Web Search

Web search vs classical IR

In classical IR, the collection is relatively static. The goal is to retrieve documents with content that is relevant to the user's information need.

Classic measures of relevance tend to ignore both *context* and *individuals*.

Web search vs classical IR

In web searching, corpus contains static and dynamic information. Goal is to retrieve high quality results that are relevant to current need.

Need may be:

informational
navigational
transactional

Newer problems emerge with respect to web search:

- distributed data
- volatile data
- large volumes - scaling issues
- redundancy (c. 30-40% (near) duplicates)
- quality (100s millions of pages of spam)
- diversity (many languages, encodings)
- complex graph structure/topology

Web Search users

Tend to make ill-defined queries:

- short queries
- imprecise terms
- sub-optimal syntax
- low effort

wide variance in terms of needs, expectations, knowledge and bandwidth

specific behaviour – c.85% people do not look at next screen; c.80% do not modify query; instead follow links

Evolution of Web Search

1st generation: use only “on page” information - word frequency etc.

2nd generation: link analysis, click through data

3rd generation: semantic analysis, integrate multiple sources of information, context analysis (spatial, query stream, personal profiling), aiding the user (re-spelling, query refinement, query suggestion), representation of results/query/collection

Anatomy of a Search Engine

Spider (robot/crawler): builds the corpus by recursively following links.

The indexer: processes the data and indexes it (fully inverted list). Different approaches adopted with respect to stemming, phrases etc.

Query processor: - query reformulation, stemming, handling Boolean operators, finds matching documents and ranks them

1ST Generation web search engines
(1995-1997, e.g. Lycos, Excite)

Extended Boolean:

- matches: exact, prefix
- operators: AND, OR, NEAR
- fields: TITLE, URL, HOST, …

Ranking:

- *tf* like factors: term frequency in document, term frequency in title
- *idf* like factors: inverse document frequency, word count in corpus, frequency in query log, frequency in language

2ST Generation web search engines

(e.g. Google)

Biggest change was the incorporation of web link information

Related to work originally undertaken in bibliometrics and citation indexing.

Exploiting web structure has become more popular.

Examples:

WebQuery: visual browsing of web; takes query and displays results using the links between them. The degree of connectivity of a page is taken as a metric.

The Clever system: check links to identify ‘hubs' and `authority pages'

PageRank system: based on Markov chains. Probability estimates of user following a link from a set of links is estimated.

Many extensions: take into account extra factors in account. For example, time, semantic interpretation of link, etc.

3rd Generation web search engines

Semantic analysis of query

Context determination:

- spatial/geographic
- query stream analysis
- personalised

Helping User

- UI/Visualisation
- spelling
- query refinement
- query suggestion

History: citation analysis

Initial work in this area can be traced back to the domain of citation analysis

The **impact factor** of a journal = *A/B* (Garfield, 1972)

A is the number of current year citations to articles appearing in the journal during previous two years.

B is the number of articles published in the journal during previous two years.

Co-citation

If a paper cites two papers A and B, then they are related or associated.

The strength of co-citation between *A* and *B* is the number of times they are co-cited.

History: citation analysis

Measure of similarity of documents (Kessler in 1963)

The **bibliographic coupling** of two documents *A* and *B* is the number of documents cited by *both A* and *B*.

Mining the Web Link Structure

The main approach in 1st generation search engines was the indexing approach. Although useful and effective there are potentially too many links returned. How does one return the most relevant?

Usually wish to select the most “authoritative” pages. Hence the search entails identifying pages that have relevancy and quality.

In addition to content, web-pages also contain many links that connect one page to another.

This web-structure contains implicitly a large number of human annotations which can be exploited to infer notions of authority (and by extension quality).

Any link to a page, p, is a positive recommendation for that page p.

The *HITS* algorithm analyses hyper-links to identify:

(Kleinberg, 1998)

- authoritative pages (best sources)
- hubs (collections of links)

Develop algorithms to exploit implicit social organisation available in the web link structure

There exists problems with identifying authoritative pages:

- authoritative pages do not necessarily refer to themselves as such

- many links are purely for navigational purposes

- advertising links

Mutually recursive heuristics used:

a good “authority page” is one which is pointed to by a number of sources

a good “hub” is one that contains many links

HITS Algorithm

Computes hubs and authorities for a particular topic specified by a normal query.

First determines a set of relevant pages for the query called the *base* set *S*.

Analyse the link structure of the web sub-graph defined by *S* to find authority and hub pages in this set.

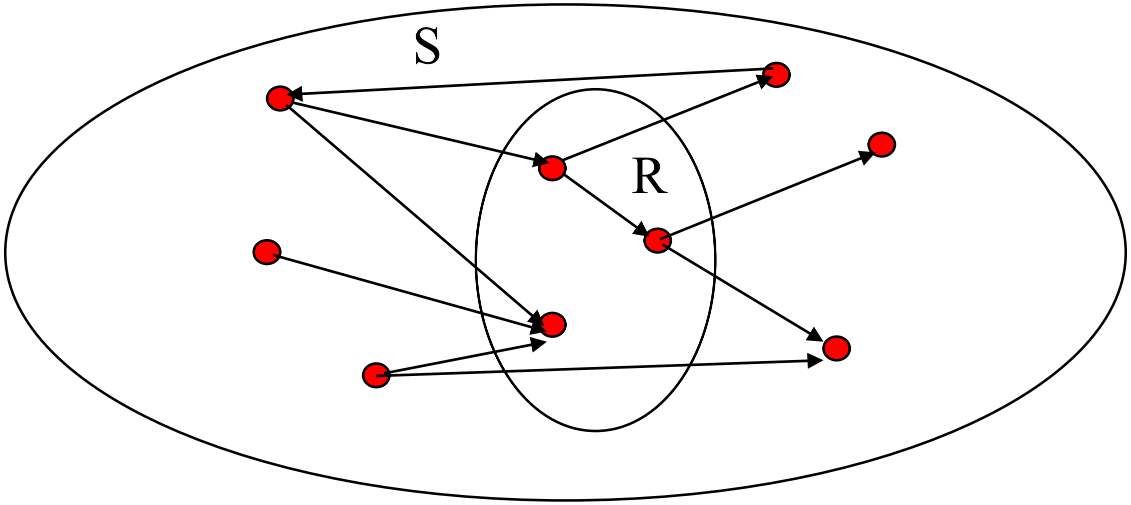
Constructing the Subgraph

For a specific query *Q*, let the set of documents returned by a standard search engine (e.g. vector space approach) be called the *root* set *R*.

Initialize *S* to *R*.

Add to *S* all pages pointed to by any page in *R*.

Add to *S* all pages that point to any page in *R*.



Iterative Algorithm

Assign to each page $p \in S$:

an authority score: a_p (vector $\boldsymbol{a}$)
a hub score: h_p (vector $\boldsymbol{h}$)

Initialise all $a_p = h_p$ to some constant value

HITS Update Rules

Authorities are pointed to by lots of good hubs:

$$a_p = \sum_{q:q \to p} h_q$$

Hubs point to lots of good authorities:

$$h_p = \sum_{q:p \to q} a_q$$

Define M to be the adjacency matrix for the subgraph defined by *S.*

$$M_{ij} = 1 \text{ for } i \in \mathrm{S}, j \in \mathrm{S} \text{ iff } i \rightarrow j$$

Can calculate authority vector, ***a***, from the matrix $M^T M$

Similarly, the hub vector, ***h***, can be calculated from the matrix $M M^T$

Other issues

Limitations of link only approach:

1. on narrowly focussed query topics, there may not be many exact references and the hubs may provide links to more general pages

2. potential drift from main topic. All links are treated as being equally important. If there is a range of topics in a hub, the focus of the search may drift

3. timeliness of recommendation is hard to identify

4. sensitivity of malicious attack

5. edges with wrong semantics

PageRank: Markov Chains

A Markov chain has two components:

- a graph/network structure; each node is called is called a state.
- a transition probability of traversing a link given that the chain is in a state.

A sequence of steps through the chain is called a *random walk*.

Random Surfer Model

Assume the web is a Markov chain.

Surfers randomly click on links, where the probability of an outlink from page A is *1/n*, where there are n outlinks from *A.*

The surfer occasionally gets *bored* and is moved to another web page (teleported), say *B,* where *B* is equally likely to be any page.

The *PageRank of a web page is the probability that the surfer will visit that page*.

Dangling Pages

A page with no outgoing links; can’t pass on rank.

Solution: Assume page has links to all web pages with equal probability.

Rank Sink

Problem: Pages in a loop accumulate rank but do not distribute it.

Solution: “Teleportation”, i.e. with a certain probability the surfer can jump to any other web page to get out of the loop.

PageRank(PR)- Definition

$$PR(W) = \frac{T}{N} + (1-T)\left(\frac{PR(W_1)}{O(W_1)} + \frac{PR(W_2)}{O(W_2)} + \ldots + \frac{PR(W_n)}{O(W_n)}\right)$$

- *W* is a web page
- *Wi* are the web pages that have a link *to W*
- *O*(W*i*) is the number of outlinks from W*i*
- *T* is the teleportation probability
- *N* is the size of the web

Efficiency

Early experiments on Google showed convergence in 52 iterations on a collection with 322 million links

Number of iterations required for convergence is empirically O(log *n*) (where *n* is the number of links)

This is quite efficient.

Other issues/Exercises - 1

What are main differences between PageRank and HITS?

Personalised Page Rank

Can bias the behaviour of page rank by changing the notion of random jumps

Instead of jumping to a random page on the web, we jump probabilistically to a page chosen from a seed set defined for a user.

Adds rank to pages of interest to user rather than random page.

Semantically/Content biased Page Rank

Page ranks treats all edges as being equally important in its random surfer model (excluding links identified as navigation and advertising links)

i.e. page rank values are distributed equally across all outgoing edges.

Extra heuristic:

Surfer is more likely to follow link relating to content of current page (or passage).

Semantically/Content biased Page Rank

The page rank values propagated from a page sum to one; standard page rank gives equal values.

We can measure a similarity between the context of a link and the linked to page. This gives a measure of semantic relatedness between pages/passages.

If users are more likely to navigate to a related page, we can assign page rank values in proportion to the relative similarity.

Temporal link analysis

Link-analysis techniques (e.g., PageRank, HITS) do not take into account associated temporal aspects of the web content

Goal is to incorporate temporal aspects (e.g., freshness, rate of change) into link-analysis techniques

Ranking based on the pages’ authority values as that value changes over time

Approach

Add annotations to the graph:

i.e. for every edge in the graph and for every vertex of the graph

maintain a set of values regarding the temporal aspects such as

- creation time
- modification times
- last modification time

Can define window of interest, freshness of edge/node, activity of edge/node

Ranking Signals?

There are many sources of evidence (or signals) that can be used to rank a page:

- content signals (BM25 and variants used)
- structural signals (anchor text etc.)
- web usage (implicit feedback, temporal context)
- link based ranking

Ranking Signals?

How best to combine signals?

Simple approach to combine PageRank (link analysis) and BM25 (content signal)

$$R(p,q) = (a)BM25(p,q) + (1-a)PR(p)$$

Ranking Signals?

More recently much attention being paid to Learning to Rank approaches:

Effectively, attempt to learn optimal way to combine all the signals

Many approaches have been adopted:

a) Learning the ranking (*NN, SVM, Bayesian networks*)
b) Learning the ranking function (Genetic Programming)

Related work: LETOR package

Evaluation?

In order to compare systems and algorithms
and to be able to guide any learning approaches,
we need some means to evaluate

Commonly used:

Prec @5, Precision @10

- impossible to obtain recall
- users tend not care about recall

Evaluation?

In IR, we have test collections and human evaluations.

In web search, can exploit click through data.

Issues – heavy tailed distribution of queries? – sufficient evidence?

Related issue: Evaluation of snippets

Other issues in web search

How to deal with duplicated data?

How to deal with near duplicates?

Query suggestions?

- diversity?
- appropriate suggestions
- predictive accuracy

Other issues

Adverserial search - the conflict between web search engines designers/creators and the ‘search engine optimisation’ community

- Recognising spam links
- augmenting link analysis algorithms to deal with such manipulation