COS 597A:
Principles of
Database and Information Systems

Information Retrieval

Traditional *database system*

- Large integrated collection of data
- Uniform access/modifcation mechanisms
- Model of data organization

Information retrieval system?

- Large ~~integrated~~ collection of information objects
- Query language(s?)
- Model of information object satisfying query

Information Retrieval

- Have collection of information "objects":
  - Text documents, Video
  - Images, 3D models
  - Audio, …
- User wants information from collection: information need
- User formulates need as a "query"
- System finds objects that "satisfy" query
  - "matches"
- System presents objects to user in "useful form"
- User determines which objects from among those presented are relevant
  - relevant = satisfy information need

Information Retrieval Issues

- Model insufficient for exact retrieval*
  - no structure imposed on info. by search system
    - no real scheme
- Query rarely exactly capture user need*
- Best matches versus all matches*
  - Too many matches to present to user
  - What and how present to user?

* not a database system

➢ algorithms for finding and scoring matches judged by effectiveness
  - rarely single correct result*

Think first about text documents

- Early digital searches – digital card catalog:
  - subject classifications, keywords
- "Full text" : words + English structure
  - No "meta-structure"
- Classic study
  - Information retrieval as old a databases
  - Gerald Salton SMART project 1960's
  - Web and large digital collections gave new "life"
- Lots of scaling since then, but models still helpful

Modeling documents

- *Document* is
  - Set of terms
  - Bag of terms
    duplicates
  - Sequence of terms
- Terms refer to *distinct words or other tokens*
  - numbers, …

7

Modeling: queries

- *Query*
  - Basic query is one term
  - Multi-term query is
    - List of terms
      - OR model: *some* terms
      - AND model: *all* terms
    - Boolean combination of terms
    - Other constraints?
- Each search engine has own query language
  - similar enough that don't need manual
  - semantics not completely clear
    - defined by search algorithm

8

Modeling: "*satisfying*"

- What determines if document satisfies query?
- That depends ….
  - Document model
  - Query model
- *START SIMPLE*
  - *better understanding*
  - *Use components of simple model later*

9

(pure) Boolean Model of IR

- Document: *set* of terms
- Query: boolean expression over terms
- Satisfying:
  - Doc. evaluates to "true" on single-term query if contains term
  - Evaluate doc. on expression query as you would any Boolean expression
  - doc satisfies query if evals to true on query

10

Boolean Model example

Doc 1: "Computers have brought the world to our fingertips. We will try to understand at a basic level the science -- old and new -- underlying this new Computational Universe. Our quest takes us on a broad sweep of scientific knowledge and related technologies… Ultimately, this study makes us look anew at ourselves -- our genome; language; music; "knowledge"; and, above all, the mystery of our intelligence. (cos 116 description)

Doc 2: "An introduction to computer science in the context of scientific, engineering, and commercial applications. The goal of the course is to teach basic principles and practical issues, while at the same time preparing students to use computers effectively for applications in computer science …" (cos 126 description)

Query:	(principles	OR knowledge)	AND (science	AND NOT(engineering))	
Doc 2:	1	0	1	1	FALSE

11

Boolean Model example

Doc 1: "Computers have brought the world to our fingertips. We will try to understand at a basic level the science -- old and new -- underlying this new Computational Universe. Our quest takes us on a broad sweep of scientific knowledge and related technologies… Ultimately, this study makes us look anew at ourselves -- our genome; language; music; "knowledge"; and, above all, the mystery of our intelligence. (cos 116 description)

Doc 2: "An introduction to computer science in the context of scientific, engineering, and commercial applications. The goal of the course is to teach basic principles and practical issues, while at the same time preparing students to use computers effectively for applications in computer science …" (cos 126 description)

Query: (principles OR knowledge) AND (science AND NOT(engineering))
Doc 1:

12

Boolean Model example

Doc 1: "Computers have brought the world to our fingertips. We will try to understand at a basic level the science -- old and new -- underlying this new Computational Universe. Our quest takes us on a broad sweep of scientific knowledge and related technologies… Ultimately, this study makes us look anew at ourselves -- our genome; language; music; "knowledge"; and, above all, the mystery of our intelligence. (cos 116 description)

Doc 2: "An introduction to computer science in the context of scientific, engineering, and commercial applications. The goal of the course is to teach basic principles and practical issues, while at the same time preparing students to use computers effectively for applications in computer science …" (cos 126 description)

Query: (principles OR knowledge) AND (science AND NOT(engineering))

Doc 1: 0 1 1 0 TRUE

13

(pure) Boolean Model of IR how "*present results in useful form*"

- most basic: give list
- meaning of order of list? => RANKING?
- There is no ranking algorithm in pure Boolean model
  - Ideas for making one without changing semantics of "satisfy"?

14

Next Model: Vector Model

- Document: *bag* of terms
- Query: *set* of terms
- Satisfying:
  - Each document is scored as to the degree it satisfies query (non-negative real number)
  - doc satisfies query if its score is >0
  - Documents are returned in sorted list decreasing by score:
    - Include only non-zero scores
    - Include only highest *n* documents, some *n*

15

How compute score?

1. There is a *dictionary* (aka *lexicon*) of all terms, numbering *t* in all
   - Number the terms 1, …, *t*
2. Rep. each document as a *t*-dimensional vector
   - The i^{th} entry of the vector is the *weight* (importance of) term *i* in the document
3. A query is a *t*-dimensional vector
   - The i^{th} entry of the vector is the *weight* (importance of) term *i* in the query
4. Calculate a vector function of the document vector and the query vector to get the score of the document with respect to the query.

16

Vector function choices

1. Measure the distance between the vectors:

$$\text{Dist}(\boldsymbol{d},\boldsymbol{q}) = \sqrt{\left(\Sigma^t_{i=1}(d_i - q_i)^2\right)}$$

   - Is *dissimilarity* measure
   - Not normalized: Dist ranges [0, inf.)
   - Fix: use $e^{-\text{Dist}}$ with range (0,1]
   - Is it the right sense of difference?

2. Measure the angle between the vectors:

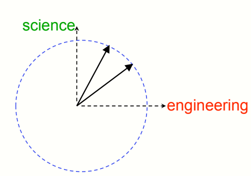
$$\text{Dot product: } \boldsymbol{d}\bullet\boldsymbol{q} = \Sigma^t_{i=1}(d_i * q_i)$$

   - Is *similarity* measure
   - Not normalized: Dist ranges (-inf., inf.)
   - Fix: use normalized dot product (cosine), range [-1,1] $(\boldsymbol{d}\bullet\boldsymbol{q}) / (|\boldsymbol{d}|*|\boldsymbol{q}|)$ where $|\boldsymbol{v}| = \sqrt{\Sigma^t_{i=1}(v_i^2)}$
   - In practice vector components are non-negative so range is [0,1]

17

Normalizing vectors

- If use unit vectors, $\boldsymbol{d} / |\boldsymbol{d}|$ and $\boldsymbol{v} / |\boldsymbol{v}|$ some of issues go away



18

Vector model: Observations

- Have matrix of terms by documents
  ⇒Can use linear algebra
- Queries and documents are the same
  ⇒Can compare documents same way
  - Clustering documents
  - Similarity search
- Document with only some of query terms can score higher than document with all query terms

19

How compute weights

- Vector model *could* have weights assigned by human intervention
  - User setting query weights might make sense
    - User decides importance of terms in own search
  - Someone setting document weights makes no sense
    - Huge number documents – billions
- Use model of documents and compute weights
  - classic model bag of terms

20

Some choices for weights

- 0/1 occur/not occur
  - problems?
- term frequency
  - longer docs versus shorter?
    - normalizing helps
  - relative frequency w.r.t other terms?
- weighted term frequence
  - account for frequency of terms in collection
  - can weight for special importance
    - e.g. in title of document - uses some ***structure*** of doc.

21

Classic weight calculation

- General notation:
  - w_{jd} is the weight of term *j* in document *d*
  - $freq_{jd}$ is the # of times term *j* appears in doc *d*
  - n_j = # docs containing term j
  - N = number of docs in collection
- Classic *tf-idf* definition of weight:

  $w_{jd} = freq_{jd} * \log(N/n_j)$

tf-idf is *"term frequency inverse document frequency"*

22

Weight of **query** components?

- Set (list) of terms, ***some choices***:
  1. w_{jq} = 0 or 1
  2. $w_{jq} = freq_{jq} * \log(N/n_j)$
     $= 0 \text{ or } \log(N/n_j)$
- Bag of terms
  - Analyze like document
    Some queries are prose expressions of ***information need***

23

Query models advantages

- Boolean
  - No ranking in pure
  + Get power of Boolean Algebra:
    expressivenss and optimize query forms
- Vector
  + Query and document look the same
  + Power of linear algebra
  - No requirement all terms present in pure

Other models and variations
probabilistic

24

Start to enhance model

- Properties of terms within documents?
- Extra information from HTML mark-up?

General Model

- Document: sequence of occurences of terms + attributes
- Query: sequence of terms
  - Can make more complicated
- Docs satisfying query: in current search engines, documents "containing" all terms
- Ranking: wide open function of document and terms
  - more and more sophisticated
  - examples
    - synonyms
    - disambiguation

Data Structure for Collection

- for each document, keep list of:
  - terms appearing
    - aggregate properties of term
      e.g. frequency
    - positions at which each term occurs
      - attributes for each occurrence of term
- keep summary information for documents

Data Structure for Collection: Invert

- for each term, keep list of:
  - documents in which it appears
    - positions at which it occurs in each doc.
      - attributes for each occurrence
- keep summary information for documents
- keep summary information for terms

Inverted Index for Collection

- for each term, keep POSTINGS LIST of:
  - each document in which it appears
    - each position at which it occurs in doc.
      - attributes for each occurrence

  POSTING

- Core structure used by query evaluation and document ranking algorithms

Index structure

```
term1:(doc ID (position, attributes)
              (position, attributes),
               …
              (position, attributes)   )
      (doc ID (position, attributes)
              (position, attributes),
               …
              (position, attributes)   )
   …
term2:(doc ID (position, attributes)
              (position, attributes),
               …
              (position, attributes)   )
   …
```

Using Web structure in IR

31

Hypertext

- document or part of document links to other parts or other documents
  - construct documents of interrelated pieces
  - relate documents to each other
- pre-dates Web
- Web "killer app."

32

How use links to improve information search

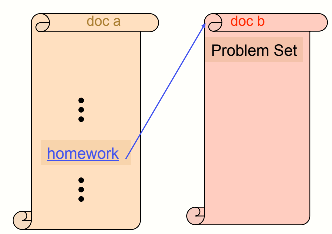
1. use anchor text (HTML)
   - anchor text labels link
   - include anchor text as text of ***document pointed to***
   - may expand vocabulary of document
   - weight?

- Similarly can use words in URL

33

Using anchor text

"homework" may not occur in *content* of doc b



List of terms in doc b:

⋮
homework: *anchor*
⋮
problem: *title* 1
⋮
set: *title* 2
⋮

34

How use links to improve information search?

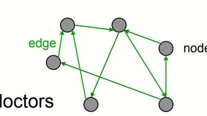
2. use structure to compute score for ranking

35

Goal

- Intuition: when Web page points to another Web page, it confers status/authority/popularity to that page
- Find a measure that captures intuition

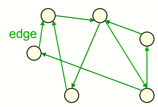
- Not just web linking
  - Citations in books, articles
  - Doctors referring to other doctors



36

Goal

- Given a directed graph with *n* nodes
- Assign each node a score that represents its importance in structure
- Most obvious: indegree
  higher indegree => better node
  - Doesn't work well



- We will look at most widely known:
  L. Page and S. Brin's (Google's) PageRank

37

PageRank

- Algorithm that gave Google the leap in quality
- Used link structure between pages in fundamental way to score pages
  - link structure centerpiece of scoring
- published
  Page, Larry and Sergey Brin, R. Motwani, T. Winograd, *The PageRank Citation Ranking: Bringing Order to the Web,* Stanford Digital Library Technologies Project TR, Jan. 1998.

38

Conferring importance

Core ideas:

- A node should confer some of its importance to the nodes to which it points
  - If a node is important, the nodes it links to should be important
- A node should not transfer more importance than it has

39

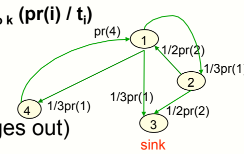
PageRank: Attempt 1

Refer to nodes by numbers 1, … , *n* (arbitrary numbering)
Let t_i denote the number of edges out of *node i* (outdegree)

Define

$$pr_{new}(k) = \sum_{i \text{ with edge from } i \text{ to } k} (pr(i) / t_i)$$

Iterate until converges



Problems

- Sinks (nodes with no edges out)
- Cyclic behavior

40

PageRank: Attempt 2

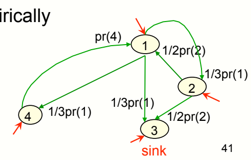
Random walk model

- Attempt 1 gives movement from node to linked neighbor with probability 1/outdegree
- Add random jump to any node

$$pr_{new}(k) = \alpha/n + (1-\alpha)\sum_{i \text{ with edge from } i \text{ to } k} (pr(i) / t_i)$$

- α parameter chosen empirically

- Helps break cycles
- Escape from sinks



41

Normalized?

- Would like $\sum_{1\le k\le n} (pr(k)) = 1$
- Consider $\sum_{1\le k\le n} (pr_{new}(k))$

$$= \sum_{1\le k\le n}\left(\alpha/n + (1-\alpha)\sum_{i \text{ with edge from } i \text{ to } k}(pr(i) / t_i) \right)$$

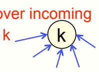
$$= \sum_{1\le k\le n}(\alpha/n)+\sum_{1\le k\le n}\left((1-\alpha)\sum_{i \text{ with edge from } i \text{ to } k}(pr(i) / t_i)\right) \; *$$

$$= \alpha + (1-\alpha)\sum_{1\le k\le n} \sum_{i \text{ with edge from } i \text{ to } k}(pr(i) / t_i)$$

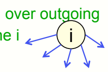
$$= \alpha + (1-\alpha)\sum_{1\le i\le n}\sum_{k \text{ with edge from } i \text{ to } k}(pr(i) / t_i) \; *$$

$$= \alpha + (1-\alpha)\sum_{i \text{ with edge from } i} pr(i))$$

*inner sum $\sum_i$ over incoming edges for one k



*inner sum $\sum_k$ over outgoing edges for one i

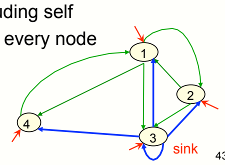


42

Problem for desired normalization

- Have
  $\sum_{1 \le k \le n} (pr_{new}(k)) = \alpha + (1-\alpha)\sum_{i \text{ with edge from } i} pr(i))$
- Missing **pr(i)** for nodes with no edges from them
  – sinks!
- Solution: add n edges out of every sink
  – Edge to every node including self
  – Gives 1/n contribution to every node

Gives desired normalization:
If $\sum_{1 \le k \le n} (pr_{initial}(k)) = 1$
then $\sum_{1 \le k \le n} (pr(k)) = 1$



43

Calculation

- Simple iterative calculation
  - Initialize $pr_{initial}(k) = 1/n$ for each node k
    - so $\sum_{1 \le k \le n} (pr_{initial}(k)) = 1$
  - $pr_{new}(k) = \alpha/n + (1-\alpha) (\sum_{i \text{ a sink}} pr(i) / n)$
    $+ (1-\alpha) \sum_{i \text{ with edge from } i \text{ to } k}(pr(i) / t_i)$
  - Choose α
    - No single best value
    - Page and Brin originally used α=.15
- Converges
  – Has necessary mathematical properties
  – In practice, choose convergence criterion
    - Stops iteration

44

Storage

$pr_{new}(k) = \alpha/n + (1-\alpha) (\sum_{i \text{ a sink}} pr(i) / n)$
$+ (1-\alpha) \sum_{i \text{ with edge from } i \text{ to } k}(pr(i) / t_i)$

- Pulled out sinks to expose storage needs
  – set of sinks
  – set of edges (i,k)
  – values of pr(i) all i, $1 \le i \le n$ - dynamic
- Social graphs, incl. Web usually sparse
  – number edges << n^2 for n nodes
  – n huge
- What access methods need?

45

PageRank Observations

- PageRank can be calculated for *any* graph
- Google calculates on entire Web graph
- Huge calculation for Web graph
  – precomputed
  – 1998 Google:
    - 52 iterations for 322 million links
    - 45 iterations for 161 million links
- PageRank must be combined with query-based scoring for final ranking
  – Many variations
  – What Google exactly does secret
  – Can make some guesses by results

46

Web-based scoring

- PageRank one of class of algorithms
- Second most well-known: HITS
  – designed at same time as PageRank by Jon Kleinberg while at IBM Almaden Research Center
  – Same general goal as PageRank
  – Distinguishes **2 kinds of nodes**
    - **Hubs:** resource pages
      – **Point to many authorities**
    - **Authorities:** good information pages
      – **Pointed to by many hubs**
- **Exploiting Web Structure an important part of information access and analysis**

How use links to improve information search?

3. include more objects to rank

48

Use of HITS

original use **after** find Web pages satisfying query:

1. Retrieve documents satisfy query and rank by term-based techniques
2. Keep top *c* documents: root set of nodes
   – *c* a chosen constant - tunable
3. Make base set:
   a) Root set
   b) *Plus* nodes pointed to by nodes of root set
   c) *Plus* nodes pointing to nodes of root set
4. Make base graph: base set plus edges from Web graph between these nodes
5. Apply HITS to base graph

using links to expand matches!

49

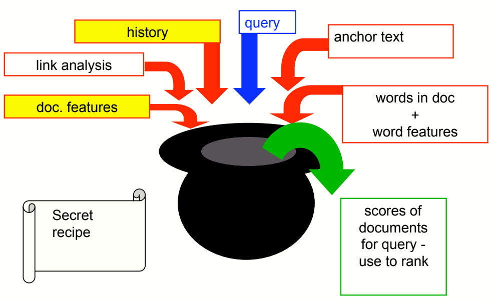
Summary: How use links to improve information search?

- use anchor text (HTML)
  – include anchor text as text of document pointed to
- use structure to compute score for ranking
  – PageRank, HITS
- include more objects to rank
  – saw in use of HITS

❖can deal with objects of mixed types
  – images, PDF, …

50

Ranking documents w.r.t. query



51

Searching non-text information objects without text annotations

(not covered in class)

52

Ways to query for something

1. Query by category/ theme
   – easiest - work done ahead of time
2. Query by describing content
   – text-based query
   – text-based retrieval?
3. Query by example
   – "similar to"
   – imprecise example - sketch

- query text docs and non-text objects with 2
- music, images dominent applications of 3

53

Query by example

- What want?
  – objects "similar to" example object
  – precise vs imprecise example
    eg. photo vs sketch
- How represent objects?
  – features of a class of objects (e.g. image)
  – how compare features?
  – what data structures?
  – what computational methods?
- Issues
  – large number of objects
  – accuracy of representation
  – large size of representation
  – complexity of computations

tradeoffs
tradeoffs

54

Features

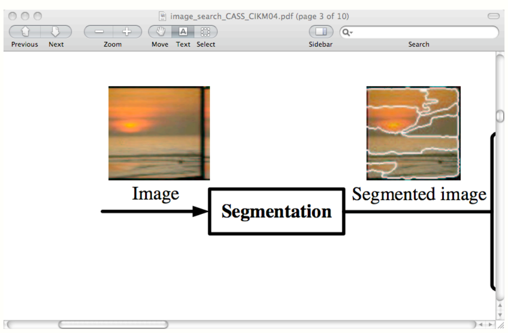
- typically vector of numbers characterizing object representation
- "similar to" ≡ close in vector space
  - threshold
  - Euclidean distance?
  - other choices for distance metric

55

Example: content-based image search one method

- region-based features of images
- query processed in same way as collection
- space-conscious: use bit vectors
- levels of representation:
  - store bit vector for each region
  - store bit vector for each image
- get close candidates: compare image bit vectors
- compare top k candidates using region bit vectors

56



57

Processing images of collection & query

- segment into homogeneous regions
  - 14 dimensional feature vectors
- threshold and transform
  - high-dimensional bit vectors - store
  - XOR for distance between regions
- build image feature vector
  - n region bit-vectors + weights ⇒ 1 image feature vector
  - L_1 distance between feature vectors
- transform image vector
  - one high-dimensional bit vector for image - store

58

Observations: region based

- Example of one regional method
  - active research area!
- Example of common techiques & goals
  - aggregate/average features
  - sample
  - course screening followed by more accurate
  - reduce dimension
  - reduce complexity of distance metric
  - reduce space
- Part of larger project - multiple media
  - CASS, Princeton, 2004

59

Image search: Commercial search engines

- Use everything you can afford to use
- Text still king!?

60