Overview

- Massive amounts of data
  - effect on precision
  - lots of data beats better algorithms
- Link analysis
  - PageRank
  - Hubs and Authorities (HITS)
  - Anchor Text
  - Link spam

Amount of data: precision

- Staggering amount of data (Google: 20PB/day in 2008)
  - challenging but surprisingly makes some things easier
- Collection ... a (random) sample of the web
  - suppose we gathered N documents, precision@10 = 40%
  - what if we gathered 4*N documents?

Big data beats clever algorithms

- Question Answering task (e.g., IBM Watson):
  - Q: Who created the character of Scrooge?
  - A: Scrooge, introduced by Charles Dickens in "A Christmas Carol" ...
  - requires heavy linguistic analysis, lots of research in TREC
- 2002, Microsoft:
  - identify (sub)verb obj), rewrite as queries:
    - Q1: "reveal the character of Scrooge"
    - Q2: "the character of Scrooge was created by"
  - run on the web, take 500 top-ranked docs
  - extract phrase: [NP:Q1 or Q2:NP], find most frequent [NP]
  - very naive approach, ignores thousands of answer patterns...
  - doesn't matter: web is huge: guaranteed to get a match

Links between pages

- Google's description of PageRank:
  - relies on the "uniquely democratic" nature of the web
  - interprets a link from page A to page B as "a vote"
  - A → B means A thinks B is worth something
  - "wisdom of the crowds": many links mean B must be good
  - content-independent measure of quality of B
- Use as a ranking feature, combined with content
  - but not all pages that link to B are of equal importance
  - a single link from Slashdot or CNN may be worth thousands
  - how many "good" pages link to B

PageRank Algorithm

- Initialize PR(X) = 100/N
  - total number of pages in our collection
- For every page X:
  - PR(X) ← \( \sum_{Y \rightarrow X} \frac{PR(Y)}{N} \)
  - y → x contributes part of its PR to x
  - spreads PR equally among out-links
  - PR scores should sum to 100%
  - use two arrays: PR(i) → PR(i+1)
- Example:
  - PR(A) = 0.18 * 9.1 + 0.02 * \( \frac{PR(B)}{N} \)
  - PR(B) = 0.18 * 9.1 + 0.02 * (PR(A) + \( \frac{PR(C)}{N} \))
  - PR(C) = 0.18 * 9.1 + 0.02 * 9.1 + 9.1
  - PR(D) = 0.18 * 9.1 + 0.02 * 9.1 + 9.1

PageRank example: result

- Algorithm converges (few iterations sufficient)
- Observations:
  - pages with no in-links: PR = (1-\( \lambda \)) * I/N = 18% / 11 = 1.6%
  - same (or symmetric) inlinks ⇒ same PR
  - one inlink from high PR >> many from low PR
- Hint: think about equilibrium

PageRank using MapReduce

- Mapper: \( \langle i, x \rangle \mapsto \text{node} \oplus \text{out-links} \)
  - for \( j=1..n \) emit \( \langle j, PR(i) \rangle \)
  - emit \( \langle i, \text{out}(i) \rangle \)
- Reducer: \( \langle j, \text{out}(i) \rangle \mapsto 1/\text{out}(i) \)
  - compute: \( PR(i) = \frac{1-\lambda}{N} + \sum_{j=1..n} \frac{PR(j)}{\text{out}(j)} \)
  - for \( j=1..n \) emit \( \langle i, PR(i) \rangle \)
  - emit \( \langle i, \text{in}(i) \rangle \)
- Results go into another reducer (multi-step job)
PageRank: sink nodes

- Consider a simple graph: A → B, assume λ = 0.8
  - PR(A) = (1-λ)/2 + λ = 0.1 ... after update 1, and stays like this
- PR(B) = (1-λ)/2 + λ = 0.18... after update 2, and stays like this
- PR(A) = PR(B) = 0.25 = 72% of the PR is lost
- B is a “sink” (dead end, no outgoing links)
  - with probability (1-λ) jump to another node (PR is shared)
  - with probability λ ... (PR is lost)
- Correction:
  - re-normalize PR to sum to 1 (over-estimates PR for high-in-link nodes)
  - connect sink nodes to every node in the graph (expensive)
  - PR(x) = \frac{1 - λ}{N} + \frac{λ}{N} \sum_{y \in S(x)} PR(y)
  - S = sum of PR over sinks (re-compute before each update)

Hubs and Authorities

- PageRank: simplistic view of a network
- Network topology: different node types:
  - “hub”: page that points to a lot of others
  - e.g. Yahoo / DMZ directory
  - “authority”: page that many others refer to
  - authoritative view on some subject
- HITS algorithm [Kleinberg, 1997]
  - Hyperspace Induced Topic Search
  - automatically determine hubs/authorities
  - PageRank as “half” of HITS

HITS algorithm

- \textit{H}(x), \textit{A}(x) ... hub, authority scores
  - initialize as 1/N, N ... number of pages
  - a good hub links to many good authorities
  - a good authority is referenced by many hubs
  - normalize A,H: \frac{\sum_{y \in N(x)} H(y)}{\sum_{y \in N(x)} A(y)} = 1
  - in practice:
    - used on result set (not all docs like PageRank)
    - developed for IBM Clever project
    - variant used by Teoma (now Ask.com)

Link Spam

- Trackback links (blogs that link to me)
  - based on IP/URL_REFERER
  - artificial feedback loops
- Links from comments on sites with high PR
  - One solution: insert rel=nofollow into links
  - link ignored during PR computation
- Link farms
  - take densely-connected graph
  - hundreds of web domains / IPs can be hosted on one machine

Anchor text

- HTML links contain anchor text
  - description of destination page
  - short, descriptive, like a query
  - re-formulated in different ways
  - human “query expansion”
  - Used in addition to page content
  - together with URL tokens
  - also “surrounding text”
- separate “weights” for every component
  - Significantly more effective than PageRank

Summary

- Massive amounts of data
  - challenging for efficiency, but improves effectiveness
- PageRank
  - probability that random surfer is currently on page x
- Hubs and Authorities (HITS)
  - asymmetric, recursive: good hubs → good authorities
- Anchor Text
  - short, concise description of content on the target page
- Link Spam
  - trackback links, link farms