Web Search. Web Spidering

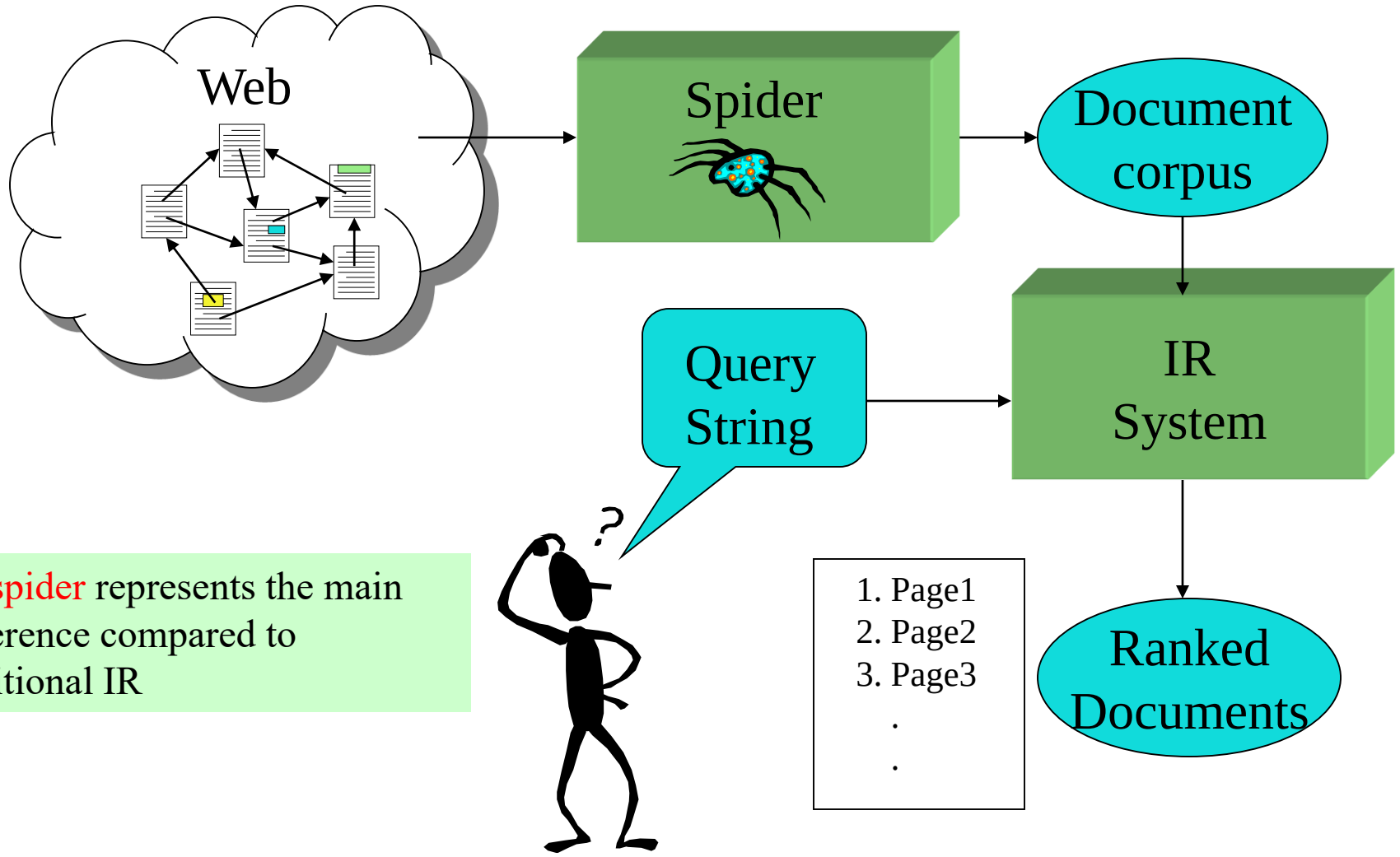
Introduction

This material was prepared by Diana Inkpen, University of Ottawa, 2005, updated 2021. Some of these slides were originally prepared by Raymond Mooney, University of Texas Austin.

Outline

- Information Retrieval applied on the Web
- The Web – the largest collection of documents available today
  - Still, a collection
  - Should be able to apply “traditional” IR techniques, with few changes
- Web Search
- Spidering

Web Search Using IR



the spider represents the main difference compared to traditional IR

The World Wide Web

- Developed by Tim Berners-Lee in 1990 at CERN to organize research documents available on the Internet.
- Combined idea of documents available by FTP with the idea of *hypertext* to link documents.
- Developed initial HTTP network protocol, URLs, HTML, and first “web server.”

Web Pre-History

- Ted Nelson developed idea of hypertext in 1965.
- Doug Engelbart invented the mouse and built the first implementation of hypertext in the late 1960's at SRI.
- ARPANET was developed in the early 1970's.
  - For what main reason?
- The basic technology was in place in the 1970's; but it took the PC revolution and widespread networking to inspire the web and make it practical.

Web Browser History

- Early browsers were developed in 1992 (Erwise, ViolaWWW).
- In 1993, Marc Andreessen and Eric Bina at UIUC NCSA developed the Mosaic browser and distributed it widely.
- Andreessen joined with James Clark (Stanford Prof. and Silicon Graphics founder) to form Mosaic Communications Inc. in 1994 (which became Netscape to avoid conflict with UIUC).
- Microsoft licensed the original Mosaic from UIUC and used it to build Internet Explorer in 1995.

Search Engine Early History

- By late 1980's many files were available by anonymous FTP.
- In 1990, Alan Emtage of McGill Univ. developed Archie (short for "archives")
  - Assembled lists of files available on many FTP servers.
  - Allowed regex search of these file names.
- In 1993, Veronica and Jughead were developed to search names of text files available through Gopher servers.

Web Search History

- In 1993, early web robots (spiders) were built to collect URL's:
  - Wanderer
  - ALIWEB (Archie-Like Index of the WEB)
  - WWW Worm (indexed URL's and titles for regex search)
- In 1994, Stanford grad students David Filo and Jerry Yang started manually collecting popular web sites into a topical hierarchy called Yahoo.

Web Search History (cont)

- In early 1994, Brian Pinkerton developed WebCrawler as a class project at U Wash. (eventually became part of Excite and AOL).
- A few months later, Fuzzy Maudlin, a grad student at CMU developed Lycos. First to use a standard IR system as developed for the DARPA Tipster project. First to index a large set of pages.
- In late 1995, DEC developed Altavista. Used many Alpha machines to quickly process large numbers of queries. Supported boolean operators, phrases.

Web Search Recent History

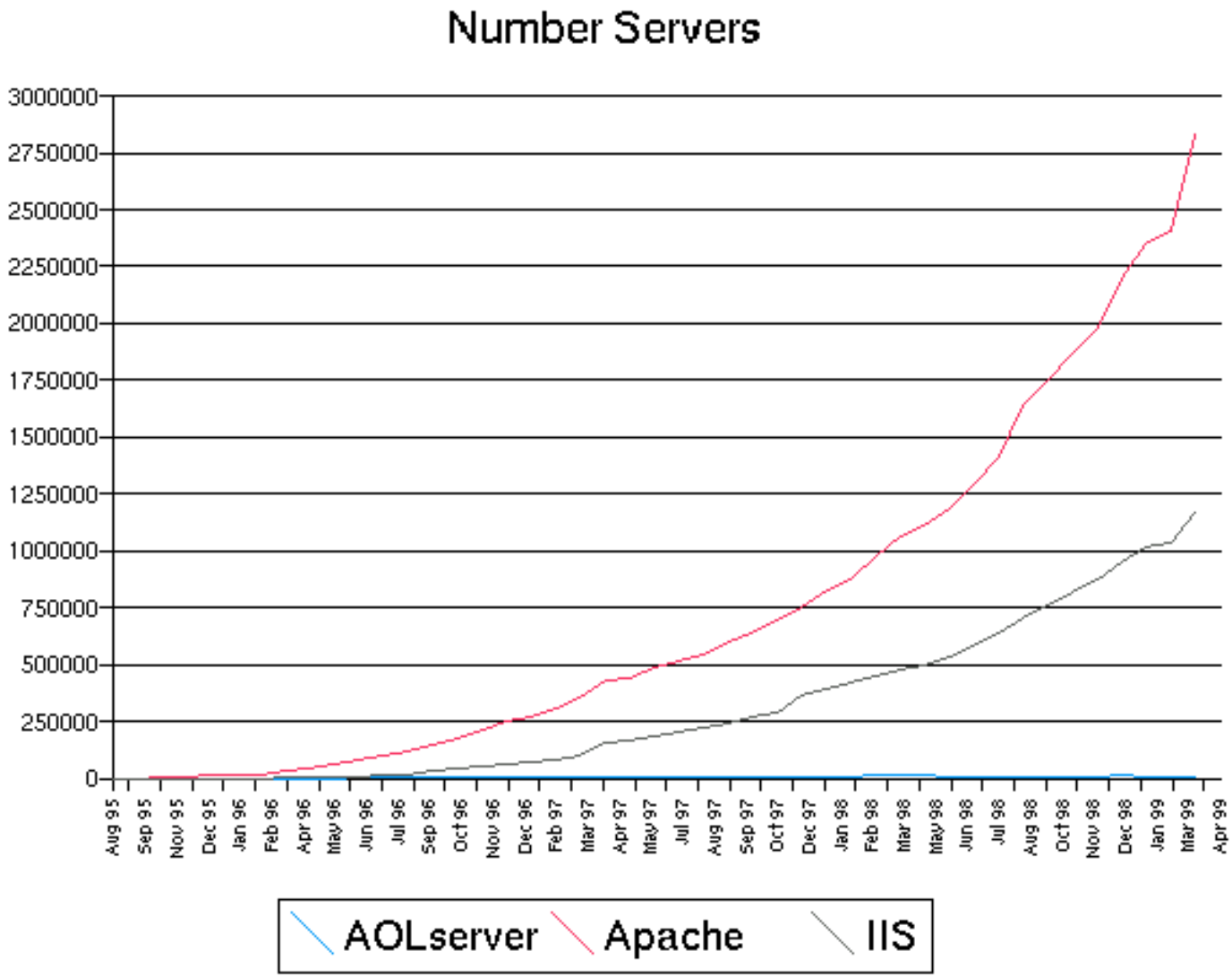
- In 1998, Larry Page and Sergey Brin, Ph.D. students at Stanford, started Google. Main advance is use of *link analysis* to rank results partially based on authority.

- Conclusions:
  - Most popular search engines were developed by graduate students
  - Don't wait too long to develop your own search engine! ☺

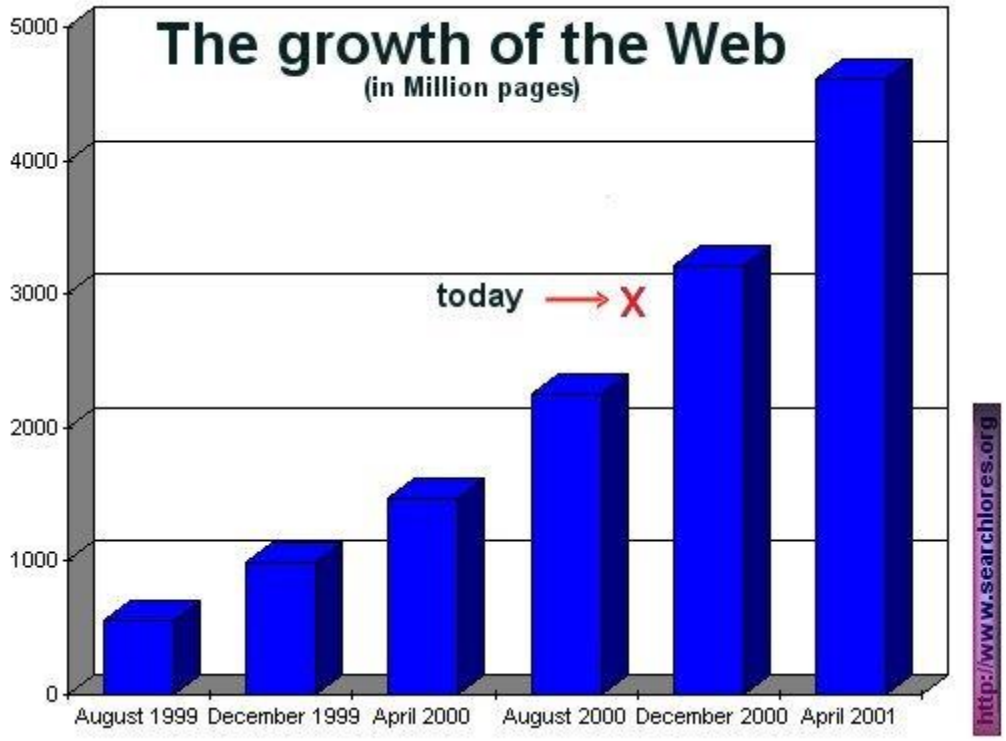
Web Challenges for IR

- Distributed Data: Documents spread over millions of different web servers.
- Volatile Data: Many documents change or disappear rapidly (e.g. dead links).
- Large Volume: Billions of separate documents.
- Unstructured and Redundant Data: No uniform structure, HTML errors, up to 30% (near) duplicate documents.
- Quality of Data: No editorial control, false information, poor quality writing, typos, etc.
- Heterogeneous Data: Multiple media types (images, video, VRML), languages, character sets, etc.

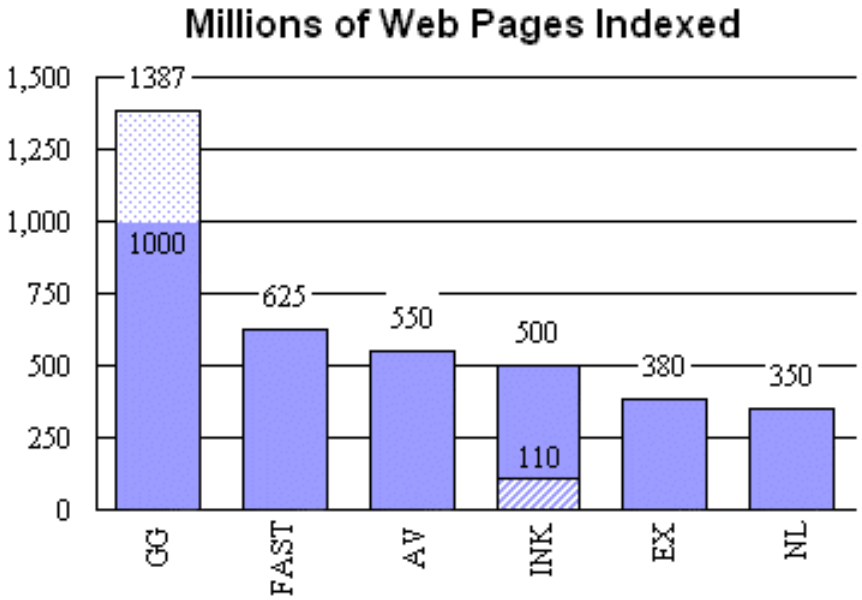
Number of Web Servers



Number of Web Pages



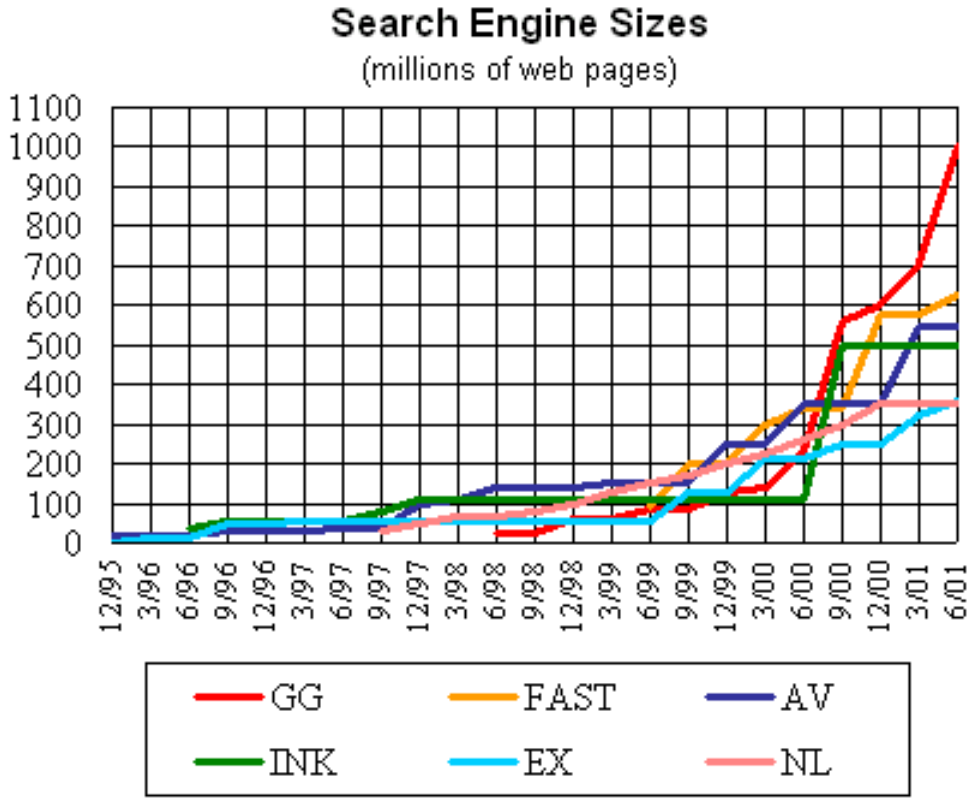
Number of Web Pages Indexed



SearchEngineWatch, Aug. 15, 2001

Assuming about 20KB per page,
1 billion pages is about 20 terabytes of data.

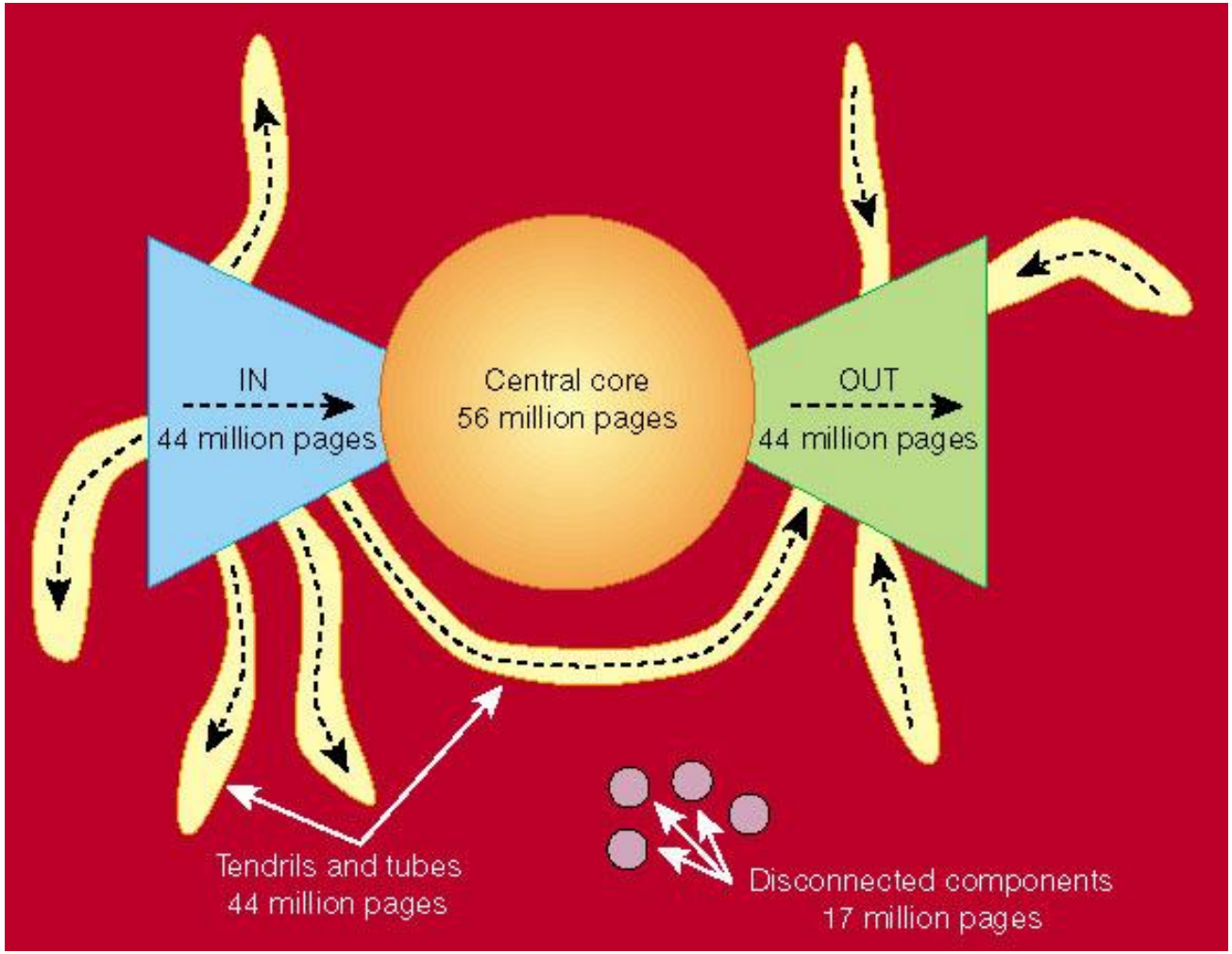
Growth of Web Pages Indexed



SearchEngineWatch, Aug. 15, 2001

Google lists current number of pages searched.

Graph Structure in the Web



http://www9.org/w9cdrom/160/160.html

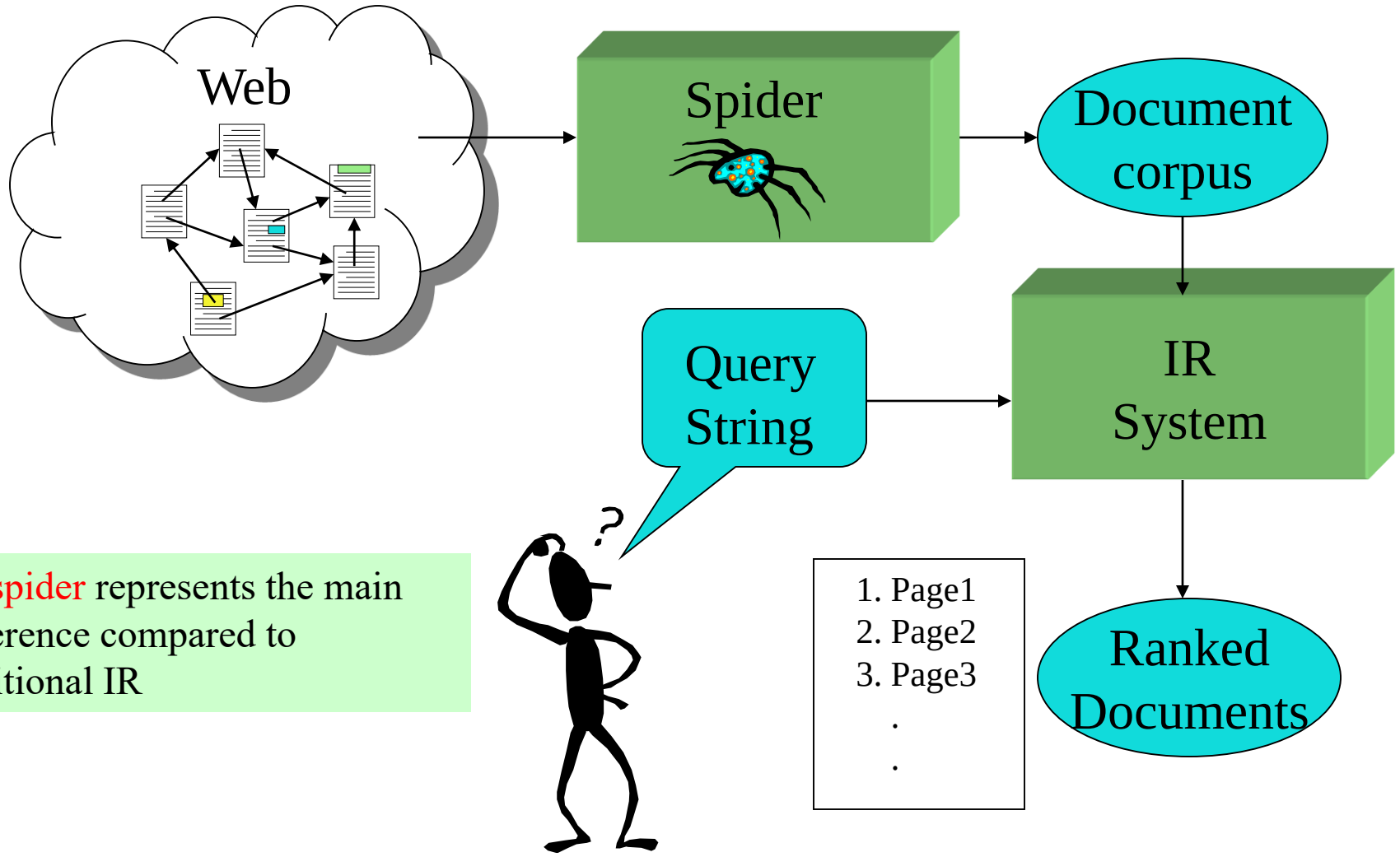
Zipf's Law on the Web

- Length of web pages has a Zipfian distribution.
- Number of hits to a web page has a Zipfian distribution.

Manual Hierarchical Web Taxonomies

- Yahoo approach of using human editors to assemble a large hierarchically structured directory of web pages.
  - http://www.yahoo.com/
- Open Directory Project is a similar approach based on the distributed labor of volunteer editors ("net-citizens provide the collective brain"). Used by most other search engines. Started by Netscape.
  - http://www.dmoz.org/

Web Search Using IR



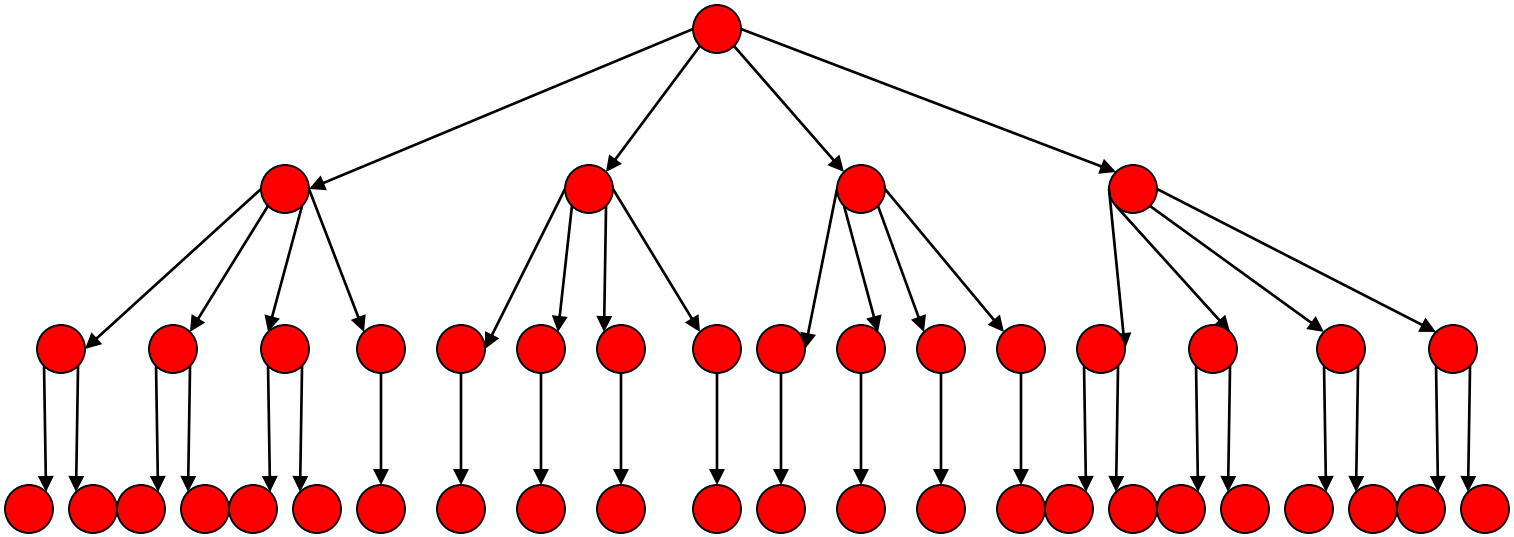
the spider represents the main difference compared to traditional IR

Spiders (Robots/Bots/Crawlers)

- Start with a comprehensive set of root URL's from which to start the search.
- Follow all links on these pages recursively to find additional pages.
- Index/Process all **novel** found pages in an inverted index as they are encountered.
- May allow users to directly submit pages to be indexed (and crawled from).

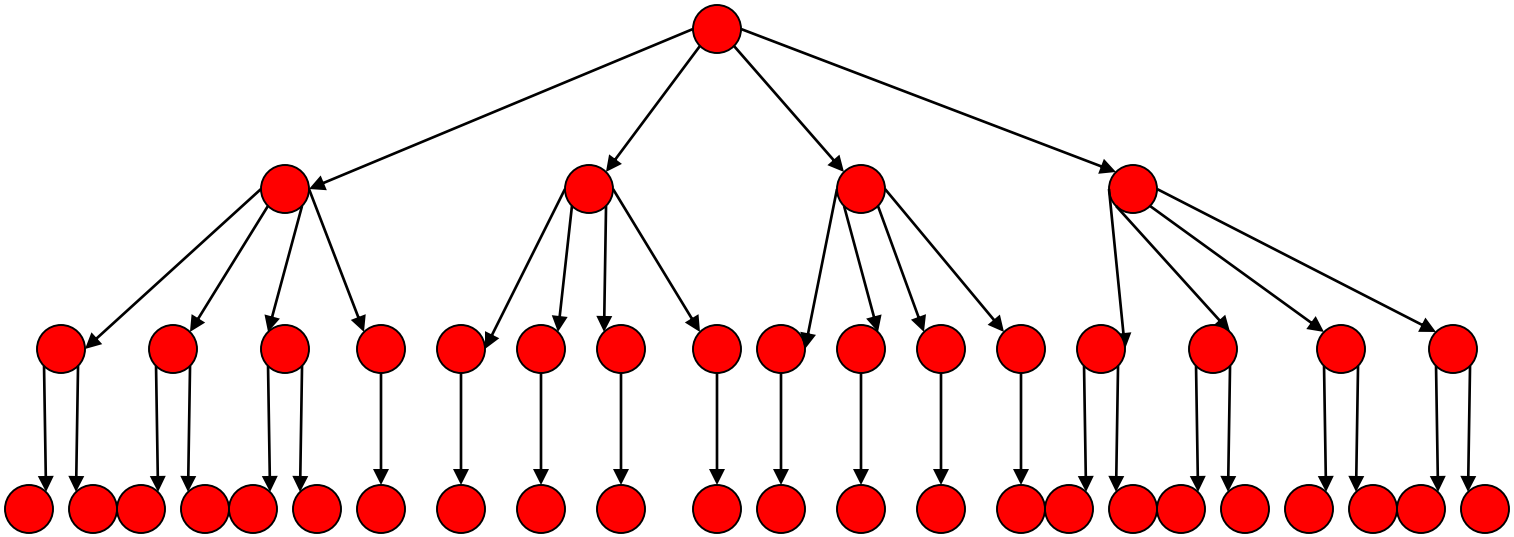
Search Strategies

Breadth-first Search



Search Strategies (cont)

Depth-first Search



Search Strategy Trade-Off's

- Breadth-first explores uniformly outward from the root page but requires memory of all nodes on the previous level (exponential in depth). Standard spidering method.
- Depth-first requires memory of only depth times branching-factor (linear in depth) but gets "lost" pursuing a single thread.
- Both strategies can be easily implemented using a queue of links (URL's).

Avoiding Page Duplication

- Must detect when revisiting a page that has already been spidered (web is a graph not a tree).
- Must efficiently index visited pages to allow rapid recognition test.
  - Tree indexing (e.g. trie), Hashtable
- Index page using URL as a key.
  - Must canonicalize URL's (e.g. delete ending "/")
  - Not detect duplicated or mirrored pages.
- Index page using textual content as a key.
  - Requires first downloading page.

Spidering Algorithm

Initialize queue (Q) with initial set of known URL's.
Until Q empty or page or time limit exhausted:
    Pop URL, L, from front of Q.
    If L is not to an HTML page (.gif, .jpeg, .ps, .pdf, .ppt…)
        continue loop.
    If already visited L, continue loop.
    Download page, P, for L.
    If cannot download P (e.g. 404 error, robot excluded)
        continue loop.
    Index P (e.g. add to inverted index or store cached copy).
    Parse P to obtain list of new links N.
    Append N to the end of Q.

Queueing Strategy

- How new links are added to the queue determines search strategy.
- FIFO (append to end of Q) gives breadth-first search.
- LIFO (add to front of Q) gives depth-first search.
- Heuristically ordering the Q gives a "focused crawler" that directs its search towards "interesting" pages.

Restricting Spidering

- You can restrict spider to a particular site.
  - Remove links to other sites from Q.
- You can restrict spider to a particular directory.
  - Remove links not in the specified directory.
- Obey page-owner restrictions (robot exclusion).

Link Extraction

- Must find all links in a page and extract URLs.
  -
  - <frame src="site-index.html">
- Must complete relative URL's using current page URL:
  - to

    http://www.site.uottawa.ca/~diana/csi4107/A1.htm
  - to

    http://www.site.uottawa.ca/~diana/csi4107/paper-presentations.html

URL Syntax

- A URL has the following syntax:
  - <scheme>://<authority><path>?<query>#<fragment>
- A *query* passes variable values from an HTML form and has the syntax:
  - <variable>=<value>&<variable>=<value>…
- A *fragment* is also called a *reference* or a *ref* and is a pointer within the document to a point specified by an anchor tag of the form:
  - <A NAME="<fragment>">

Link Canonicalization

- Equivalent variations of ending directory normalized by removing ending slash.
  - http://www.site.uottawa.ca/~diana/
  - http://www.site.uottawa.ca/~diana
- Internal page fragments (ref's) removed:
  - http://www.site.uottawa.ca/~diana/csi1102/index.html#Eval
  - http://www.site.uottawa.ca/~diana/csi1102/index.html

Anchor Text Indexing

- Extract anchor text (between <a> and) of each link followed.
- Anchor text is usually descriptive of the document to which it points.
- Add anchor text to the content of the destination page to provide additional relevant keyword indices.
  - Evil Empire
  - IBM

Anchor Text Indexing (cont'd)

- Helps when descriptive text in destination page is embedded in image logos rather than in accessible text.
- Many times anchor text is not useful:
  - "click here"
- Increases content more for popular pages with many in-coming links, increasing recall of these pages.
- May even give higher weights to tokens from anchor text.

Robot Exclusion

- Web sites and pages can specify that robots should not crawl/index certain areas.
- Two components:
  - Robots Exclusion Protocol: Site wide specification of excluded directories.
  - Robots META Tag: Individual document tag to exclude indexing or following links.

Robots Exclusion Protocol

- Site administrator puts a "robots.txt" file at the root of the host's web directory.
  - http://www.ebay.com/robots.txt
  - http://www.cnn.com/robots.txt
- File is a list of excluded directories for a given robot (user-agent).
  - Exclude all robots from the entire site:

```
User-agent: *
Disallow: /
```

Robot Exclusion Protocol Examples

- Exclude specific directories:

```
User-agent: *
Disallow: /tmp/
Disallow: /cgi-bin/
Disallow: /users/paranoid/
```

- Exclude a specific robot:

```
User-agent: GoogleBot
Disallow: /
```

- Allow a specific robot:

```
User-agent: GoogleBot
Disallow:
```

Robot Exclusion Protocol Details

- Only use blank lines to separate different User-agent disallowed directories.
- One directory per “Disallow” line.
- No regex patterns in directories.

Robots META Tag

- Include META tag in HEAD section of a specific HTML document.
  - <meta name="robots" content="none">
- Content value is a pair of values for two aspects:
  - index | noindex: Allow/disallow indexing of this page.
  - follow | nofollow: Allow/disallow following links on this page.

Robots META Tag (cont)

- Special values:
  - all = index,follow
  - none = noindex,nofollow
- Examples:

<meta name="robots" content="noindex,follow">

<meta name="robots" content="index,nofollow">

<meta name="robots" content="none">

Robot Exclusion Issues

- META tag is newer and less well-adopted than “robots.txt”.
- Standards are conventions to be followed by “good robots.”
- Companies have been prosecuted for “disobeying” these conventions and “trespassing” on private cyberspace.

Multi-Threaded Spidering

- Bottleneck is network delay in downloading individual pages.
- Best to have multiple threads running in parallel each requesting a page from a different host.
- Distribute URL's to threads to guarantee equitable distribution of requests across different hosts to maximize through-put and avoid overloading any single server.
- Early Google spider had multiple co-ordinated crawlers with about 300 threads each, together able to download over 100 pages per second.

Directed/Focused Spidering

- Sort queue to explore more “interesting” pages first.
- Two styles of focus:
  - Topic-Directed
  - Link-Directed

Topic-Directed Spidering

- Assume desired topic description or sample pages of interest are given.
- Sort queue of links by the similarity (e.g. cosine metric) of their source pages and/or anchor text to this topic description.
  - Related to Topic Tracking and Detection

Link-Directed Spidering

- Monitor links and keep track of in-degree and out-degree of each page encountered.
- Sort queue to prefer popular pages with many in-coming links (*authorities*).
- Sort queue to prefer summary pages with many out-going links (*hubs*).
  - Google's PageRank algorithm

Keeping Spidered Pages Up to Date

- Web is very dynamic: many new pages, updated pages, deleted pages, etc.
- Periodically check spidered pages for updates and deletions:
  - Just look at header info (e.g. META tags on last update) to determine if page has changed, only reload entire page if needed.
- Track how often each page is updated and preferentially return to pages which are historically more dynamic.
- Preferentially update pages that are accessed more often to optimize freshness of popular pages.