CmpE 473
Internet Programming

Pınar Yolum
pyolum@cmpe.boun.edu.tr

Department of
Computer Engineering
Boğaziçi University

Chapter 11
Searching the Web

Web

- Content
  - Static pages (such as HTML pages)
  - Dynamic pages (created on demand)
    - Information accessible through authorization
    - Information based on choices
- Access
  - Linked through other pages
  - Location known by owner

Search Engine

Typically contains four components:

1. Crawler: Retrieve pages from the Web
2. Indexer: Indexes the information on the retrieved pages
3. Ranker: Determines the importance of a page
4. Retrieval engine: Performs lookups

Crawler

- Start from a set of seed URLs
- For all pages in the set
  1. Retrieve the Web page with one URL
  2. Find other links from that URL
  3. Add them to the set
- Little human intervention
- What can you not find with crawlers?

Crawler Challenges

- Memory management
  - A URL is roughly 16 bytes
  - A billion URLs take 16GB!
  - Main memory vs. secondary storage
- Graph traversal
  - Which URL to follow next?
  - BFS, DFS, Reputation-based
- Link extracting
  - Parse the Web page to find the links
  - Different types of files, HTML, Word, …

Crawler Challenges

- Robot Exclusion Standard
  - Agreement between the Webmaster and the crawler
  - Pages that should not be crawled are identified in robots.txt in the root directory
  - Example: http://www.bbc.co.uk/robots.txt
- For each user agent, specify which directories are disallowed
- Ex: User-agent: * Disallow: /cgi-bin
- Ex: User-agent: Googlebot Disallow: /*.doc$
- Insert inside Web page
  - <META NAME="ROBOTS" CONTENT="NOINDEX, NOFOLLOW">

Crawler Performance

- Speed
  - Measured in crawled pages/sec
  - Over 112pages/sec using 2 computers
  - Higher for commercial search engines
- Coverage
  - Measured in number of Web servers hit or number of visited pages
- Quality
  - Measured in ?

Indexer

- Stores the content from the crawled pages
- Extracts words for the retrieval engine to look up pages
- Index roughly 30% of the corpus
  - Indexing 1 billion pages of 10 K
  - Results in a 3TB index!

Indexer

- Document preprocessor
  - Find the words to index a Web page
  - Eliminate stop words such as the, and, I
  - Use techniques from IR to decide on relevant words (such as word frequency)
- Use inverted file structures
  - Forward (Document) index: Assign a unique id to a page and relate to all index terms
  - Dictionary: Sorted list of index terms with pointers to inverted list; contain number of occurrences of a term in a file
  - Inverted index: Keeps pointers from terms to all the documents that contain the terms

Ranker

- Lookup in the index can return hundreds of results
- Connectivity-based
  - Which pages are more linked?
- Content-based
  - Number of matched terms
  - Location of terms in the document
  - Frequency of terms

Retrieval Engine

- Takes user's query and translates that to different queries for the indexer
- Merges results of different queries
- Can potentially do more clever lookups
- Example
  - Query: Car
  - Can try to search for synonyms or translations in different languages

PageRank

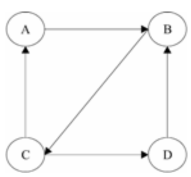
- Heuristic used to rank Web pages
- Assigns a grade to Web pages based on their authoritativeness
- If an authoritative Web page *A* links to page *B*, then B is authoritative, too
- Initially every page's PageRank value is 1
- Then start calculations using the following recursive formula
- Until the difference between two successive calculations is small
- Then normalize

PageRank

- P(i): PageRank of i; N(j): Neighbors of j; K(i): Pages that point to page i; d: Damping factor=0.85

$$P(i) = d \sum_{j \in K_i} \frac{P(j)}{|N_j|} + (1-d)$$

- Example:



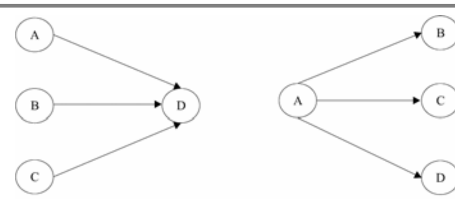
Topic Neighborhood

- Based on a query, compute
  – Starting set of pages [≤200, in practice]
- Add pages
  – Linked from starting set
  – Linking to starting set [≤ 50, in practice]
- This is the neighborhood graph **G=(V, E)** for a given query
  – Presumably contains the most relevant pages to the original query

HITS

- Identify some Web pages as an authority and some pages as hubs
- Hubs point to good authorities (know the right pages)
- Authorities are pointed to by good hubs (their authority is acknowledged)
- For each node calculate the hub and authority value
- Start by initializing both to 1/n

HITS



(a) Node D is an authority (c) Node A is a hub

$$a_i = \sum_{j \in K_i} h_j \qquad (1)$$

$$h_i = \sum_{j \in N_i} a_j \qquad (2)$$

PageRank vs. HITS

- PageRank is query-independent; HITS is query-dependent
- Both correspond to matrix computations.
- Can be unstable: changing a few links can lead to quite different rankings.
- HITS suffers from topic drift where the best hubs and authorities may not be on the original topic.
- PageRank doesn't handle pages with no out-edges very well, because they decrease the PageRank overall.

Deep Web

- Current approaches
  - Work best with text (in txt or html)
  - Handle text in other formats (pdf, ppt, doc)
  - Ignore multimedia data
- Most data accessible over the Web is not in static pages
  - Invisible to conventional search engines
  - Estimates of about 95% of the Web
- Trawling to get data behind forms
  - Sometimes Google returns results from ACM database

Reputation Approaches: 1

How to evaluate and select in open settings

- Commercially applied, e.g., gittigidiyor.com
- After every transaction, the participants get an opportunity to rate each other on a small, fixed scale (-1, 0, +1) plus text.
- Ratings are revealed individually and in aggregation to others.

Role of Reputation Agencies

The agency (or market) is the authority that

- Authenticates users
- Records ratings
- Aggregates and reveals ratings
- Owns ratings: to capture participants
- Provides the conceptual schema for
  - Capturing ratings (typically a number and text)
  - How to aggregate them
  - How to decay them over time

Reputation Approaches: Limitations

- Assumes that identities of participants don't change—those with a good record would wish to preserve it.
- Ratings may be given in collusion.
- Ratings may be given in retaliation.
- Users of ratings don't know the parties who gave the rating.
- Ratings, once given, may be revealed to all.

Endorsements

- Unlike reputation, endorsements are
  - From known party
  - To known party
  - Each party can decide how to aggregate them
- Like reputation, an endorsement is based on a conceptual model and may include a rating
- An endorsement matters if it is from a trusted party
- How do you decide whom to trust?
  - Hard-coded
  - Organizational or social factors
  - More endorsements: chains of endorsements

Certificates

Endorsements limited to assertions of identity.

- How to obtain a certificate
  - Principal P contacts Registration Authority
  - RA authenticates P and forwards to CA, which
    - Issues key pair for P
    - Signs certificate with P's name, public key (X.509)
    - Publishes certificate in a repository
  - P can use its private key and certificate
- How do you trust the RA and CA?

Recommender Systems

- Motivation: target customers better to cross-sell, up-sell
- Typically, centralized approaches for making recommendations
  - Collaborative filtering: find things liked by people similar to you (e.g., amazon.com)
  - Matchmaking: cluster parties with similar interests learned by asking them (e.g., the Intellectual Matchmaker)

Memory-Based Approaches

- Consider all users directly: prediction for active user is weighted sum of votes by others, where the weight corresponds to similarity or correlation between active user and each of the others.
  - GroupLens (led to Net Perceptions) uses Pearson correlation.
  - Can use vector similarity instead.
  - How to correlate when users overlap on few services?
  - Weight in favor of less commonly used services.

Model-Based Approaches

Build a model from the users; then use the model for predictions.

- Cluster the users and then place active user in one of the clusters.
- Build a structured representation, e.g., a decision tree.
  - Decision nodes correspond to different votes.
  - For example, a tree may represent a model that users who like coffee and vanilla don't like chocolate: use this model to predict a specific user's preferences.

Collaborative Filtering

- Predicting an *active user's* vote (rating for an item) based on votes by others and the active user's votes *elsewhere*.
  - *Explicit:* fill a ratings form (cumbersome)
  - *Implicit:* purchase history, browsing, return visits
- Data is always incomplete
  - Biased, generally positively (non-null values are positives)

CF Algorithms: Prediction

- I = items; I_i = items rated by user i; v_{ij} = rating given by user i to item j; $\overline{v_i}$ = average vote by user i.

$$\overline{v_i} = \frac{1}{|I_i|}\Sigma_{j \in I_i} v_{ij}$$

- Predicted vote of active user (a) for item j is (n=#users; weights w_{ai} reflect similarity between users a and i):

$$p_{aj} = \overline{v_a} + \frac{\Sigma_{i=1}^{n} w_{ai}(v_{ij} - \overline{v_i})}{\Sigma_{i=1}^{n} |w_{ai}|}$$

CF Algorithms: Weights Defined

- In GroupLens, the weights are given by correlation of ratings:

$$w_{ai} = \frac{\Sigma_j (v_{aj} - \overline{v_a})(v_{ij} - \overline{v_i})}{\sqrt{\Sigma_j (v_{aj} - \overline{v_a})^2 \Sigma_j (v_{ij} - \overline{v_i})^2}}$$

Recommending Products vs. Services

- Products (by a product vendor)
  - The recommender is the provider
  - Votes are known to recommender
  - Votes are given prior to usage (buying)
  - Repetition is less likely (buy the same book)
- Services (by a service registry)
  - The recommender is not the provider
  - Votes are not necessarily known to recommender
  - Votes are given after usage
  - Repetition can occur but not known to registry

Motivation for Referrals

The above approaches artificially separate three aspects of service discovery and selection:

- *Discovery* via lookup and network navigation.
- *Ratings* (as in reputation calculations).
- *Recommendation* through similarity of needs and preferences.

Service Communities

- Each principal
  - Provides services to others
  - Exploits services provided by others
  - Has a personal agent
- The agents assist their users in
  - Evaluating the services and referrals provided by others
  - Maintaining contact lists
  - Deciding whom to contact for a service

Why Referral Systems?

- Collaborative filtering: aggregate results---no one to trust (or blame)
- Opinions of those you can rate and who have similar needs might be more trustworthy
  - Reveal honest ratings to trustworthy peers
  - Trust ratings obtained from trustworthy peers