Cap. 02: Modeling (Structured Text Models)

October 4, 1999
Introduction

- Keyword-based query answering considers that the documents are flat i.e., a word in the title has the same weight as a word in the body of the document.
- But, the document structure is one additional piece of information which can be taken advantage of.
- For instance, words appearing in the title or in sub-titles within the document could receive higher weight.
Introduction

- Consider the following information need:
  - Retrieve all documents which contain a page in which the string “atomic holocaust” appears in italic in the text surrounding a Figure whose label contains the word *earth*

- The corresponding query could be:
  - `same-page( near( “atomic holocaust”, Figure( label( “earth” )))`
Introduction

- Advanced interfaces that facilitate the specification of the structure are also highly desirable.
- Models which allow combining information on text content with information on document structure are called structured text models.
- Structured text models include no ranking (open research problem).
Basic Definitions

- **Match point**: the position in the text of a sequence of words that match the query
  - Query: “atomic holocaust in Hiroshima”
  - Doc dj: contains 3 lines with this string
  - Then, doc dj contains 3 match points
- **Region**: a contiguous portion of the text
- **Node**: a structural component of the text such as a chapter, a section, etc.
Non-Overlapping Lists

- Due to Burkowski, 1992.
- Idea: divide the text in *non-overlapping* regions which are collected in a *list*
- Multiple ways to divide the text in non-overlapping parts yield multiple lists:
  - a list for chapters
  - a list for sections
  - a list for subsections
- Text regions from distinct lists might overlap
Non-Overlapping Lists

- $L_0$: Chapter
- $L_1$: Sections
- $L_2$: SubSections
- $L_3$: SubSubSections
Non-Overlapping Lists

- Implementation:
  - single inverted file that combines keywords and text regions
  - to each entry in this inverted file is associated a list of text regions
  - lists of text regions can be merged with lists of keywords
Non-Overlapping Lists

- Regions are non-overlapping which limits the queries that can be asked.
- Types of queries:
  - select a region that contains a given word
  - select a region A that does not contain a region B (regions A and B belong to distinct lists)
  - select a region not contained within any other region
Conclusions

- The non-overlapping lists model is simple and allows efficient implementation.
- But, types of queries that can be asked are limited.
- Also, model does not include any provision for ranking the documents by degree of similarity to the query.
- What does structural similarity mean?
Proximal Nodes

- Due to Navarro and Baeza-Yates, 1997
- Idea: define a strict hierarchical index over the text. This enriches the previous model that used flat lists.
- Multiple index hierarchies might be defined
- Two distinct index hierarchies might refer to text regions that overlap
Definitions

- Each indexing structure is a strict hierarchy composed of:
  - chapters
  - sections
  - subsections
  - paragraphs
  - lines
- Each of these components is called a *node*
- To each node is associated a text region
Proximal Nodes

Chapter
Sections
SubSections
SubSubSections

holocaust → 10 → 256 → ⋮ → 48,324
Proximal Nodes

- Key points:
  - In the hierarchical index, one node might be contained within another node
  - But, two nodes of a same hierarchy cannot overlap
  - The inverted list for keywords complements the hierarchical index
  - The implementation here is more complex than that for non-overlapping lists
Proximal Nodes

- Queries are now regular expressions:
  - search for strings
  - references to structural components
  - combination of these
- Model is a compromise between expressiveness and efficiency
- Queries are simple but can be processed efficiently
- Further, model is more expressive than non-overlapping lists
Proximal Nodes

- Query: find the sections, the subsections, and the subsubsections that contain the word “holocaust”
  - [(section) with ("holocaust")]
- Simple query processing:
  - traverse the inverted list for “holocaust” and determine all match points
  - use the match points to search in the hierarchical index for the structural components
Proximal Nodes

- Query: [(section) with ("holocaust")]
- Sophisticated query processing:
  - get the first entry in the inverted list for “holocaust”
  - use this match point to search in the hierarchical index for the structural components
  - Innermost matching component: smaller one
  - Check if innermost matching component includes the second entry in the inverted list for “holocaust”
  - If it does, check the third entry and so on
  - This allows matching efficiently the nearby (or proximal) nodes
Conclusions

- Model allows formulating queries that are more sophisticated than those allowed by non-overlapping lists.
- To speed up query processing, nearby nodes are inspected.
- Types of queries that can be asked are somewhat limited (all nodes in the answer must come from a same index hierarchy!)
- Model is a compromise between efficiency and expressiveness.