Information Retrieval Methods

Helena Ahonen-Myka

Spring 2007, part 13

Searching the Web

In this part

- Searching the Web
  - Challenges of web searching
  - Architecture of search engines
  - Crawling the web
  - Queries
  - Ranking
  - Trends
- Some final issues (exam, etc,)

2

Challenges of Web search

- Distributed data
  - Data spans over many computers and platforms
  - Available bandwidth and reliability on the network interconnections varies widely
- High percentage of volatile data
  - New computers/sites/pages can be added and removed easily
  - We also have dangling links etc. when domain or file names change or disappear
- Large volume
  - Scaling issues difficult to cope with

3

Challenges of Web search

- Unstructured and redundant data
  - No conceptual structure/organization
  - HTML pages are only semi-structured
  - Much data is repeated (copies/mirrors)
- Quality of data
  - There is no editorial process → data can be false, invalid, poorly written, with many typos
- Heterogeneous data
  - Multiple media types, multiple formats
  - Different languages, different alphabets

4

Search engine architecture

- Most search engines use a crawler-indexer architecture
  - Crawlers are programs that traverse Web sending new or updated pages to a server where they are indexed
    - a crawler runs on a local system and sends requests to remote Web servers
  - The index is used in a centralized fashion to answer queries submitted from different places in the Web

5

Search engine architecture

- Although the administration of crawling and indexing is centralized, there can be physically a large number of specialized servers
  - E.g. Google has 450 000 servers located in clusters in cities around the world
    - Name servers, web servers, crawling servers, index servers, document servers, ad servers, spelling servers
    - (The power required for the servers: 20 MW; could cost on the order of US$2 millions/month in electricity charges...)

6

Crawling the web

- *Crawlers, robots, spiders, wanderers, walkers, knowbots,…*
- A *web crawler* is a program which browses the Web in a methodical, automated manner
- Web crawlers are mainly used to create a copy of all the visited pages for later processing (indexing) by a search engine
- Crawlers can also be used for automating maintenance tasks on a Web site, such as checking links or validating HTML code

7

Crawling the web

- Crawling is started with a set of URLs (*seeds*)
  - Users may be allowed to submit URLs
  - Popular URLs can be used
- As the crawler visits a URL, it identifies all the hyperlinks in the page and adds them to the list of URLs to visit (the *crawl frontier*)
- URLs from the frontier are recursively visited according to a set of policies

8

Problems with crawlers

- Web servers receive requests from different crawlers, increasing their load
- Web traffic increases, because crawlers retrieve entire objects, but most of the content is discarded
- Information is gathered independently by each crawler, without coordination between all the search engines

9

Crawling policies

- The behavior of a Web crawler is the outcome of a combination of policies
  - A *selection policy*: which pages to download
  - A *re-visit policy*: when to check for changes to the pages
  - A *politeness policy*: how to avoid overloading websites
  - A *parallelization policy*: how to coordinate distributed web crawlers

10

Crawling policies

- Selection policies
  - Breadth-first or depth-first fashion
  - The entire site may be crawled, just a sample of pages, or pages up to a certain depth
  - Submitted start URLs are crawled faster than non-submitted URLs (which have to be detected first)
- Re-visit policies
  - Re-visit all pages in the collection with the same frequency
  - Re-visit more often the pages that change more frequently
  - Popular pages may be re-visited more frequently

11

Crawling policies

- Politeness policies
  - The robots exclusion protocol: the web site administrators can indicate in file *robots.txt* which parts of the site should not be accessed by crawlers
  - Enough delay between requests, e.g. 1-20s
  - Web crawlers typically identify themselves to a Web server (by using the User-agent field of an HTTP request)
    - Web site administrator can contact the owner of the web crawler, if there are problems

12

Crawling policies

- Parallelization policies
  - A parallel crawler runs multiple processes in parallel
  - The goal is to maximize download rate while avoiding repeated downloads of the same page
    - The same URL can be found by two different crawling processes

13

Crawling the web

- Some part of the web cannot be indexed
  - E.g. dynamically generated content (*deep/hidden/invisible web*) and password protected pages
  - The crawler may try to retrieve indefinite number of pages from a database with different input values → *a crawler trap*
  - Indexing may succeed in specific cases when the syntax and semantics of the query parameters are known, e.g. with dictionaries
    - Or can these resources be used directly in query processing phase?

14

Queries

- Usually some combination of Boolean and term set queries is used
- E.g. Google:
  - Default operation: multiple search terms are processed as and AND operation:
    - apple orange fruit salad → apple AND orange AND fruit AND salad
  - Limited support for Boolean queries, e.g. no nesting
  - No truncation

15

Queries

- The user does not always know how the query is modified
- E.g. Google automatic stemming: plural/singular forms, synonyms, grammatical variants are added to (some) terms
  - Query: *socially responsible investing*
  - Matches also words: *investment, SRI,…*
  - Operator + can be used in front of a term to require that no expansion should be done

16

Queries (Google)

- Proximity searching
  - Phrases can be enclosed in double quotes
  - Google also detects phrase matches even when the quotes are not used and usually ranks phrase matches higher
- No case sensitive searching: using either lower or upper case results in the same hits
- Stop words
  - Stop words within a phrase will automatically be searched
  - Other stopwords can be searched with the + sign
  - If only stopwords, no + signs needed

17

Ranking

- Most search engines use variations of the Boolean and/or vector model
- Primary condition: result documents contain search terms
  - Problem: usually too many results → other conditions needed
- Ranking algorithms can use hyperlink information
  - Hyperlinks encode latent human judgment
  - The number of links that point to a page provides a measure of its popularity and quality

18

Ranking

- Ranking algorithms
  - HITS (Hypertext Induced Topic Search) by Kleinberg, 1998
  - PageRank by Page and Brin (Google), 1998

19

Ranking: HITS

- Ranking scheme depends on the query
  - the initial answer set = the documents found using search terms
- The initial answer set is expanded to the set of pages S that
  - point to pages in the answer or
  - are pointed to by pages in the answer
- Pages that have many links pointing to them in S are called *authorities* (= should have relevant content)
- Pages that have many outgoing links are called *hubs* (= should point to similar content)

20

Ranking: HITS

- a positive two-way feedback exists
  - better authority pages have incoming edges from good hubs
  - better hub pages have outgoing edges to good authorities

21

Ranking: HITS

- Let *H(p)* = the hub value of page *p*
- Let *A(p)* = the authority value of page *p*
- The following equations are satisfied for all pages *p*:

$$H(p) = \sum_{u \in S | p \to u} A(u) \qquad A(p) = \sum_{v \in S | v \to p} H(v)$$

- The values can be determined through an iterative algorithm

22

Ranking: HITS

- Depending on the needs of the user, good authorities or hubs (or both) are returned
- It is possible that a result document does not contain search words
  - but pages that link to the page contain them
  - e.g. the home page of Toyota may not tell that Toyota is a car manufacturer

23

Ranking: PageRank

- Part of Google's ranking algorithm
- Simulates a user navigating randomly in the Web
  - the user jumps to a random page with probability *q* or
  - follows a random hyperlink (on the current page) with probability *1-q*
- This user never goes back to a previously visited page following an already traversed hyperlink backwards
- The probability of being in each page can be computed → the value is used to estimate the quality of the page

24

Ranking: PageRank

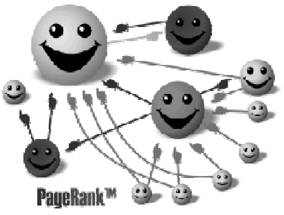
- Let *C(p)* be the number of outgoing links of page *p*
- suppose that page *a* is pointed to by pages p_1 to p_n → the PageRank, *PR(a)*, is:

$$PR(a) = q + (1-q)\sum_{i=1}^{n} PR(p_i)/C(p_i)$$

- where *q* must be set by the system (e.g. *0.15*)
- Pagerank can be computed using an iterative algorithm

25

Ranking: PageRank



26

Displaying the results: Google

- Results are ordered by relevance
  - also many other criteria (150?) are used than PageRank
- Pages are also clustered by site
  - Only two pages per site will be displayed, with the second indented
  - Others are available via the *[More results from…]* link
- Display:
  - Title, URL, a brief extract showing text near the search terms, the file size, a link to a cached copy of the page

27

Trends…?

- storing content (not just indexing)
- user-generated content
- tagging vs. hierarchies
- communities, social networks, recommendation systems
- services, not just information
- e.g. blogs, YouTube, Flickr, MySpace,…

28

Finally…

- return the project work report by Friday, 9 March at 24:00
  - if you are late we will reduce 2 points/day
- write the report in HTML and tell only the URL to Niina
- exam on Monday, 26 February at 9-12
  - in room B123

29

Course components

- Exercises: max 5 points
- Project work: max 15 points
- Exam: max 40 points

30

Exam

- Example question types (the actual exams may differ from this example!)
  - "Define": Explain (max 5 points/question, max half a page/question), e.g., *Quorum search*, or *the implementation of a proximity operator*, or *filtering and routing.*
  - "Compute": (max 12 points) *Given a document-term matrix with term frequencies, compute document similarities, similarity between a given query and documents.*
  - "Essay": Describe (max 13 points, 1-1.5 pages) a certain IR concept in full, e.g., *relevance feedback*

31

Exam

- When answering
  - a "Define" question, please be short and precise.
  - a "Compute" question, please be precise and give all stages and formulas that you use in you computations. Justify what you do.
  - an "Essay" question, try to be complete (i.e. telling all the essential aspects you know about it) without writing nonsense or trivial things.

32

Thank you!!!

- For participating!
- For good comments and discussion!

33