SIENA*college*
Computer Science

Computer Science 180
Web Design
Siena College
Fall 2011

Topic Notes: Web Search

The World Wide Web is a vast collection of information on pretty much every topic imaginable. The information is of varying degrees of accuracy and usefulness. If we are looking for information about a particular subject, how can we find the "best" pages that contain relevant and useful information?

The Early Days

In the early history of the web, the number of sites and pages was relatively small. Small enough that in the early days, Berners-Lee maintained a list of servers.

This did not last long – it was not practical for humans to maintain such lists as the number of sites grew. Some "manual" attempts continued: Yahoo!'s early existence was as a *web directory*, where links to web sites were maintained by category.

It soon became clear that automation was needed – the search engine.

Search Engines

Search engines have the ability to find web sites based on one or more *search terms* or *words*. Given the sheer size of the web, every aspect must be automated.

The idea of a search engine predated the web. In 1990, the *Archie* search tool was created to search for files (by name) on public FTP servers. Archie would periodically list the files on known servers to create a searchable index. It did not retrieve the actual files.

This idea was applied to web pages by a variety of projects and companies.

Let's consider how modern search engines work.

1. Web *crawlers* (often called *robots*) visit web pages to collect the words found on that page, then follow the links on the page (to find other pages).

2. This information is used to create and update a huge *index* of words and the web sites where those words are found.

3. The words specified for a search are located in the index, and a list of web pages that contain the words are gathered.

4. These pages are *ranked* to determine the order of their presentation in the search results.

A web crawler or robot interacts with web servers in much the same way that browers do. They send requests for a particular file and receive the page contents. Whereas browsers then display the page, a crawler will add or update the page contents in its index, and will add any hyperlinks in the document to its list of pages to index if it they are not already there.

We can see the action of web crawlers by looking at a site's web server logs.

Web crawlers must balance the frequency with which they visit sites with the cost of performing the requests. Frequent crawls means a more up-to-date index, but at the expense of more resources used, both by the web crawler and on the server.

While there is nothing stopping a web crawler from requesting any file that is accessible to regular browsers, a mechanism has been developed whereby a web site can inform web crawlers which of its files it would like to have indexed by crawlers. The *robot exclusion standard* involves the placement of a file `robots.txt` with instructions about which files should not be indexed.

Given a search engine's index and a search query, there will be many pages that are "matches" for the search query. Google, for example, tells about how many pages in its index match each query.

The *ranking* of pages containing the search results is crucial. A search engine needs to present those pages that are most likely to contain what the searcher is seeking. A searcher needs to know that the pages presented first are the most relevant, authoritative, and accurate.

Search engines can have very simple ranking schemes, such as ranking pages based on the number of times the search terms appear on the page. The importance of the terms might depend on where in the page they are found. A search term found in a page title or heading tag or even earlier in the document might be given more weight than one found in a regular paragraph of text late in the document.

Simple schemes, however, can be easily influenced by web developers, who want their pages to appear high on the list of links returned by search engines. They could place popular search terms repeatedly on their pages, even hidden from view (in a hidden element, or in white text on a white background, or using any number of other tricks).

Search engines work to defeat such tricks while some web developers continually try to figure out how best to increase their pages' rankings.

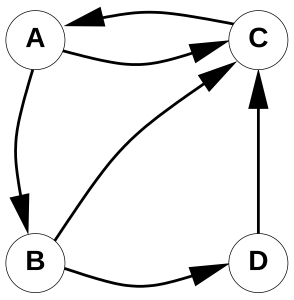
Many additional rules and heuristics are used in an attempt to rank, automatically, the "best" pages to match a given search query. Pages that are somehow "official" or otherwise authoritative should be given a higher rank.

Google uses a relatively straightforward idea called *PageRank*: the more other sites that link to a given site, the more likely it is to be authoritative. So sites are ranked in the search results based (in large part) upon this measure. Moreover, links to a page from highly-ranked (i.e., important) pages contribute to a higher rank for that page.

PageRank was developed by Page and Brin at Stanford, and later was licensed to Google when they started the company.

Think about the "major sites" on the Internet. Many, many people link to `espn.com` sites, but it is much less likely that `espn.com` links to them. If your page is worthy of some links from pages on `espn.com`, PageRank will consider it also to be a relatively important page.

Consider a small example with 4 pages:



Initially, all ranks are equal, so there is an equal chance of visiting any of our pages. We'll label each with a rank of 1. Now, suppose there is a 75% chance that someone viewing a page will follow a link on that page, each link with equal probability. That leaves a 25% chance that the person will stop "browsing" and jump randomly to any page.

Given equal chances of each page as the first stop, what are the likelihoods of each page being our next visited page? Well, it's the 25% chance that we will arrive here randomly, plus the chances from among that other 75% that someone will click a link from elsewhere to arrive at this page. This leads to the following:

$$PR(A) = 0.25 + 0.75(PR(C))$$
$$PR(B) = 0.25 + 0.75(PR(A)/2)$$
$$PR(C) = 0.25 + 0.75(PR(A)/2 + PR(B)/2 + PR(D))$$
$$PR(D) = 0.25 + 0.75(PR(B)/2)$$

We could solve this system of 4 equations in 4 unknowns, or iterate to a solution:

```
Iteration  0: PR(A)= 1.000, PR(B)= 1.000, PR(C)= 1.000, PR(D)= 1.000
Iteration  1: PR(A)= 1.000, PR(B)= 0.625, PR(C)= 1.750, PR(D)= 0.625
Iteration  2: PR(A)= 1.562, PR(B)= 0.625, PR(C)= 1.328, PR(D)= 0.484
Iteration  3: PR(A)= 1.246, PR(B)= 0.836, PR(C)= 1.434, PR(D)= 0.484
Iteration  4: PR(A)= 1.325, PR(B)= 0.717, PR(C)= 1.394, PR(D)= 0.563
Iteration  5: PR(A)= 1.296, PR(B)= 0.747, PR(C)= 1.439, PR(D)= 0.519
Iteration  6: PR(A)= 1.329, PR(B)= 0.736, PR(C)= 1.405, PR(D)= 0.530
Iteration  7: PR(A)= 1.304, PR(B)= 0.748, PR(C)= 1.422, PR(D)= 0.526
Iteration  8: PR(A)= 1.316, PR(B)= 0.739, PR(C)= 1.414, PR(D)= 0.531
Iteration  9: PR(A)= 1.311, PR(B)= 0.744, PR(C)= 1.419, PR(D)= 0.527
Iteration 10: PR(A)= 1.314, PR(B)= 0.741, PR(C)= 1.416, PR(D)= 0.529
Iteration 11: PR(A)= 1.312, PR(B)= 0.743, PR(C)= 1.417, PR(D)= 0.528
Iteration 12: PR(A)= 1.313, PR(B)= 0.742, PR(C)= 1.416, PR(D)= 0.529
Iteration 13: PR(A)= 1.312, PR(B)= 0.742, PR(C)= 1.417, PR(D)= 0.528
Iteration 14: PR(A)= 1.313, PR(B)= 0.742, PR(C)= 1.417, PR(D)= 0.528
Iteration 15: PR(A)= 1.313, PR(B)= 0.742, PR(C)= 1.417, PR(D)= 0.528
```

Iteration 1 is the ranks (chances of arriving at the page) after 1 click. Then Iteration 2 is after 2, and so on. This eventually converges and we have the relative ranks of our pages.

PageRank is one factor in Google's rankings. Other factors come in to play as well, including that Google ranks sites higher that pay to be ranked higher.

Image and other Media Search

Simple word matching is a very effective mechanism for finding text in a web page. But what if we are trying to search for other kinds of media that match a search query?

Computers are very bad at determining the contents of an image. Research in computer vision is progressing, but many tasks that are very simple for a human are still very difficult for a computer. Bottom line, at least for now, is that giving a computer an image and asking it to come up with some keywords that are relevant to that image will lead to a very limited set of results.

If a search engine wants to support an image search (and it probably does), it will need to have some other way besides a straight computational image analysis to obtain a list of keywords.

Image searches are still unsatisfying in some cases, but do return reasonable results in other cases.

Some keywords for images can likely be found near the image in a web page. Those may be captions or even keywords entered manually by a human (think of the image keywords on sites like Flickr). But which words are going to be good keywords for the image? What if there is no obvious context?

This is an example of a more general problem of creating reliable *metadata* – data that describes some data (like an image file, an audio clip, or a video clip).

We certainly can't expect to have humans list keywords for all images on the web. Or can we? Luis Von Ahn of CMU is an expert in "human computation" – getting people to perform useful work for free by distributing the work among a very large number of people. We will watch part of his talks on this soon, but for now, we'll look at what he calls the "ESP Game".

On the web: The ESP Game at
`http://www.gwap.com/gwap/gamesPreview/espgame/`

On the web: "The ESP Game" at Wikipedia at
`http://en.wikipedia.org/wiki/ESP_game`

With this game, people need to agree upon words they think of when an image is presented. The fact that there needs to be some agreement helps avoid inaccurate metadata creation either through misspellings, incorrect image interpretation, or malicious attempts to introduce bad metadata.

See von Ahn discuss this in a Google Tech Talk:

On the web: Luis von Ahn Google Tech Talk on Human Computation at
`http://video.google.com/videoplay?docid=-8246463980976635143&q=google+tech+`
(start around 7:30 for the ESP Game section)

Note that Google has recently introduced an image-based image search. Instead of searching by keywords:

On the web: Google Search by Image at
`http://www.google.com/insidesearch/searchbyimage.html`