



Chapter 3



Query Structures

1



In this chapter

- **Covers the various structures that have been proposed**
- **Used for queries to a retrieval system**

6-May-99

IR/Query Structure

2



Query Structures

- Relatively brief, quite constrained, indeed
- May not satisfy normal rules of syntax
- Must be transformed into format that can be matched to the document

6-May-99

IR/Query Structure

3



Query Structures

- Word frequency counts
 - useful in processing longer document
 - little or no use in queries
- Within the endosystem
 - the document is transformed into an internal representation and further transformed into a format that can be used in the matching process

6-May-99

4



Matching Criteria

- Exact match
 - with numerical or business databases
- Range match
 - a range of acceptable value ;
select name, age from person where age > 25
 - possible on terms that have a natural order (numerical or alphabetical)

6-May-99

IR/Query Structure

5



Matching Criteria

- Approximate match
 - most text and image databases
 - data are not well organized
 - type of queries posed are not that precise

6-May-99

IR/Query Structure

6



Boolean Queries

- Think of a query in one of two forms
 - natural language
"give me the names of all documents containing sentences with the words 'computer' and 'science'"
 - list of terms : one standard model is Boolean query

6-May-99

IR/Query Structure

7



Boolean Queries

- Based on concepts from logic or Boolean algebra (AND, OR, NOT)
- Boolean expressions are formed from queries

6-May-99

IR/Query Structure

8

Boolean expression

- Represent a request to determine what documents contain a set of keywords

Query
Find all documents containing 'information'

Boolean expression
information

6-May-99

IR/Query Structure

9

Boolean expression

- Some queries attempt to find document that do not contain a particular pattern

Query
Find all documents that do not
containing 'information'

Boolean expression
not information

6-May-99

IR/Query Structure

10

Boolean expression

- Most queries involve searching more than one term

Query
Find all documents containing
'information' and 'retrieval'

Boolean expression
information and retrieval

6-May-99

IR/Query Structure

11

Boolean expression

- Most queries involve searching more than one term

Query
Find all documents containing
'information' or 'retrieval' (or both)

Boolean expression
information or retrieval

6-May-99

IR/Query Structure

12

Boolean expression

- Most queries involve searching more than one term

Query
Find all documents containing
'information' or 'retrieval' (but not both)

Boolean expression
information xor retrieval

6-May-99

IR/Query Structure

13

Boolean expression

- Boolean expression may be formed from other Boolean expressions to yield rather complex structures

Query
Find all documents containing
'information', 'retrieval' or not containing
both 'retrieval' and 'science')

Boolean expression
(information and retrieval) or not
(retrieval and science)

6-May-99

IR/Query Structure

14

Example

Document	Terms
doc1	algorithm,information,retrieval
doc2	retrieval,science
doc3	algorithm,information,science
doc4	pattern,retrieval,science
doc5	science,algorithm

information \Rightarrow {doc1, doc3}

6-May-99

IR/Query Structure

15

Example

Boolean expression
information and retrieval



{doc1, doc3} \cap {doc1, doc2, doc4} = {doc1}

6-May-99

IR/Query Structure

16

Example

Boolean expression
information or retrieval



$\{\text{doc1, doc3}\} \cap \{\text{doc1, doc2, doc4}\} =$
 $\{\text{doc1, doc2, doc3, doc4}\}$

Boolean Queries

- The search may be expanded by stemming, or reduction of a word to its root form

Example

- Query
 - restaurants AND Mideastern OR vegetarian AND inexpensive
- The query might be replaced by
 - restaurants AND Mideast OR veget AND inexpens

Boolean Queries

- Another expansion technique is to use a thesaurus or list of related terms
 - Indian could be included in expanding the term vegetarian, even though the country is not considered to be Mideastern



Stemming Algorithms

- Used in IR to reduce the size of index files
- Terms can be stemmed
 - at indexing time
 - at search time

6-May-99

IR/Query Structure

21



Stemming at indexing time

- 4 efficiency
- 4 index file compression
- 4 no resources at search time

7 Information about full terms will be lost

6-May-99

IR/Query Structure

22



Taxonomy for stemming algo

- 1 Affix (suffix, prefix) removal algorithms
- 2 Successor variety stemmers
 - use the frequencies of letter sequences in a body of text as the basis of stemming

6-May-99

IR/Query Structure

23



Taxonomy for stemming algo

- 3 The n-gram method
 - conflates terms based on the number of diagram or n-grams they share
- 4 Table lookup
 - F terms and their corresponding stems can also be stored in a table

6-May-99

IR/Query Structure

24

Criteria for judging stemmer

- Correctness
 - Overstemming : too much terms removed which effect IR performance
 - retrieval nonrelevant documents
 - Understemming : too little terms removed
 - relevant document will not be retrieved

6-May-99

IR/Query Structure

25

Criteria for judging stemmer

- Retrieval effectiveness
 - measured with recall and precision
 - speed, size
- Compression performance

6-May-99

IR/Query Structure

26

Stemmer use in searching

- CATALOG system : stemmed at search time rather than at indexing time

Look for : system users

Search term : users

Term	Occurrences
1. User	15
2. Users	1
3. Used	3
4. Using	2

6-May-99

IR/Query Structure

27

Table Lookup

- Store a table of all index terms and their stems
- Using a B-tree or hash table, such lookup would be very fast

<u>Term</u>	<u>Stem</u>
engineering	engineer
engineered	engineer
engineer	engineer

6-May-99

IR/Query S.....

28

To be next

- N-gram stemmers
- Affix removal stemmers

N-gram Stemmers

- Adamson and Boreham 1974 reported this method call "the shared digram method"
- This is a bit confusing since no stem is produced
- Association measures are calculated between pairs of terms based on shared unique digrams

N-gram stemmers

Statistics => st ta at ti is st ti ic cs
unique digrams = at cs ic is st ta ti

Statistical => st ta at ti is st ti ic ca al
unique digrams = al at ca ic is st ta ti

Share unique digrams = at ic is st ta ti

N-gram stemmers

Similarity measure used was Dice's coefficient

$$S = 2C/(A+B)$$

A = # of unique digrams in the 1st word

B = # of unique digrams in the 2nd

C = # of unique digrams shared by A,B

$$S = (2 \cdot 6) / (7 + 8) = 0.80$$

A = 7

Statistics => st ta at ti is st ti ic cs

unique digrams = at cs ic is st ta ti

B = 8

Statistical => st ta at ti is st ti ic ca al

unique digrams = al a is st ta ti

C = 6

Share unique digrams = at ic is st ta ti

33

N-gram stemmers

- Once such a similarity matrix is available
- Terms are clustered using a single link clustering method



6-May-99

IR/Query Structure

34

N-gram stemmers

- Most pairwise similarity measures were 0
 - matrix will be sparse (thin)
- Using similarity value of 0.60
 - found that clusters formed were correct



6-May-99

IR/Query Structure

35

Affix Removal Stemmers

- Remove the plurals from terms
- If a word ends in "ies" but not "eies" or "aies" then "ies" -> "y"

Ex. Babies -> baby
directories->directory
but not
trees

6-May-99

IR/Query Structure

36

Affix Removal Stemmers

- Remove the plurals from terms

If a word ends in “ies” but not “eies” or “aies”
then “ies” -> “y”

Ex. Babies -> baby

directories->directory

Problem

Skies -> ski sky not match

Solve by recoding or partial matching

6-May-99

IR/Query Structure

37

Affix Removal Stemmers

- Recoding is a context sensitive transformation

if a stem ends in a “i” following a “k”
then i->y

Ex. ski -> sky

6-May-99

IR/Query Structure

38

Affix Removal Stemmers

- Remove the plurals from terms

Word ends in “es” but not “aes” or “ees” or “oes”
then “es” -> “e”

Ex. messages -> message

garages -> garage

but not bees-> bee shoes-> shoes

6-May-99

IR/Query Structure

39

Affix Removal Stemmers

- Remove the plurals from terms

If a word ends in “s” but not “us” or “ss”
then “s” -> NULL

Ex. address, dress, mass
thesaurus

6-May-99

IR/Query Structure

40

Stemmers

- Most Stemmers currently in use are iterative longest match stemmers
 - developed by Lovins 1968
- In addition to Lovins, longest match Stemmers have been reported by
 - Salton 1968
 - Dawson 1974
 - Porter 1980 and Paice 1990

6-May-99

IR/Query Structure

41

Stemming to compress files

- Stem can ↓ the size of index files
- Harman 1991 report that
 - using Lovins stemmer for database of 1.6 MB ↓ of source text
 - can ↓ 20% of index file from .47 MB → .38 MB
 - for database 50MB, index was ↓ from 6.7 → 5.8 MB saving 13.5%

6-May-99

IR/Query Structure

42

Problem in Boolean Queries

1 In a pure Boolean query

- * No good way to weight terms for significance either present or absent
- * Little control over how important a given term is to the query

6-May-99

IR/Query Structure

43

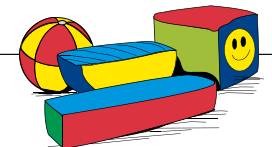
Problem in Boolean Queries

Boolean request

music by Beethoven, preferably a sonata

Boolean query

Beethoven AND sonata



6-May-99

IR/Query Structure

44

Problem in Boolean Queries

2 Retrieval results that are wrong because of a misstated query

- incorrect interpretation of Boolean connectives AND, OR
- not fully conversant with logical conventions, misuse them in certain situations

6-May-99

IR/Query Structure

45

Example :

- A person seeking Saturday night entertainment may specify an interest in
 - dinner AND sports AND symphony
 - dinner OR sports OR symphony
 - dinner AND (sports OR symphony)

6-May-99

IR/Query Structure

46

Problem in Boolean Queries

3 The order of precedence

- rely on () to group term
- NOT \Rightarrow AND \Rightarrow OR
- people tend to interpret this ambiguity one way or another depend on the semantic relationships among the 3 terms

6-May-99

IR/Query Structure

47

Problem in Boolean Queries

coffee AND croissant OR muffin
differ from

raincoat AND umbrella OR sunglasses

- Most users of IS are not well trained in Boolean algebra
- Many users do not access an info retrieval system on a regular basis

6-May-99

IR/Query Structure

48

Problem in Boolean Queries

4 User is free to construct a highly complex query

- Boolean query can be recast into
 - disjunctive normal form (DNF)
 - conjunctive normal form (CNF)

6-May-99

IR/Query Structure

49

Disjunctive Normal Form

- Three levels of expression in a DNF
 - Terms : words that occur either naturally or negated
 - concert, NOT play is valid
 - Conjuncts : joined by AND
 - dinner AND concert AND NOT play is valid

6-May-99

IR/Query Structure

50

Disjunctive Normal Form

- Disjuncts : joined by OR
(swimming AND tennis) OR
(baseball AND NOT football)

Query in this form can be split into smaller queries

6-May-99

IR/Query Structure

51

Disjunctive Normal Form

(A AND B) OR (A AND NOT C)



(A AND B AND C) OR
(A AND B AND NOT C) OR
(A AND NOT B AND C)

6-May-99

IR/Query Structure

52

Conjunctive Normal Form

- The roles of AND and OR interchanged
 - OR to form disjunct
 - AND to form conjunct
- (concert OR dinner OR NOT play)
AND(swimming OR tennis) AND
(baseball OR NOT football)

6-May-99

IR/Query Structure

53

Normalization

- The process of transforming a Boolean query into either DNF or CNF
- Used of truth table

6-May-99

IR/Query Structure

54

The full DNF

- Full DNF for the query is formed by taking the truth table rows that are true and combining them with OR

6-May-99

IR/Query Structure

55

The full CNF

- Begin by forming the full DNF using the false rows of the table
- Interchanging AND and OR

$$\text{NOT}(A \text{ AND } B) = (\text{NOT } A) \text{ OR } (\text{NOT } B)$$

$$\text{NOT}(A \text{ OR } B) = (\text{NOT } A) \text{ AND } (\text{NOT } B)$$

$$\text{NOT}(\text{NOT } A) = A$$

6-May-99

IR/Query Structure

56

Example :

(A AND B AND NOT C) OR (B AND C)

- CNF for the query is determined by negating expression and expanding it

NOT((A AND B AND NOT C) OR (B AND C))
~~NOT(A AND B AND NOT C) AND NOT(B AND C)~~

6-May-99

IR/Query Structure

57

Normalization of Vocabulary

- There are extensive rules which guide the form of the thesaurus entries
 - terms should be in noun form
 - noun phrases should avoid preposition unless they are commonly known
 - singularity of terms
 - ordering of term within phrase

6-May-99

IR/Query Structure

58

Problem in Boolean Queries

5 Controlling the size and composition of the retrieved set

- One solution begin with a more restrictive query
 - Frame new query by adding more terms to original query or construct an entirely new set of term

6-May-99

IR/Query Structure

59

Problem in Boolean Queries

5 Controlling the size and composition of the retrieved set

- Sorting doc by the number of matching query terms provides a rough ranking of the doc
 - uses thesaurus to expand the query terms

6-May-99

IR/Query Structure

60

Problem in Boolean Queries

6 Philosophical problem

- since this is a list not a Boolean expression
- Structural differ between the query and the doc
- Better to think of Boolean retrieval as a mapping process rather than a matching process

6-May-99

IR/Query Structure

61

Ranking Algorithms

- Because of the complex query syntax required
- The ranking approach is more effective for end-users
 - allow user to input a simple query such as a sentence or a phrase (no Boolean connectors)

6-May-99

IR/Query Structure

62

Ranking Algorithms

- the results being ranked based on co-occurrence of query terms
- modified by statistical term weighting
- eliminate the often wrong Boolean syntax used by end-users
- provides some results even if a query term is incorrect

6-May-99

IR/Query Structure

63

How ranking is done

- Textual data set uses i unique terms
- Document can then be represented by a vector $(t_1, t_2, t_3, \dots, t_n)$

$t_i = 1$ if term i is present

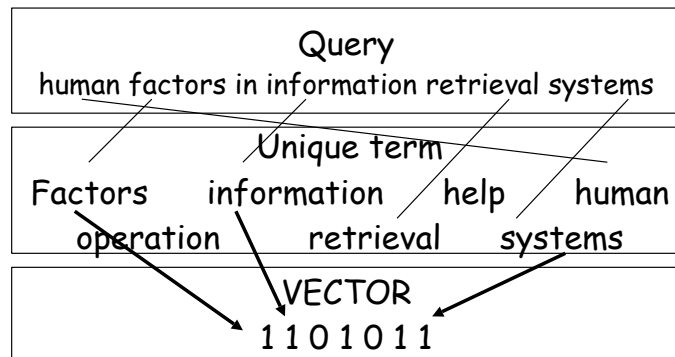
$t_i = 0$ if term i is absent

6-May-99

IR/Query Structure

64

Conceptual Term Weighting

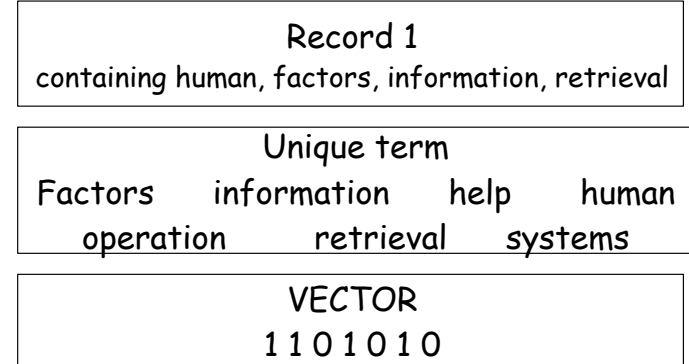


6-May-99

IR/Query Structure

65

Conceptual Term Weighting

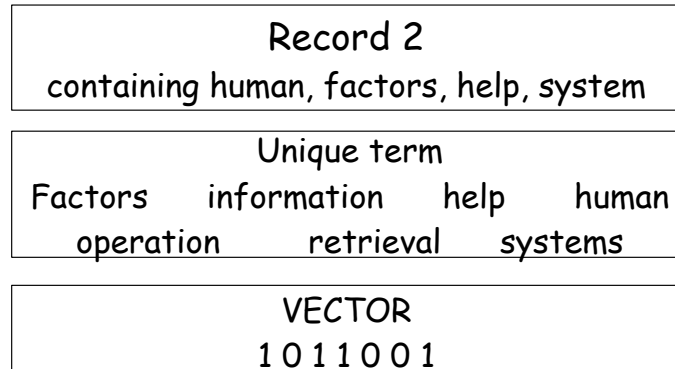


6-May-99

IR/Query Structure

66

Conceptual Term Weighting

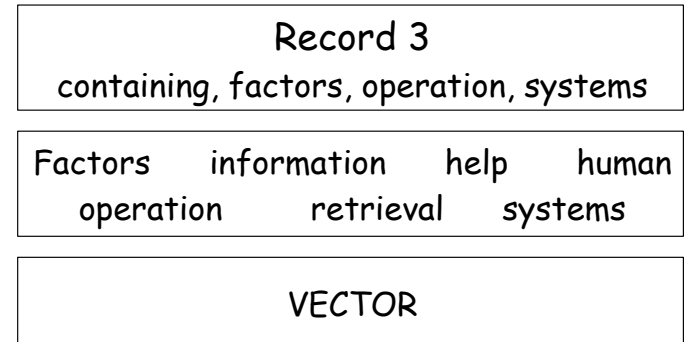


6-May-99

IR/Query Structure

67

Conceptual Term Weighting



6-May-99

IR/Query Structure

68

Simple match

Q (1 1 0 1 0 1 1) 4
R1 (1 1 0 1 0 1 0)
Q (1 1 0 1 0 1 1) 3
R2 (1 0 1 1 0 0 1)
Q (1 1 0 1 0 1 1) ?
R3 (1 0 0 0 1 0 1)

6-May-99

IR/Query Structure

69

Conceptual Term Weighting

- Possible to perform the same operation using weighted vectors
- Term is weighted by the total number of times it appears in the record
- Term-weights could reflect different measures

6-May-99

IR/Query Structure

70

Weighted match

Q (1 1 0 1 0 1 1) 13
R1 (2 3 0 5 0 3 0)
Q (1 1 0 1 0 1 1) 8
R2 (2 0 4 5 0 0 1)
Q (1 1 0 1 0 1 1) ?
R3 (2 0 0 0 2 0 1)

6-May-99

IR/Query Structure

71

Ranking models

- Divided into two types
 - rank the query against individual documents
 - the vector space model
 - probabilistic model
 - rank the query against entire sets of related documents

6-May-99

IR/Query Structure

72