

# Query Languages

Berlin Chen 2004

Reference:

1. *Modern Information Retrieval*, chapter 4

# The Kinds of Queries

- Data retrieval
  - Pattern-based querying
  - Retrieve docs that contains (or exactly match) the objects that satisfy the conditions clearly specified in the query
  - A single erroneous object implies failure!
- Information retrieval
  - Keyword-based querying
  - Retrieve *relevant* docs in response to the query (the formulation of a user information need)
  - Allow the answer to be ranked

# The Kinds of Queries

- On-line databases or CD-ROM archives
  - High level software packages should be viewed as query languages
  - Named “protocols”

Different query languages are formulated and then used at different situations, by considering

- The underlying retrieval models (ranking algorithms)
- The content (**semantics**) and structure (**syntax**) of the text

Models: Boolean, vector-space, HMM ....

Formulations/word-treating machineries: stop-word list, stemming, query-expansion, ....

# The Retrieval Units

- The retrieval unit: the basic element which can be retrieved as an answer to a query
  - A set of such basic elements with ranking information
- The retrieval unit can be a file, a doc, a Web page, a paragraph, a passage, or some other structural units
- Simply referred as “docs”




# Keyword-based Querying

- Keywords
  - Those words can be used for retrieval by a query
  - A small set of words extracted from the docs
    - Preprocessing is needed
- Characteristics of keyword-based queries
  - A query composed of keywords and the docs containing such keywords are searching for
  - Intuitive, easy to express, and allowing for fast ranking
  - A query can be a single keyword, multiple keywords (**basic queries**), or more complex combination of operation involving several keywords
    - **AND, OR, BUT, ...**

# Keyword-based Querying (cont.)

- **Single-word queries**
  - **Query:** The elementary query is a word
  - **Docs:** The docs are long sequences of words
  - What is a **word** in **English** ?
    - A word is a sequence of *letters* surrounded by *separators*
    - Some characters are not letters but do not split a word, e.g. the hyphen in 'on-line'
    - Words possess **semantic/conceptual** information

# Keyword-based Querying (cont.)

- **Single-word queries** (cont.)
    - The use of word statistics for IR ranking
      - Word occurrences inside texts
        - **Term frequency** (tf): number of times a word in a doc
        - **Inverse document frequency** (IDF): number of docs in which a word appears
      - Word positions in the docs (*see next slide*)
        - May be required, e.g., a interface that **highlights each occurrence of a specific word**
- similarity between a query and doc
- 

# Keyword-based Querying (cont.)

SCAN - Speech Content Based Audio Navigator

File Search Scan

QUERY:

RESULTS - "What is the status of the trade deficit with Japan"

| RANK | PROGRAM                   | DATE  | STORY | SCORE | LENGTH | HITS |
|------|---------------------------|-------|-------|-------|--------|------|
| 1    | NPR All Things Considered | 05/31 | 3     | 15.63 | 27.65  | 6    |
| 2    | NPR All Things Considered | 05/10 | 15    | 13.89 | 512.42 | 16   |
| 3    | NPR/PRI Marketplace       | 06/14 | 4     | 13.82 | 166.40 | 14   |
| 4    | ABC World News Now        | 06/13 | 6     | 13.44 | 30.00  | 3    |
| 5    | NPR All Things Considered | 05/21 | 4     | 11.14 | 13.62  | 3    |
| 6    | NPR All Things Considered | 05/31 | 3     | 10.92 | 17.02  | 3    |
| 7    | NPR/PRI Marketplace       | 06/14 | 3     | 10.87 | 30.00  | 4    |
| 8    | CNN Headline News         | 06/07 | 18    | 9.83  | 183.55 | 6    |
| 9    | NPR/PRI Marketplace       | 06/11 | 23    | 9.82  | 203.21 | 11   |
| 10   | NPR/PRI Marketplace       | 06/14 | 6     | 9.41  | 90.33  | 4    |

OVERVIEW - NPR All Things Considered 05/10

deficit  
status  
japan  
trade

ASR TRANSCRIPTS - NPR All Things Considered 05/10

"expanding defense cooperation span is a part of our pacific democracy defense program will strengthen are lines and serve on mutual interest that while president clinton is earth credit for renewing inspecting those ties on his recent trip the administration's amateurs and in a factory posturing on trade disputes"

"buster and those ties and assess state of the president's recent attempt of damage control in nineteen ninety four that lead administration for both a trade war and lost and then declared victory even though present but received nothing the clinton a station shows funk war dead and then contradictory tactics"

"did not work for the force camp and saving deregulation competition and economic reform the result has been an increase in both the bilateral trade deficit and japanese trade nationalism the merchandise trade that has no sacred is anthony here no but i do not agree with president clinton's decision"

"the normal eyes relations with vietnam until they could could have and should receive more returned from vietnam the decision has been made the case is not closed there are many outstanding issues in our relationship with vietnam was shared economic and other enters can only be realized"

"after the outcome achieved fullest possible accounting for a missing servicemen and vietnam must understand that further progress on the field of the a. m. i. a. issue remain are biased bilateral priority now it is simply that i think we all saw to be very forthright flat out but i have fun"

"that out neo from about are commercial relations with china was incredible is right the nineteen ninety four when a funnv favored extension of most favored nation status was the best way to promote are long term interest in china"

Selection Length:  seconds

AT&T AT&T Labs Research



# Keyword-based Querying (cont.)

- **Context queries**

- Complement single-word queries with ability to search words in a given context, i.e., **near other words**
- **Words appearing near each other may signal a higher likelihood of relevance than if they appear apart**
- E.g., *Phrases of words or words are proximal in the text*

# Keyword-based Querying (cont.)

- **Context queries** (cont.)

- Two types of queries

- **Phrase**

- A sequence of single-word queries

- Q: "enhance" and "retrieval"

- D: "...enhance the retrieval..."

- Not all systems implement it!

- **Proximity**

- **A relaxed version** of the phrase query

- A sequence of single words (or phrases) is given together with a maximum allowed distance between them

- E.g., two keywords occur within four words

- Q: "enhance" and "retrieval"

- D: "...enhance the power of retrieval..."

Features:

1. Separators in the text or query may not be the same
2. uninteresting words are not considered

Features:

1. May not consider word ordering

# Keyword-based Querying (cont.)

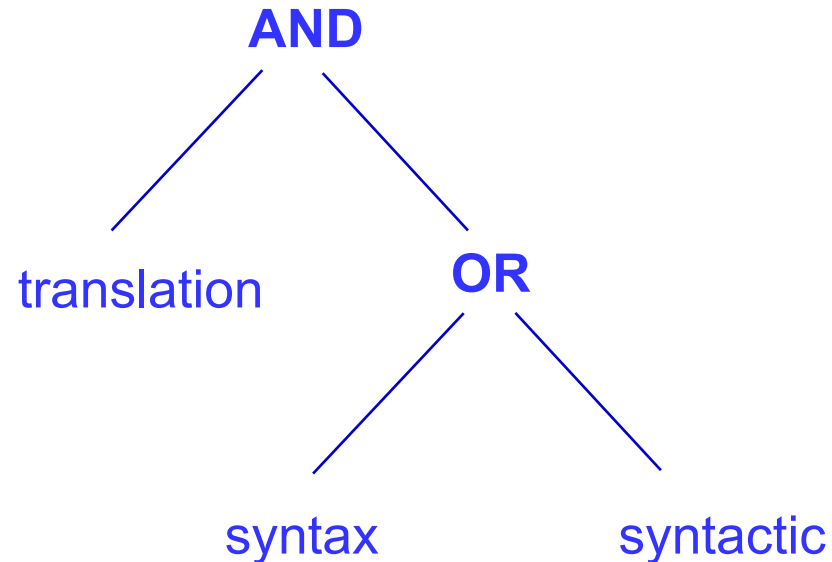
- **Context queries** (cont.)
  - Ranking
    - **Phrases**: analogous to single words
    - **Proximity queries**: the same way if physical proximity is not used as a parameter in ranking
      - Just as a hard-limiter
      - But physical proximity has semantic value !

How to do better ranking ?

# Keyword-based Querying (cont.)

- **Boolean Queries**

- Have a syntax composed of atoms (basic queries) that retrieve docs, and of **Boolean operators** which work on their operands (sets of docs)



Leaves: basic queries

Internal nodes: operators

A query syntax tree.

# Keyword-based Querying (cont.)

- **Boolean Queries** (cont.)

- Commonly used operators

- **OR**, e.g.  $(e_1 \text{ OR } e_2)$

- Select all docs which satisfy  $e_1$  or  $e_2$ . Duplicates are eliminated

$e_1$  and  $e_2$  are basic queries

| $e_1$    | $e_2$ | $e_1 \text{ OR } e_2$ | $e_1 \text{ AND } e_2$ | $e_1 \text{ BUT } e_2$ |
|----------|-------|-----------------------|------------------------|------------------------|
| $d_3$    | $d_4$ | $d_3$                 | $d_7$                  | $d_3$                  |
| $d_7$    | $d_7$ | $d_4$                 |                        | $d_{10}$               |
| $d_{10}$ | $d_8$ | $d_7$                 | $d_8$                  |                        |
|          |       | $d_{10}$              |                        |                        |

- **AND**, e.g.  $(e_1 \text{ AND } e_2)$

- Select all docs which satisfy both  $e_1$  and  $e_2$

- **BUT**, e.g.  $(e_1 \text{ BUT } e_2)$

- Select all docs which satisfy  $e_1$  but not  $e_2$

- Can use the inverted file to filter out undesired docs

No partial matching between a doc and a query  
No ranking of retrieved docs are provided!

# Keyword-based Querying (cont.)

- **Boolean Queries** (cont.)
  - **A relaxed version:** a “fuzzy Boolean” set of operators
    - The meaning of AND and OR can be relaxed
      - *all*: the AND operator
      - *one*: the OR operator (at least one)
      - *some*: retrieval elements appearing in more operands (docs) than the OR
    - Docs are ranked higher when having a larger number of elements in common with the query
- Naïve users have trouble with Boolean Queries

# Keyword-based Querying (cont.)

- **Natural language**
  - Push the fuzzy Boolean model even further
    - The distinction between AND and OR are complete blurred
  - A query can be an enumeration of words or/and context queries
  - Typically, a query treated as a bag of words (ignoring the context ) for the vector space model
    - Term-weighting, relevance feedback, etc.
  - All the documents matching a portion of the user query are retrieved
    - Docs matching more parts of the query assigned a higher ranking
  - Negation also can be handled by penalizing the ranking score
    - E.g. some words are not desired

# Keyword-based Querying (cont.)

- **Natural language**

```
? target
Input search terms separated by spaces (e.g., DOG CAT FOOD). You
can enhance your TARGET search with the following options:
- PHRASES are enclosed in single quotes
  (e.g., 'DOG FOOD')
- SYNONYMS are enclosed in parentheses
  (e.g., (DOG CANINE))
- SPELLING variations are indicated with a ?
  (e.g., DOG? to search DOG, DOGS)
- Terms that MUST be present are flagged with an asterisk
  (e.g., DOG *FOOD)
Q = QUIT   H = HELP
? komodo dragon food diet nutrition
Your TARGET search request will retrieve up to 50 of the
statistically most relevant records.
Searching 1997-1998 records only
...Processing Complete
      Your search retrieved 50 records.
Press ENTER to browse results  C = Customize display  Q = QUIT
H = HELP
```



# Pattern Matching

- Pattern matching: allow the retrieval of docs based on some **patterns**
  - A pattern is a set of syntactic features must occur in a text segments
    - Segments satisfying the pattern specifications are said to “match the pattern”
    - E.g. the prefix of a word
  - A kind of data retrieval
- Pattern matching (data retrieval) can be viewed as an enhanced tool for information retrieval
  - Require more sophisticated data structures and algorithms to retrieve efficiently

# Pattern Matching (cont.)

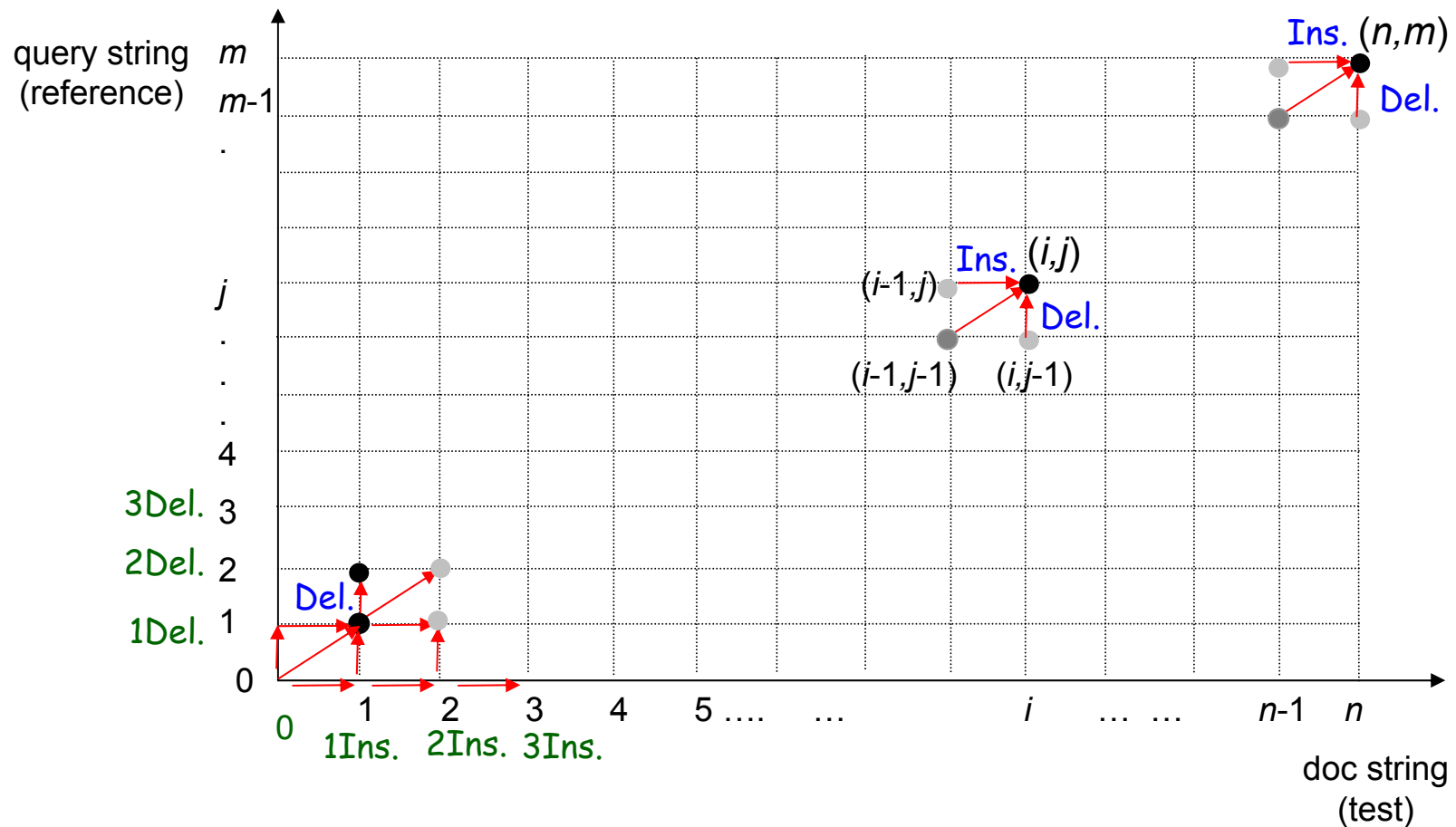
- Types of patterns
  - **Words:** most basic patterns
  - **Prefixes:** a string from the beginning of a text word
    - E.g. ‘comput’: ‘computer’, ‘computation’,...
  - **Suffixes:** a string from the termination of a text word
    - E.g. ‘ters’: ‘computers’, ‘testers’, ‘painters’,...
  - **Substrings:** A string within a text word
    - E.g. ‘tal’: ‘coastal’, ‘talk’, ‘metallic’, ...
  - **Ranges:** a pair of strings matching any words lying between them in lexicographic order
    - E.g. between ‘held’ and ‘hold’: ‘hoax’ and ‘hissing’,...

# Pattern Matching (cont.)

- **Allowing errors:** a word together with an error threshold
  - Useful for when query or doc contains typos or misspelling
  - Retrieve all text words which are ‘similar’ to the given word
  - **edit (or Levenshtein) distance:** the minimum number of character insertions, deletions, and replacements needed to make two strings equal
    - E.g. ‘flower’ and ‘flo wer’
  - **maximum allowed edit distance:** query specifies the maximum number of allowed errors for a word to match the pattern

# Pattern Matching (cont.)

- String Alignment: Using Dynamic Programming



# Pattern Matching (cont.)

- String Alignment: Using Dynamic Programming

Step 1: Initialization :

```
G[0][0] = 0;
for i = 1,..., n { //test
    G[i][0] = G[i - 1][0] + 1;
    B[i][0] = 1; //Insertion
} (Horizontal Direction)
for j = 1,..., m { //reference
    G[0][j] = G[0][j - 1] + 1;
    B[0][j] = 2; // Deletion
} (Vertical Direction)
```



Step 2: Iteration:

```
for i = 1,..., n { //test
    for j = 1,..., m { //reference
        G[i][j] = min [
            G[i - 1][j] + 1 (Insertion)
            G[i][j - 1] + 1 (Delection)
            G[i - 1][j - 1] + 1 (if LR[i] != LT[i], Substitution)
            G[i - 1][j - 1] (if LR[i] = LT[i], Match)
        ]
        B[i][j] {
            1; //Insertion, (Horizontal Direction)
            2; //Deletion, (Vertical Direction)
            3; //Substitution (Diagonal Direction)
            4; //match (Diagonal Direction)
        }
    } //for j, reference
} //for i, test
```

Step 3: Measure and Backtrace :

$$\text{String Error Rate} = 100\% \times \frac{G[n][m]}{m}$$

String Accuracy Rate = 100% – Word Error Rate

Optimal backtrace path = (B[n][m] → ..... → B[0][0])

if B[i][j] = 1 print " LT[i]" ; //Insertio n, then go left

else if B[i][j] = 2 print "LR[j] " ; //Deletion , then go down

else print "LR[j] LR[i]" ; //Hit/Matc h or Substituti on, then go down diagonally

Note: the penalties for substitution, deletion and insertion errors are all set to be 1 here

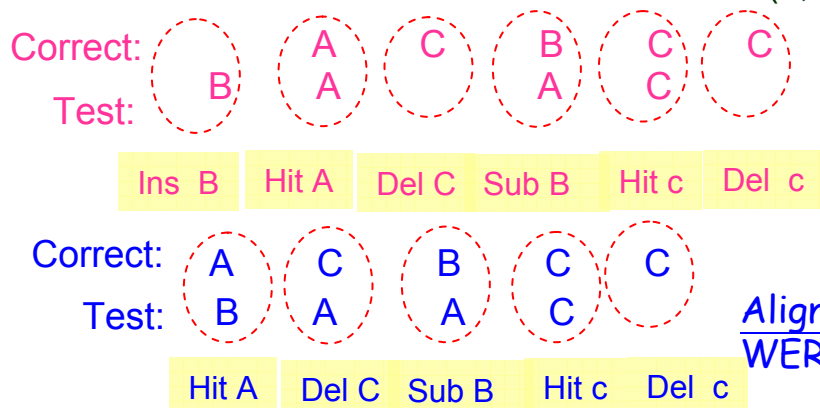
# Pattern Matching (cont.)

- String Alignment: Using Dynamic Programming

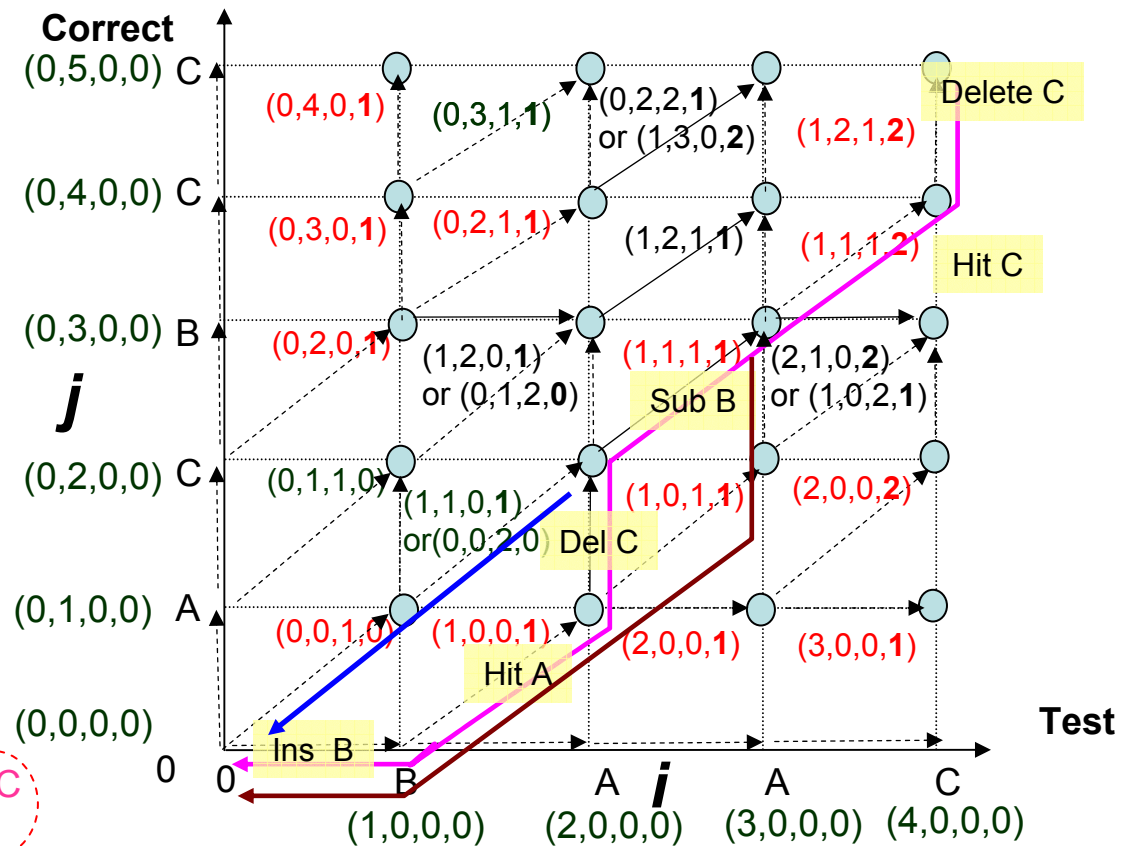
Note: the penalties for substitution, deletion and insertion errors are all set to be 1 here

(Ins, Del, Sub, Hit)

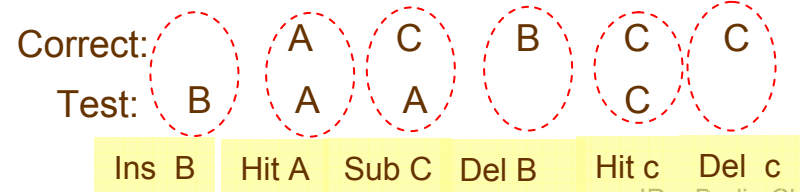
Alignment 1: WER= 80%



Alignment 2:  
 WER=80%



Alignment 3:  
 WER=80%



# Pattern Matching (cont.)

## – Regular Expressions

- General patterns are built up by simple strings and several operations
- **union**: if  $e_1$  and  $e_2$  are regular expressions, then  $(e_1 \mid e_2)$  matches what  $e_1$  or  $e_2$  matches
- **concatenation**: if  $e_1$  and  $e_2$  are regular expressions, the occurrences of  $(e_1 e_2)$  are formed by the occurrences of  $e_1$  immediately followed by those of  $e_2$
- **repetition** (Kleene closure): if  $e$  is a regular expression, then  $(e^*)$  matches a sequence of zero or more contiguous occurrence of  $e$
- Example:
  - ‘pro (blem | tein) (s |  $\varepsilon$ ) (0 | 1 | 2)\*’ matches words ‘problem2’, ‘proteins’, etc.

# Pattern Matching (cont.)

## – Extended Patterns

- Subsets of the regular expressions expressed with a simpler syntax
- System can convert extended patterns into regular expressions, or search them with specific algorithms
- E.g.: **classes of characters:**

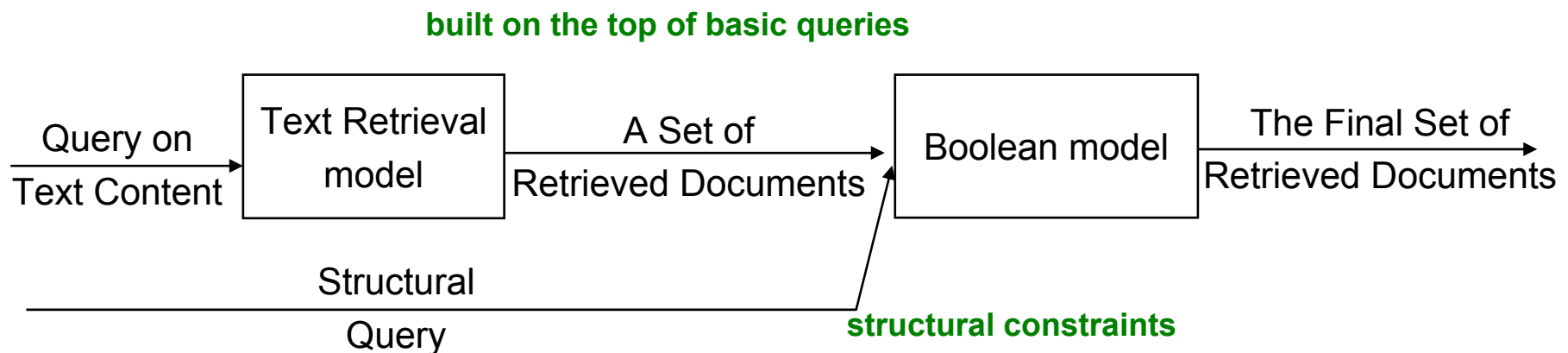
| RE              | Expansion                 | Match                     | Example Patterns        |
|-----------------|---------------------------|---------------------------|-------------------------|
| <code>\d</code> | <code>[0-9]</code>        | any digit                 | <code>Party_of_5</code> |
| <code>\D</code> | <code>[^0-9]</code>       | any non-digit             | <code>Blue_moon</code>  |
| <code>\w</code> | <code>[a-zA-Z0-9_]</code> | any alphanumeric or space | <code>Daiyu</code>      |
| <code>\W</code> | <code>[^\w]</code>        | a non-alphanumeric        | <code>!!!!</code>       |
| <code>\s</code> | <code>[_\r\t\n\f]</code>  | whitespace (space, tab)   |                         |
| <code>\S</code> | <code>[^\s]</code>        | Non-whitespace            | <code>in_Concord</code> |



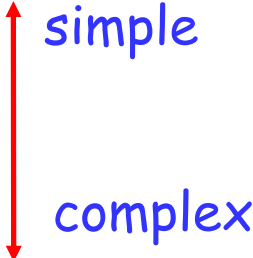
# Structural Queries

- Docs are allowed to be queried with respect to both their text content and structural constraints
  - **Text content:** words, phrases, or patterns
  - **Structural constraints:** containment, proximity, or other restrictions on the structural elements (e.g., chapters, sections, etc.)
- Standardization of languages used to represent structured text, e.g., HTML...

## Mixing contents and structures in queries



# Structural Queries (cont.)

- Three main (text) structures discussed here
    - Form-like fixed structure
    - Hierarchical structure
    - Hypertext structure
- 

What structure a text may have?

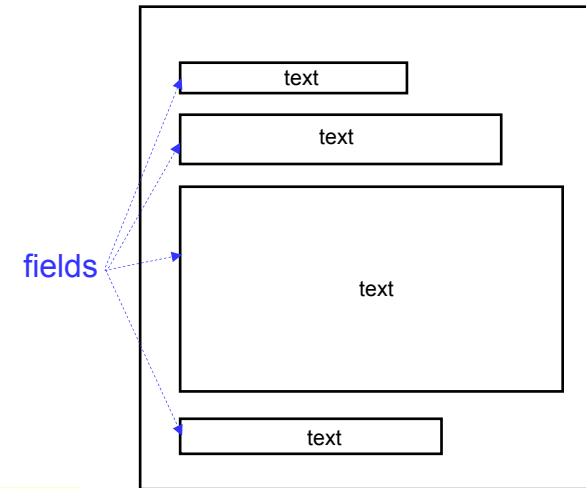
What can be queried about that

structure? (the query model)

How to rank docs?

# Form-like Fixed Structure (cont.)

- Docs have a fixed set of **fields**, much like a filled form
  - Each field has some text inside
  - Some fields are not presented in all docs
  - Text has to be classified into a field
  - Fields are not allow to nest or overlap
  - A given pattern only can be associated with a specified filed

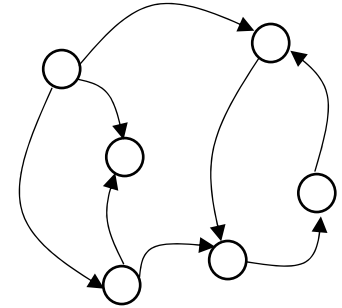


couldn't represent the text hierarchy

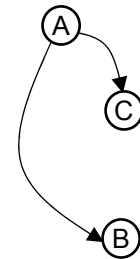
- E.g., a **mail achieve** (sender, receiver, date, subject, body ..)
  - Search for the mail sent to a given person with “football” in the subject field
- Compared with the relational database systems
  - Different fields with different data types **more rigid !**


# Hypertext Structure (cont.)

- A hypertext is a directed graph where
  - Nodes hold some text (**content**)
  - The links represents connection (**structural connectivity**) between nodes or between positions inside the nodes



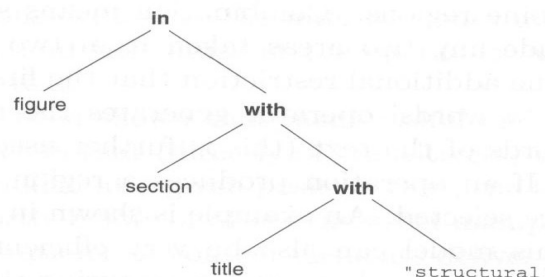
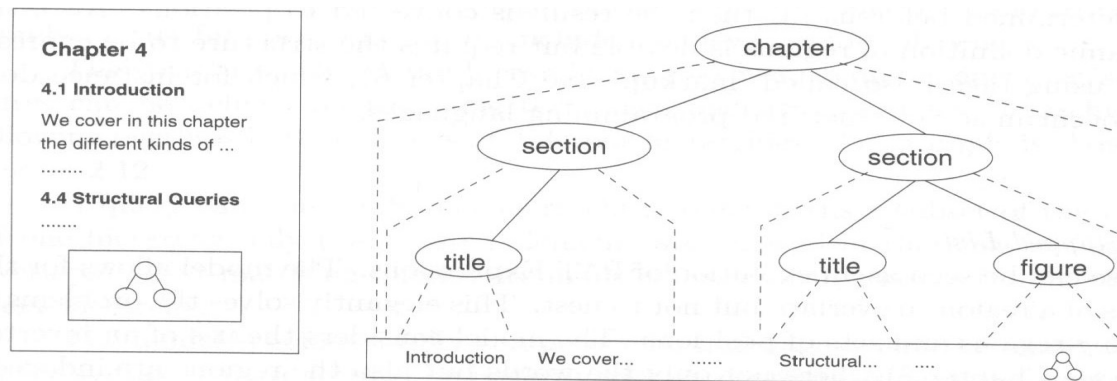
- Retrieval from a hypertext began as a merely navigational activity
  - **Manually** traverse the hypertext nodes following links to search what one wanted
  - **It's still difficult to query the hypertext based on its structure**



- An interesting proposal to combine browsing and searching on the web  **WebGlimpse**
  - Allow classical navigation plus the ability to search by content in the neighborhood of the current node

# Hierarchical Structure (cont.)

- An intermediate structuring model which lies between form-like fixed structure and hypertext structure
- Represent a recursive decomposition of the text and is a natural model for many text collections
  - E.g., books, articles, legal documents,...



A parsed query used to retrieve the figure

# Issues of Hierarchical Structure

- Static or dynamic structure
  - **Statistic**: one or more explicit hierarchies can be queried, e.g., by ancestry
  - **Dynamic**: not really a hierarchy, the required elements are built on the fly
    - Implemented over a normal text index
- Restrictions on the structure
  - The text or the answers may have restrictions about nesting and/or overlapping for efficiency reasons
  - In other cases, the query language is restricted to avoid restricting the structure

*The more powerful the model, the less efficiently it can be implemented*

# Issues of Hierarchical Structure (cont.)

- Integration with text

- Effective Integration of queries on text content with queries on text structure
- From perspectives of classical IR models and structural models, respectively

Classical model: primary -> text  
secondary->structure  
Structural model: primary -> structure  
secondary->text

- Query language

- Some features for queries on structure including selection of areas that
  - Contain (or not) other areas
  - Are contained (or not) in other areas
  - Follow (or are followed by) other areas
  - Are close to other areas
- Also including set manipulation

# Query Protocols

- The query languages used automatically by software applications to query text databases
  - Standards for querying CD-ROMs
  - Or, intermediate languages to query library systems
  
- Important query protocols
  - Z39.50
    - For bibliographical information systems
    - Protocols for not only the query language but also the client-server connection
  - WAIS (Wide Area Information Service)
    - A networking publishing protocol
    - For querying database through the Internet



# Query Protocols (cont.)

- CD-ROM publishing protocols
  - Provide “disk interchangeability”: flexibility in data communication between primary information providers and end users
  - Some example protocols
    - CCL (Common Command Language)
    - CD-RDx (Compact Disk Read only Data exchange)
    - SFQL (Structured Full-text Query Languages)

# Trends and Research Issues

- Types of queries and how they are structured

