INFSCI 2140
Information Storage and Retrieval
Lecture 2: Models of Information Retrieval:
Boolean model

Peter Brusilovsky
http://www2.sis.pitt.edu/~peterb/2140-061/

Final Group Projects

- Groups of variable size - match the project complexity
- Goal - learn more about IR, do some useful and practical work
- Educational IR system
- Exploring IR aspects
- Analysis
What are your strong sides?

Are you good in:
- Reading and analyzing text
- Programming in C or Java
- Database programming
- Web programming
- Developing educational material
- Graphic arts and design

Educational IR system

Small educational programs: demo-simulation / assessment / ITS. Learn nuts and bolts of some complex aspect and help others to learn

Here are some examples from the past semantics
- Boolean query
- Boolean document matching
- Evaluation measures
- Vectors, distances, similarities
Exploring IR aspects

- Explore Information Visualization
  - Small application that we can demonstrate

- Any part of IR process to practice
  - Document preparations and processing
  - Simple search engine
  - Similarity-based navigation
  - Clustering
  - Adaptive recommendation

Analysis

- Select a small field of interest on the frontier of IR, analyze a few relevant papers and systems, write an essay (indexed!), prepare a short talk for the last lecture
  - Improving Web search
  - Machine learning for IR
  - Some type of information visualization
  - Clustering
  - Web recommenders...
Overview

- Information Retrieval Models
- Boolean Model
  - Boolean queries
  - DNF and CNF
  - Boolean matching

From information need to documents

Documents → Surrogates

Information need → Query
Classic ad-hoc IR

- Documents form an unstructured document space
- Information need made explicit in a form of a query
- User searches once - one query against the document space (ad-hoc IR)
- The IR system returns an ordered subset of documents that are relevant to the query

User’s prospect

- Query
  - a question in natural language
  - a list of terms
- Matching
  - The obvious exact match
  - The range match
  - Approximate match? (“tell me something about…”, “tell me something that I don’t know?”)
Problems with matching

- The fact that a document contains a term requested in a query doesn’t mean that the document should be retrieved:
  - we have to consider all the term and the condition expressed in the query
  - it doesn’t mean that the document is strongly related to the term
  - a document can’t be suitable for other reasons (age for example)

Exact Match

- Examples
  - all the documents authored by …
  - all the documents that have in the title the word …
- It is possible with well structured documents and databases
Range Match

- Example
  - all the documents authored from “date1” to “date2”.
- Range match works with numerical databases

Approximate match

- Necessary if the information need is not precise and if the database is not well organized.
- Sometimes it is necessary to combine the different kind or queries.
Document Space

For our purposes we call document space the set of document representation.

Document space is organized in some way. For example, it could be possible to calculate a distance or a similarity between different documents.
Designer’s prospects for Q & M

- Can we consider query as a document?
  - No
    - let’s consider a query as a characteristic function defined for document space
  - Yes
    - let’s put a query into a space and calculate “closeness” or similarity
- In any case we need a measure for relevance (“closeness”, similarity…)

Query as a characteristic function

- If a query is NOT a part of the document space, we can consider it as a characteristic function
- Each query defines a characteristic function f(d)
  - It takes a document as a parameter
  - Its value represents document relevancy
Yes or No relevance functions

- The query defines a characteristic function: a function that have a value 0 or 1 depending on document relevancy to a query

\[
\{ \begin{align*}
1 & \text{ if relevant} \\
0 & \text{ if not relevant}
\end{align*} \]

Complex relevance functions

- In more sophisticated systems the evaluation function can have values in the interval \([0,1]\), allowing to rank documents.

\[
\{ \begin{align*}
1 & \text{ if relevant} \\
0 & \text{ if not relevant}
\end{align*} \]
Models: Classic and New

- **Boolean Model**
  - Classic
  - Extended
  - Fuzzy

- **Vector Model**
  - Classic
  - Others (generalized, LSI, Neural Networks)

- **Probabilistic Model**
  We will work by models, discussing Q&M for each model separately

### Boolean Model

- **Set Theoretic Approach**
- Documents form a large set
- A query defines a subset (i.e., 0 or 1)
- Each *elementary query* defines an elementary subset
  - Depending on the kind of the query
- A complex query is produced from elementary queries using Boolean functions
- The subset for the complex query is produced by performing Boolean set operations with elementary subsets
Boolean queries

- Boolean queries are based on Boolean Algebra / set theory
- Terms are joined using Boolean operations such as AND, OR, NOT
- The search can be expanded using stemming, thesaurus or list of related terms
- Example:
  restaurants AND (mideastern OR vegetarian) AND inexpensive

Elementary Query

- Formatted fields
  - Exact match (Year = 1999)
  - Range match (Price < $50)
- Text fields
  - Word inclusion (Programming in Title)
  - Stemming/wildcards (Program$ in Title)
  - Phrase inclusion (“Programming language” in Title)
  - Approximate match (sounds like “John”)
Venn Diagrams for set operations

- Used to clarify different set operations

Boolean Operators

- **Q1 AND Q2**
  - Documents that are in BOTH sets: Q1 and Q2
- **Q1 OR Q2**
  - Documents that are in at least in one set: Q1 or Q2
- **NOT Q1**
  - All documents except the one in set Q1
Other Boolean Operators

- **Q1 \ Q2**
  - Logical “minus” all documents from Q1 except those that belong to Q2
  - Used also as “binary NOT” (Q1 NOT Q2)

- **Q1 XOR Q2**
  - Exclusive OR - documents that belong to exactly one set: Q1 or Q2, but not both
  - In other words (Q1 OR Q2) \ (Q1 AND Q2)

Some Special Operators

- **Proximity (TNT)**
  - (Information within 1 word from Retrieval)
  - This is really a special elementary query to structured text fields

- **NOF (N of)**
  - 2 of (Sashimi, Sushi, Shabu-Shabu)
  - This is a simple form that can be expressed by a regular query
Complex Queries

- We can use braces to determine the order of operation
  - $(\text{NOT } (Q_1 \text{ OR } Q_2 \text{ OR } Q_3) \text{ AND } (Q_4 \text{ AND } \text{NOT } Q_5) \text{ OR } Q_6)$
- Complex queries are hard to write, understand and perform
- Normalization - converting a query to one of the *normal forms*

Understanding complex queries

- You should be able to understand a complex query as
  - A sequence of set theory operations
  - A Boolean function returning 0-1 for each combination of parameters
- Could be on tests/exams
- Work with our Venn diagram and CNF/DNF applications
Normal Forms

- Every Boolean query can be recast into:
  - Conjunctive Normal Form (CNF)
    - $Q_1 \text{ AND } Q_2 \text{ AND } Q_3 \text{ AND } Q_4$
    - Each $Q$ is a disjunct
  - Disjunctive Normal Form (DNF)
    - $Q_1 \text{ OR } Q_2 \text{ OR } Q_3 \text{ OR } Q_4$
    - Each $Q$ is a conjunct

Disjunctive Normal Form (DNF)

- Term: positive or negated elementary fact
  - Year = 1999; NOT (Director = Spielberg);
  - Fire; NOT Retrieval
- Conjuncts: conjunction of terms
  - Year = 1999 AND NOT (Director = Spielberg)
- Query: disjunction of conjuncts
  - $(a \text{ AND } b) \text{ OR } (c \text{ AND } d) \text{ OR } (\text{NOT } a \text{ AND } d)$
**Boolean queries: DNF**

- The advantage is that the query can be split into many different queries. The results of the different queries is merged to produce the response to the original query:
  
  \[(a \land b) \lor (c \land d) \lor (\neg a \land d)\]

**Conjunctive Normal Form (CNF)**

- **Term**: positive or negated elementary fact
  - Year = 1999; NOT (Director = Spielberg);
  - Fire; NOT Retrieval
- **Disjuncts**: disjunction of terms
  - Year = 1999 OR NOT (Director = Spielberg)
- **Query**: conjunction of disjuncts
  \[(a \lor b) \land (c \lor d) \land (\neg a \lor d)\]
Boolean queries: CNF

- Using a thesaurus can be easy to expand a CNF query to have a broader query:
  \[(a \text{ OR } b) \text{ AND } (c \text{ OR } d) \text{ AND } (\text{NOT } a \text{ OR } d)\]

Normalization

- Convert each Boolean function to NFs
- Key: Truth Table
  - Example: \((A \text{ OR NOT } B) \text{ AND } C\)

<table>
<thead>
<tr>
<th>A</th>
<th>B</th>
<th>C</th>
<th>not B</th>
<th>A or (not B)</th>
<th>(A or (not B)) and C</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>1</td>
<td>0</td>
</tr>
<tr>
<td>0</td>
<td>0</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td>0</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>0</td>
<td>1</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>1</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>1</td>
<td>0</td>
</tr>
<tr>
<td>1</td>
<td>0</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td>1</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>0</td>
</tr>
<tr>
<td>1</td>
<td>1</td>
<td>1</td>
<td>0</td>
<td>1</td>
<td>1</td>
</tr>
</tbody>
</table>

- Normalization to DNF
  - Consider T rows
- Normalization to DNF
  - Consider F rows
Example: Normalize the query \( [A \lor \neg B] \) and C to DNF

<table>
<thead>
<tr>
<th>A</th>
<th>B</th>
<th>C</th>
<th>not B</th>
<th>A\lor(\neg B)</th>
<th>[A\lor(\neg B)]\land C</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>1</td>
<td>0</td>
</tr>
<tr>
<td>0</td>
<td>0</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td>0</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>0</td>
<td>1</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>1</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>1</td>
<td>0</td>
</tr>
<tr>
<td>1</td>
<td>0</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td>1</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>0</td>
</tr>
<tr>
<td>1</td>
<td>1</td>
<td>1</td>
<td>0</td>
<td>1</td>
<td>1</td>
</tr>
</tbody>
</table>

Example: Normalize the query \( [A \lor \neg B] \) and C to DNF

<table>
<thead>
<tr>
<th>A</th>
<th>B</th>
<th>C</th>
<th>not B</th>
<th>A\lor(\neg B)</th>
<th>[A\lor(\neg B)]\land C</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>1</td>
<td>0</td>
</tr>
<tr>
<td>0</td>
<td>0</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td>0</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>0</td>
<td>1</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>1</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>1</td>
<td>0</td>
</tr>
<tr>
<td>1</td>
<td>0</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td>1</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>0</td>
</tr>
<tr>
<td>1</td>
<td>1</td>
<td>1</td>
<td>0</td>
<td>1</td>
<td>1</td>
</tr>
</tbody>
</table>

\([(\neg A)\land(\neg B)\land C] \lor [A \land \neg B \land C] \lor [A \land B \land C] \)
Normalization to CNF

A way to obtain a CNF is to use the false (0) row of the table and simplifying them using DeMorgan’s laws and double negation law.

DeMorgan’s laws

1st law: \[ \text{not} (A \text{ and } B) = (\text{not} A) \text{ or } (\text{not } B) \]
2nd law: \[ \text{not} (A \text{ or } B) = (\text{not} A) \text{ and } (\text{not } B) \]

double negation

\[ \text{not}(\text{not } A) = A \]

Example: Normalize the query \([A \text{or}(\text{not } B)] \text{ and } C\) to CNF

<table>
<thead>
<tr>
<th>A</th>
<th>B</th>
<th>C</th>
<th>not B</th>
<th>Aor(notB)</th>
<th>[Aor(notB)]and C</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>1</td>
<td>0</td>
</tr>
<tr>
<td>0</td>
<td>0</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td>0</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>0</td>
<td>1</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>1</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>1</td>
<td>0</td>
</tr>
<tr>
<td>1</td>
<td>0</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td>1</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>0</td>
</tr>
<tr>
<td>1</td>
<td>1</td>
<td>1</td>
<td>0</td>
<td>1</td>
<td>1</td>
</tr>
</tbody>
</table>
Example CNF

Step 1: form the full DNF query using the false row of the table

\[(\neg A) \land (\neg B) \land (\neg C)\] or
\[(\neg A) \land B \land (\neg C)\] or
\[(\neg A) \land B \land C\] or
\[A \land (\neg B) \land (\neg C)\] or
\[A \land B \land (\neg C)\]

<table>
<thead>
<tr>
<th>A</th>
<th>B</th>
<th>C</th>
<th>not B</th>
<th>A or (not B)</th>
<th>(A or (not B)) and C</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>0</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>0</td>
</tr>
<tr>
<td>0</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td>0</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>0</td>
<td>1</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>1</td>
<td>0</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>0</td>
</tr>
<tr>
<td>1</td>
<td>0</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td>1</td>
<td>1</td>
<td>0</td>
<td>1</td>
<td>1</td>
<td>0</td>
</tr>
<tr>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
</tr>
</tbody>
</table>

Example CNF

step 2: negate the whole DNF query

\[\neg \{((\neg A) \land (\neg B) \land (\neg C)) \lor ((\neg A) \land B \land (\neg C)) \lor ((\neg A) \land B \land C) \lor (A \land (\neg B) \land (\neg C)) \lor (A \land B \land (\neg C))\}\]
Example CNF

step 3 : apply the 2nd DeMorgan law

not {{[(not A) and (not B) and (not C)]} and
not {[(not A) and (not B) and (not C)]} and
not {[ (not A) and B and C] } and
not {[A and (not B) and (not C)] } and
not {[A and (not B) and (not C)]}

Example CNF

step 4 : apply the 1st DeMorgan law

[not (not A) or not((not B) or not(not C))] and
[not (not A) or (not B) or not (not C)] and
[not (not A) or (not B) or (not C)] and
[(not A) or not((not B) or not(not C))] } and
[(not A) or (not B) or not(not C)]
Example CNF

**step 5**: apply the double negation law

\[
[A \lor B \lor C] \land [A \lor (\neg B) \lor C] \land [(A) \lor (\neg B) \lor (\neg C)] \land [(\neg A) \lor B \lor C] \land [(\neg A) \lor (\neg B) \lor C]
\]

Boolean Matching

- The query is not a part of the document space. It is a Boolean expression that represent some conditions on the index terms or representation of the desired documents.

\[
\text{(computer AND memory) AND (NOT chip)}
\]

query

\[
\begin{align*}
1 & \text{ if relevant} \\
0 & \text{ if not relevant}
\end{align*}
\]

Doc. representation
Boolean Matching

- How to define match for elementary query?
- How to define match for main boolean operations?
  - Q1 AND Q2
  - Q1 OR Q2
  - NOT Q1
- How to order?

Efficiency for elementary queries

- Fast ways to match
- Inverted Indexing (primary and secondary key):
  - Fixed value search (year = 1999)
  - Full term search (Adaptive in text)
- Hashing
- Range search (Year < 1990)
  - Sorted array search
Boolean queries and inverted file

**Doc1**: the cat is on the mat
**Doc2**: the mat is on the floor

**Inverted file**
- cat:doc1,1
- floor:doc2,5
- mat:doc1,5;doc2,1

Query: cat

**Inverted file**
- cat:doc1,1
- floor:doc2,5
- mat:doc1,5;doc2,1

Answer: doc1
### Boolean queries and inverted file

**Query:** cat AND floor

<table>
<thead>
<tr>
<th>Inverted file</th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>cat:doc1,1</td>
<td></td>
<td></td>
</tr>
<tr>
<td>floor:doc2,5</td>
<td></td>
<td></td>
</tr>
<tr>
<td>mat:doc1,5;doc2,1</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

**Answer:** No document

The algorithm looks at these two lists and try to found elements in common.

### Boolean queries and Term-document matrix

- Rows represent document terms
- Columns represent documents

**Doc1:** the cat is on the mat

**Doc2:** the mat is on the floor

<table>
<thead>
<tr>
<th></th>
<th>Doc1</th>
<th>Doc2</th>
</tr>
</thead>
<tbody>
<tr>
<td>cat</td>
<td>1</td>
<td>0</td>
</tr>
<tr>
<td>floor</td>
<td>0</td>
<td>1</td>
</tr>
<tr>
<td>mat</td>
<td>1</td>
<td>1</td>
</tr>
</tbody>
</table>
Boolean queries and Term-document matrix

Query: Mat

<table>
<thead>
<tr>
<th></th>
<th>Doc1</th>
<th>Doc2</th>
</tr>
</thead>
<tbody>
<tr>
<td>cat</td>
<td>1</td>
<td>0</td>
</tr>
<tr>
<td>floor</td>
<td>0</td>
<td>1</td>
</tr>
<tr>
<td>mat</td>
<td>1</td>
<td>1</td>
</tr>
</tbody>
</table>

Answer: Doc1 and Doc2

More efficiency

- Slower ways to match
- Full comparison
  - Exact comparison (size < 5.9)
  - Text matching
- Combination
  - Do some fast search first, defining a smaller subset, then comprehensive search
Benefits of Boolean model

- Integration of formatted databases and full-text document collections
- Object oriented databases
  - A record is an object and denotes a document
- Boolean Queries work with classic and OO databases (unlike vector model)!
- Example of combined Boolean search

Problems of Boolean Model

- Can’t assign significance for terms
- Complex Boolean queries are hard for users
  - misstated queries
- Order of precedence of OR and AND does matter (A or B and C)
- The problem of NOT - open corpus
- Hard to make efficient queries
- Ordering of the results is hard (yes/no)
- Controlling the size of the answer set
Better GUI for Boolean search

Softening Boolean queries

- Boolean characteristic function provides Exact Match
  - Each document is either in or out

- Softening approach
  - The case of A or B or C
  - Document that satisfy A and B is better than A but worse than A and B and C
  - Provides the ordering and handles the cases with too few or no results
Homework (Part 1)

Using the system at the URL (Boolean query servlet)
http://kt1.sis.pitt.edu:8080/ir/uquery/matrix.html
answer to the following questions (and report the query used):

- Did Mr. Flanagan write any book about “algorithms”?
  Answer to the question and report the query used

- Did Prof. Brusilovsky write any book about Java?
  Answer to the question and report the query used

- List the books about Java or Perl language? (look at the title)
  Answer to the question and report the query used

- List the books related to Java were published in 1999?
  Answer to the question and report the query used

- List the books published by Java Soft were not authored by Flanagan?
  Answer to the question and report the query used

Homework (Part 2)

- Take an example of an advanced online search system (the one you have used in HW1 or any other) retrieval systems and analyze it from the prospect of IR models presented at the lectures 3 and 4. What kind of model this system is based upon? Can you recognize one of the models we have learned? What kind of queries the system allows? How you could use boolean operators or perform Boolean search with this engine? What kinds of usual Boolean operators it has? Which kind of proximity operators it uses? Does it provide a form-based search for users unprepared to work with complex boolean expressions? We have discussed all that (and a number of other issues) and now you should be able to apply your knowledge for thinking in a real world context. In addition to that, please, specify what other conditions you can specify in your search request to fine-tune the matching. Does the system enable you to restrict the search to a subset of documents or to consider for matching only a specific areas of documents?