "forwards" in the main text.

Alternative **(c)** is reasonable for long annotations, where there seems to be little semantic correlation, if any, between the tail end of the embedded annotation and the main text following it. But for short annotations, such as hyperlinks, possibility **(a)**—which is included in **(c)**—seems reasonable. One tends to consider short annotations, such as pointers or short footnotes, differently from long annotations, which may at times extend to several paragraphs. It is true that this notion of how to define a short or long annotation is a subjective one, but for our application, a subjective judgment is crucial, since it governs also the standard measures recall and precision. Both are defined in terms of the number of "relevant" items, and this relevance is assessed by means of the user's intuition. We should thus decide how to let the length of the annotation influence the retrieval process.

Fixing a global constant threshold seems, at first, not to be reasonable. Depending on language and context, the number of consecutive words that still are semantically related may greatly vary. An alternative would be to define a "short" annotation as one not extending beyond a single sentence. In that case, words belonging to a second or later sentence of a long annotation are very rarely connected to words in the main text following the annotation. There remains, however, the problem of defining what exactly should be considered as a sentence. Just basing this definition on the occurrence of certain punctuation signs may be misleading: certain authors, especially modern poets, use periods and commas very scarcely, if at all; on the other hand, not every dot ends a sentence, as can be seen by the hyperlinks displayed in the introduction. As another example, consider the words `Hello! I said to Mr. Smith.` This would be parsed as three instead of only one sentence, and even the more sophisticated rule of defining the end of a sentence by the appearance of a period (or exclamation point, etc) followed by a space and a capitalized letter, would fail in this case. Such problems in the definition of a sentence were reported for the *Trésor de la Langue Française* [12], a large French

database, written entirely in upper case.

Thus we revert back to fixing a small, rather arbitrary limit to the length of what we shall consider a short annotation, say, 20 words. Punctuation signs within these 20 words are ignored. It is of course easy to come up with examples for which this rule of thumb will fail, but in many cases it will help not to overlook relevant occurrences, and thereby improve recall, but still filter out some irrelevant ones, and thus improve precision as well.

3.2 Implementation issues

From the implementation point of view, possibility **(b)** is the easiest to implement, since the coordinate including the tag-field provides all the necessary information, and the distance can be computed by simple subtraction of the values in the word-fields of the coordinates. However, a word in an annotation at w has a negative distance to the first few words in the main text following w. This distance becomes positive only for words further away from w down the main text. This seems counter-intuitive, and therefore we abandon possibility **(b)**.

As to **(a)** and **(c)**, we need, in addition to the tag in each coordinate, also knowledge about the length of each of the annotations, and their starting points. It's not enough to know the length without the start; in the sentence:

$$a_1 \quad a_2 \quad a_3 \quad a_4 \quad b_2 \quad b_1 \quad c_1 \quad c_2 \quad a_5 \quad a_6,$$

the lengths of the b and c annotations are identical to those of (2), and also the coordinates of a_4 and b_2, 4 and 5 respectively, are the same in both. And yet b_2 precedes a_4 in (2), and vice versa here.

For possibilities **(a)** and **(c)**, the coordinates of words in annotations have thus to be further extended to the form $(a, d, p, s, w, tag, strt, len)$, with $strt$ being the index of the annotated word in the main text, and len the length (in words) of the annotation the current coordinate belongs to. Since generally, the total amount of annotations is small relative to the main text, having additional fields in the corresponding coordinates is not a considerable overhead.

Consider the query $a_3 \ \ (1,3) \ \ b_3$ applied to a text including the sentence

$$a_1 \quad a_2 \quad a_3 \quad b_1 \quad b_2 \quad b_3 \quad \cdots \quad b_8 \quad b_9 = a_3 \quad b_{10} \quad a_4 \quad a_5 = b_3 \quad a_6 \quad a_7$$

where the words $b_1 \cdots b_{10}$ constitute an annotation and b_9 and a_3 are occurrences of the same word, and similarly for a_5 and b_3. Under **(c)**, considering the annotation as a long one, the first instance, with a_3 in the main text and b_3 in the annotation, is retrieved, but the second, where a_3 is in the annotation and b_3 in the main text, is not retrieved. To distinguish between these two cases, in the implementation, we might first think that it suffices to restrict retrieval to the case when the first word (a_3) is in the main text and the second (b_3) in the annotation. However, our query is clearly equivalent to $b_3 \ \ (-3,-1) \ \ a_3$, for which we would then retrieve the second rather than the first appearance of the keyword pair.

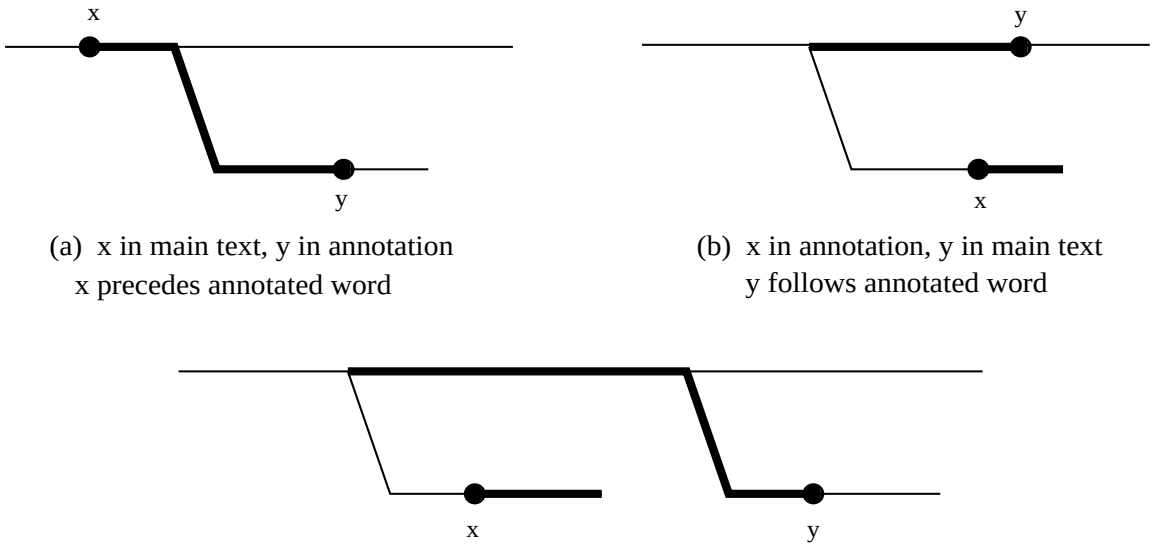
We should thus redefine the notion of the *distance* $d(x,y)$ from the word x to the word y, by means of the procedure in Figure 1. It corresponds to possibility **(a)**, which takes the embedded annotation words into account. Note, however, that the distance between two words belonging both to the main text, ignores all annotations. We assume here the default level, i.e., that the distance is measured in words. Therefore, if the *author*, *document*, *paragraph* and *sentence* fields of the given coordinates do not match, the distance between them is defined as infinity, so that in any case the constraints $l_i \leq d(x,y) \leq u_i$ will not be satisfied.

Figure 2 is a schematic representation of some of the possible cases. The horizontal line on the top of each of the drawings represents the main text, and the segments branching off represent annotations. The position of the words x and y in the text and/or annotation are represented by bullets, and the distance $d(x,y)$ corresponds, in each case, to the total length of the heavy lines. Pointers to the cases depicted in Figure 2 also appear in the algorithm of Figure 1.

Note that the definition of the distance is straightforward if both words are in the main text or in the same annotation, and also in the case where one of the words, say x, is in the main text, and the other is in an annotation to a word of the main text that follows x. In the other cases, the definition of the distance involves also the *len* field. For example, consider the case 2(b), where x is in the annotation and y is in the main text following the

```
Distance function d(x, y)
{
    if a(x) ≠ a(y) or d(x) ≠ d(y) or p(x) ≠ p(y) or s(x) ≠ s(y)
                        /* if not same author, document, paragraph and sentence */
        d(x, y) ⟵ ∞
    else
        if tag(x) = tag(y) and /* if strt field exists: */ strt(x) = strt(y)
                            /* both in main text or in same annotation */
            d(x, y) ⟵ w(y) − w(x)
        else
            if tag(x) > 0 and tag(y) = 0    /* y in main text; x in annotation */
                if strt(x) < w(y)       /* annotated word precedes y — Fig 2(b) */
                    d(x, y) ⟵ w(y) − w(x) + len(x)
                else                    /* annotated word follows y */
                    d(x, y) ⟵ w(y) − w(x)      /* which is negative */
            else if tag(y) > 0 and tag(x) = 0    /* x in main text; y in annotation */
                if strt(y) < w(x)       /* annotated word precedes x */
                    d(x, y) ⟵ w(y) − w(x) − len(y)
                else                    /* annotated word follows x — Fig 2(a) */
                    d(x, y) ⟵ w(y) − w(x)
            else            /* both x and y in annotations, but in different ones */
                if strt(x) = strt(y)    /* annotating same word */
                    d(x, y) ⟵ ∞
                else if strt(x) < strt(y)                          /* Fig 2(c) */
                    d(x, y) ⟵ w(y) − w(x) + len(x)
                else                    /* strt(x) > strt(y) */
                    d(x, y) ⟵ w(y) − w(x) − len(y)
}
```

Figure 1: *Distance evaluation algorithm for word operator under alternative* **(a)**



(a) x in main text, y in annotation
x precedes annotated word

(b) x in annotation, y in main text
y follows annotated word

(c) Both x and y in (different) annotations

Figure 2: *Schematic representation of the distance* $d(x, y)$

annotated word. The distance $d(x, y)$ consists of two parts: the distance from x to the end of the annotation, d_1, plus the distance from the annotated word to y, d_2. To evaluate d_1, note that $w(x)$ is the index of x within the sentence, so that $w(x) - strt(x)$ gives the relative index of x within the annotation; therefore $d_1 = len(x) - (w(x) - strt(x))$. The value of d_2 is clearly $w(y) - strt(x)$ and we get

$$d(x, y) = d_1 + d_2 = w(y) - w(x) + len(y).$$

The other cases are evaluated similarly.

In the special case in which both x and y are in different annotations to the same point of the main text, the distance is set to ∞, since there are two just as plausible interpretations. For instance, in (2), $d(b_1, c_2)$ could be 3 as well as -1, depending on whether we consider the annotation b_1 b_2 as preceding or as following c_1 c_2. However, if b_1 and c_2 are in annotations to different main text words w_1, w_2, then the distance between them is finite even if the query is restricted to annotations only and excludes the main text, when the main text serves only as a "catalyst".

The general retrieval procedure is given in Figure 3. For the ease of description, we assume here that the keywords are processed from left to right, from A_1 to A_m (see (1)).

Query processing

{

Partial-Solution(1) $\longleftarrow$ $\bigcup_{j=1}^{n_1} \mathcal{C}(A_{1j})$

for $i \longleftarrow 2$ **to** m

for $j \longleftarrow 1$ **to** n_i

for all $(\ldots, x) \in$ Partial-Solution$(i-1)$

for all $y \in \mathcal{C}(A_{ij})$

if $l_{i-1} \leq d(x,y) \leq u_{i-1}$

add $(\ldots, x, y)$ **to** Partial-Solution(i)

Solution $\longleftarrow$ Partial-Solution(m)

}

Figure 3: *Query processing algorithm*

In reality, the order may depend on the number of coordinates of each of the keywords, since much CPU time and many accesses to secondary storage may be saved if we start first with the rare keywords. On the other hand, it is not always possible to process the keywords in increasing order of the number of their occurrences, because of the varying metrical constraints.[6] For more details on this procedure, the reader is referred to [3].

3.3 Alternative implementation

We have so far considered the possibility of extending the coordinate structure to solve our problem of information retrieval in the presence of annotations. If most of the user queries restrict their attention to the main text, it might be more efficient to construct not only one, but two or several concordances.

[6] For example, if in the query "`A` (2,3) `B` (1,3) `C`" the keyword `A` is the one with the lowest number of occurrences, then the next one to be dealt with should be `B` even if `C` has less occurrences than `B`. If we would deal first with `C`, assuming that it must appear between 3 to 6 words after `A`, and only then turn to look for `B`, 1 to 3 words before `C`, then the following string would be retrieved: "... `x A x x x B x C x` ..."; however, it does not satisfy the query because the distance from `A` to `B` is 4. Hence the order of processing the keywords is more restricted. In fact, at any stage, one of the keywords can be chosen which is *adjacent* to one of those already handled.

There would be one concordance for the main text, and another for all the words within annotations, or, if the annotations are of several kinds, one could even consider having a separate concordance for each type of annotation. Different time/space tradeoffs are involved in this choice, and the usefulness of each of the methods may depend on the types of the most popular queries.

Having the annotations in separate concordances will reduce the necessary storage space, since no tags are needed. On the other hand, every query, unless specifically restricted to the main text alone, will now have to search (and thus access) several files. The coordinates of annotation words will still have to include also an exact reference to the annotated word, so as to permit the retrieval of text portions in which part of the keywords appear in the main text, and part in one or several annotations.

4. Conclusion

We have presented some approaches to the problem of dealing with annotations in full-text retrieval systems. The goals were: (i) to improve the retrieval performance by discovering more relevant items; and (ii) at the same time to reduce the number of non-relevant items that will be retrieved; and (iii) to permit selective retrieval, ignoring at times all the annotations, concentrating at others exclusively on certain subsets of annotations. In particular, we pointed to the possible asymmetry in the relationships between a word in an annotation and a word of the main text preceding the insertion point of the annotation on the one hand, and a word in an annotation and a word of the main text following this insertion point on the other hand. This and other metrical problems were dealt with by extending the notion of distance between words, as given in the distance evaluation algorithm.

References

[1] **Agosti M., Melucci M., Crestani F.,** Automatic authoring and construction of hypermedia for information retrieval, *Multimedia Systems* **3** (1995) 15–24.

[2] **Bookstein A., Klein S.T.,** Using Bitmaps for Medium Sized Information Retrieval Systems, *Information Processing & Management* **26** (1990) 525–533.

[3] **Choueka Y., Fraenkel A.S., Klein S.T., Segal E.**, Improved Techniques for Processing Queries in Full-Text Systems, *Proc. 10-th ACM-SIGIR Conf.,* New Orleans (1987) 306–315.

[4] **Croft W.B., Turtle H.R.,** Retrieval strategies for Hypertext, *Information Processing & Management* **29** (1993) 313–324.

[5] **DeBra P., Houben G., Kornatzky Y., Post R.,** Information retrieval in distributed Hypertexts, *Proc.* 4^{th} *RIAO–94 Conference*, New-York (1994) 481–491.

[6] **Faloutsos C.,** Signature files: Design and performance comparison of some signature extraction methods, *Proc. ACM-SIGMOD Conf.,* Austin, ACM, New York (1985) 63–82.

[7] **Faloutsos C., Christodulakis S.,** Signature files: An access method for documents and its analytical performance evaluation, *ACM Trans. on Office Inf. Systems* **2** (1984) 267–288.

[8] **Fraenkel A.S.,** All about the Responsa Retrieval Project you always wanted to know but were afraid to ask, Expanded Summary, *Jurimetrics J.* **16** (1976) 149–156.

[9] **Frei H.P., Stieger D.,** The use of semantic links in Hypertext Information Retrieval, *Information Processing & Management* **31** (1995) 1–13.

[10] **Frisse M.E.,** Searching for information in a hypertext medical handbook, *Communications of the ACM* **31** (1988) 880–886.

[11] **Guinan C., Smeaton A.F.,** Information retrieval from Hypertext using dynamically planned guided tours, *Proc. 2-nd European Conf. on Hypertext and Hypermedia*, Milano, (1992) 122-130.

[12] **Klein S.T., Bookstein A., Deerwester S.,** Storing Text Retrieval Systems on CD-ROM: Compression and Encryption Considerations, *ACM Trans. on Information Systems* **7** (1989) 230–245.

[13] **Nielsen J.,** *Hypertext and Hypermedia*, Academic Press Professional, Boston (1993).

[14] **Sacks-Davis R., Kent A., Ramamohanarao K.,** Multikey access methods based on superimposed coding techniques, *ACM Trans. on Database Systems* **12** (1987) 655–696.

[15] **Salton G,**, *Automatic Text Processing,* Addison-Wesley, Reading, MA (1989).

[16] **Salton G., McGill M.J.,** *Introduction to Modern Information Retrieval*, McGraw Hill, New York (1983).

[17] **Savoy J.,** An extended vector-processing scheme for searching information in Hypertext systems, *Information Processing & Management* **32** (1996) 155–170.