Information Retrieval

Web cralwing and search

Hamid Beigy

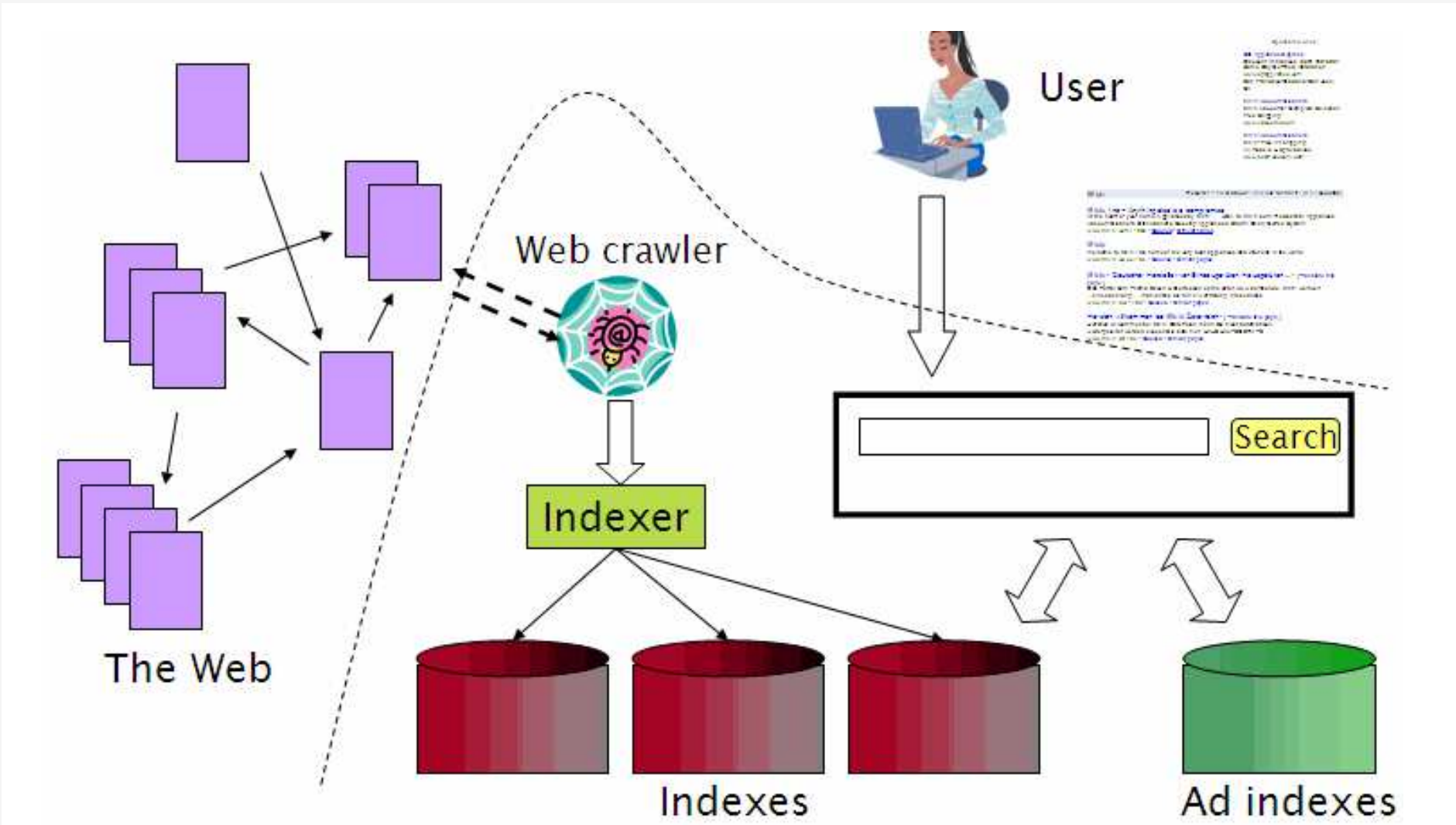
Sharif university of technology

December 23, 2018

Table of contents

Introduction

Web Search



Web Search

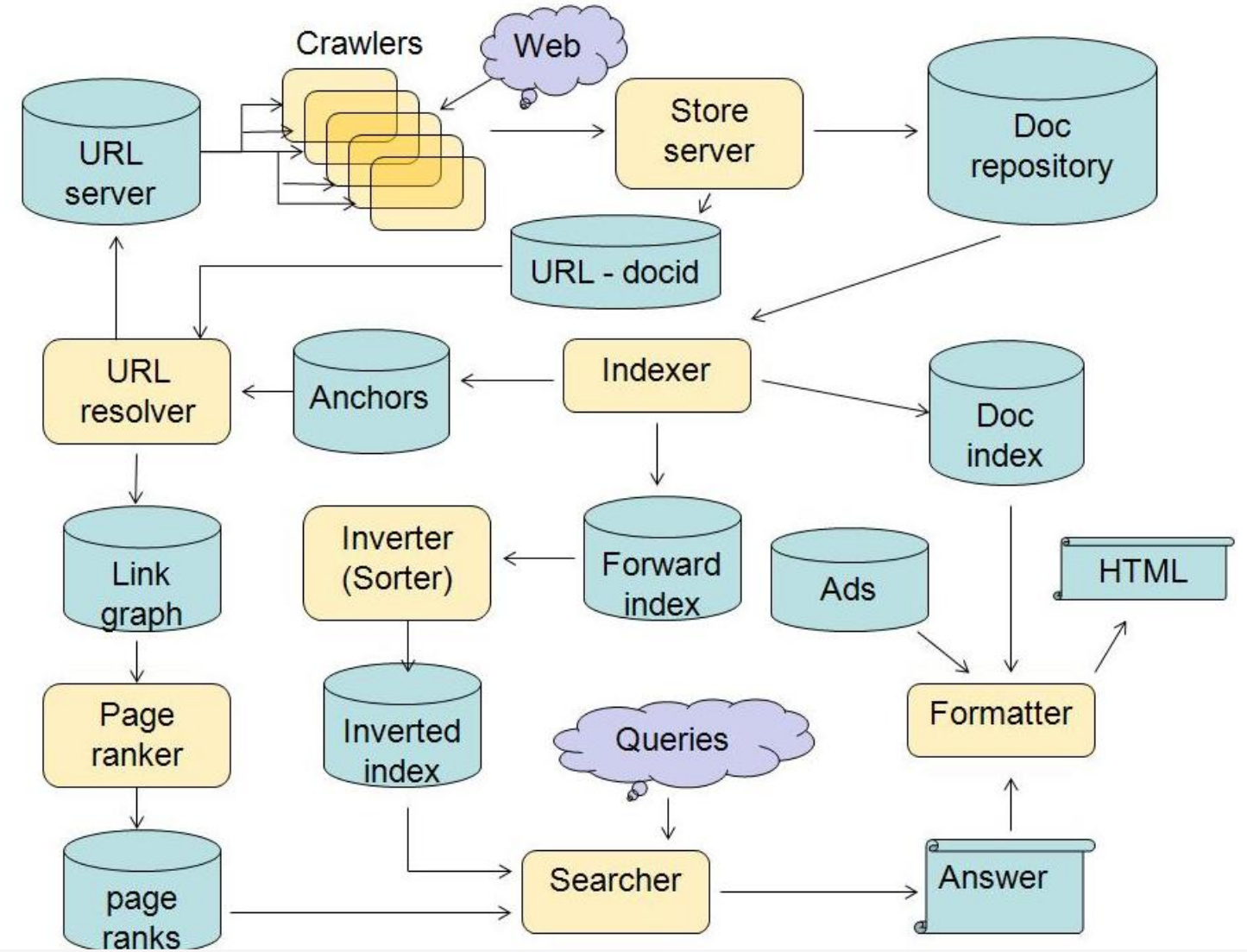
The World Wide Web is huge.

1. 100,000 indexed pages in1994.
2. 10,000,000,000s indexed pages in 2013.
3. Most queries will return millions of pages with high similarity.
4. Content(text)alone cannot discriminate.
5. Use the structure of the Web(a graph).
6. Gives indications of usefulness of each page.

Without web search engines

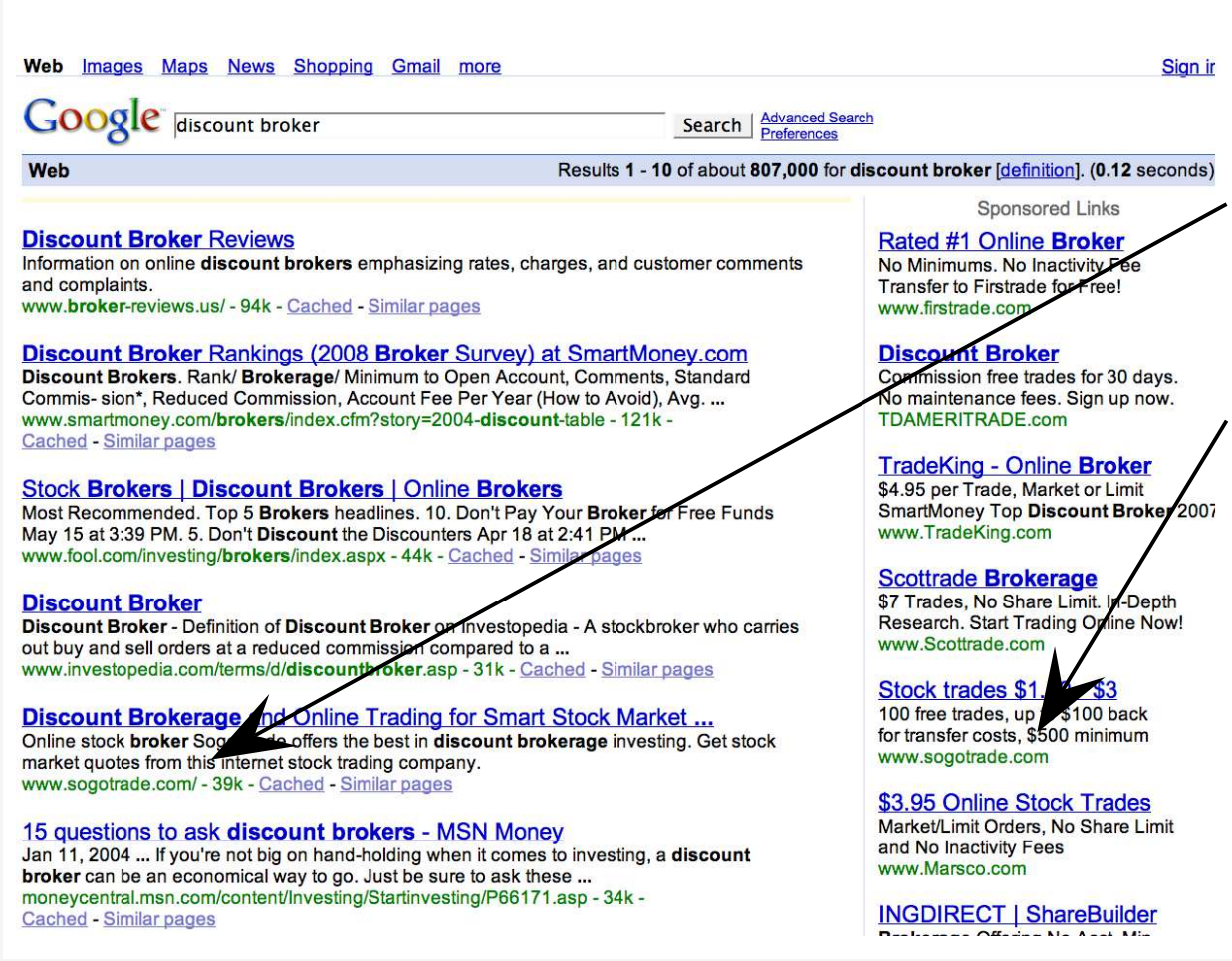
1. Without search, content is hard to find.
2. Without search, there is no incentive to create content.
   - Why publish something if nobody will read it?
   - Why publish something if I dont get ad revenue from it?
3. Somebody needs to pay for the web.
   - Servers, web infrastructure, content creation
   - A large part today is paid by search ads.
   - Search pays for the web.
4. On the web, search is not just a nice feature, search is a key enabler of the web.

Google (1998)


Crawlers
Web
URL server
Store server
Doc repository
URL - docid
URL resolver
Anchors
Indexer
Doc index
Link graph
Inverter (Sorter)
Forward index
Ads
HTML
Page ranker
Inverted index
Queries
Formatter
page ranks
Searcher
Answer

Results of a query

Web pages (left) and ads (right)



SogoTrade appears in search results.

SogoTrade appears in ads.

Do search engines rank advertisers higher than non-advertisers?

All major search engines claim no.

Duplicate detection

Duplicate detection

1. The web is full of duplicated content (30%–40%).
2. More so than many other collections
3. Exact duplicates (easy to eliminate by using hash/fingerprint)
4. Near-duplicates (difficult to eliminate)
5. For the user, it's annoying to get a search result with near-identical documents.
6. We need to eliminate near-duplicates.

Detecting near-duplicates

1. Compute similarity with an edit-distance measure
2. We want syntactic (as opposed to semantic) similarity.
   True semantic similarity (similarity in content) is too difficult to compute.
3. We do not consider documents near-duplicates if they have the same content, but express it with different words.
4. Use similarity threshold θ to make the call is/isn't a near-duplicate.
   For example, two documents are near-duplicates if similarity $> \theta = 80\%$.

Represent each document as set of **shingles**

- A shingle is simply a word n-gram.
- Shingles are used as features to measure syntactic similarity of documents.
- For example, for $n = 3$, a rose is a rose is a rose would be represented as this set of shingles:
  { a-rose-is, rose-is-a, is-a-rose }
- We can map shingles to $1..2^m$ (e.g., $m = 64$) by fingerprinting.
- From now on: s_k refers to the shingle's fingerprint in $1..2^m$.
- We define the similarity of two documents as the Jaccard coefficient of their shingle sets.

Spam

The goal of spamming on the web

1. You have a page that will generate lots of revenue for you if people visit it.
2. Therefore, you would like to direct visitors to this page.
3. One way of doing this: get your page ranked highly in search results.

Spam technique: Keyword stuffing / Hidden text

1. Misleading meta-tags, excessive repetition
2. Hidden text with colors, style sheet tricks etc.
3. Used to be very effective, most search engines now catch these

The war against spam

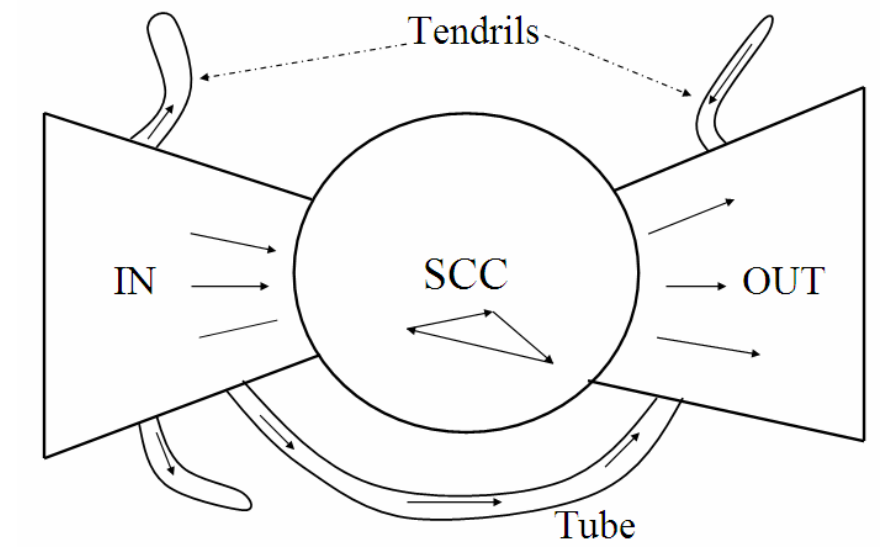
1. Quality indicators
   - Links, statistically analyzed (PageRank etc)
   - Usage (users visiting a page)
   - No adult content (e.g., no pictures with flesh-tone)
   - Distribution and structure of text
2. Combine all of these indicators and use machine learning
3. Editorial intervention
   - Blacklists
   - Top queries audited
   - Complaints addressed
   - Suspect patterns detected

Web IR

Web IR: Differences from traditional IR

1. Links: The web is a hyperlinked document collection.
2. Queries: Web queries are different, more varied and there are a lot of them. How many?
3. Users: Users are different, more varied and there are a lot of them. How many?
4. Documents: Documents are different, more varied and there are a lot of them. How many?
5. Context: Context is more important on the web than in many other IR applications.
6. Ads and spam

Bowtie structure of the web



1. Strongly connected component (SCC) in the center
2. Lots of pages that get linked to, but don't link (OUT)
3. Lots of pages that link to other pages, but don't get linked to (IN)
4. Tendrils, tubes, islands

How do users evaluate search engines?

1. Classic IR relevance (as measured by F) can also be used for web IR.
2. Equally important: Trust, duplicate elimination, readability, loads fast, no pop-ups

Search in a hyperlinked collection

- Web search in most cases is interleaved with navigation (with following links).
- Different from most other IR collections
- Distributed content creation: no design, no coordination
- Unstructured (text, html), semistructured (html, xml), structured/relational (databases)
- Dynamically generated content

Size of the web

Size of the web

1. What is size? Number of web servers? Number of pages? Terabytes of data available?
2. Some servers are seldom connected (such as your laptop running a web server)
3. The dynamic web is infinite.

Sampling methods

1. Random queries
2. Random searches
3. Random IP addresses
4. Random walks

Web crawler

Basic crawler operation

1. Initialize queue with URLs of known seed pages
2. Repeat
   - Take URL from queue
   - Fetch and parse page
   - Extract URLs from page
   - Add URLs to queue
3. Fundamental assumption: The web is well linked.

What's wrong with the simple crawler

1. Scale: we need to distribute.
2. We can't index everything: we need to subselect. How?
3. Duplicates: need to integrate duplicate detection
4. Spam: need to integrate spam detection
5. Politeness: we need to be "nice" and space out all requests for a site over a longer period (hours, days)
6. Freshness: we need to recrawl periodically.
   - Because of the size of the web, we can do frequent recrawls only for a small subset.
   - Again, subselection problem or prioritization

What a crawler must do?

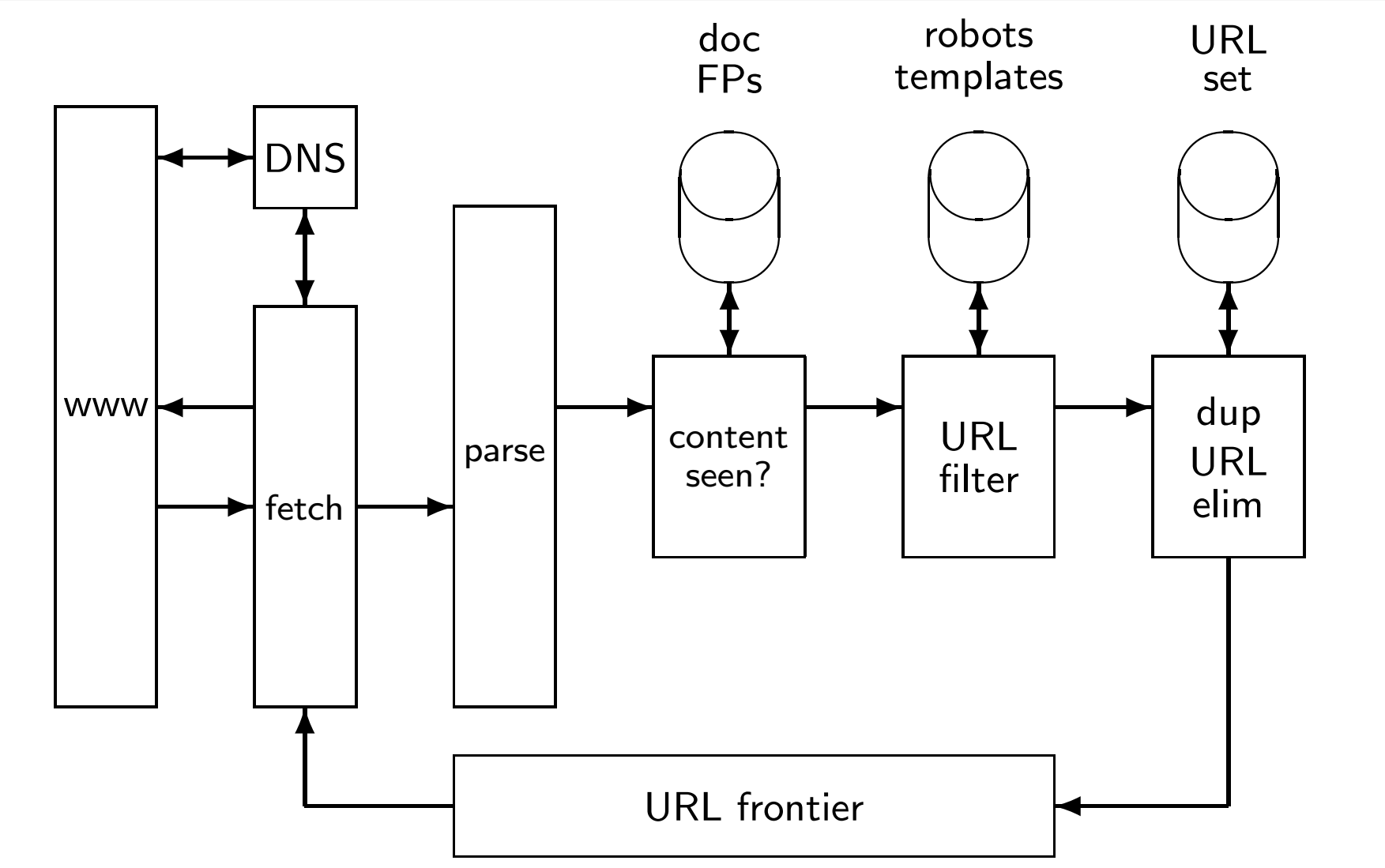
1. Be polite
   - Don't hit a site too often
   - Only crawl pages you are allowed to crawl: robots.txt
2. Be robust
   - Be immune to duplicates, very large pages, very large websites, dynamic pages etc

Robots.txt

1. Protocol for giving crawlers ("robots") limited access to a website, originally from 1994
2. Examples:
   - User-agent: *
     Disallow: /yoursite/temp/
   - User-agent: searchengine
     Disallow: /
3. Important: cache the robots.txt file of each site we are crawling

A real crawler

Basic crawl architecture



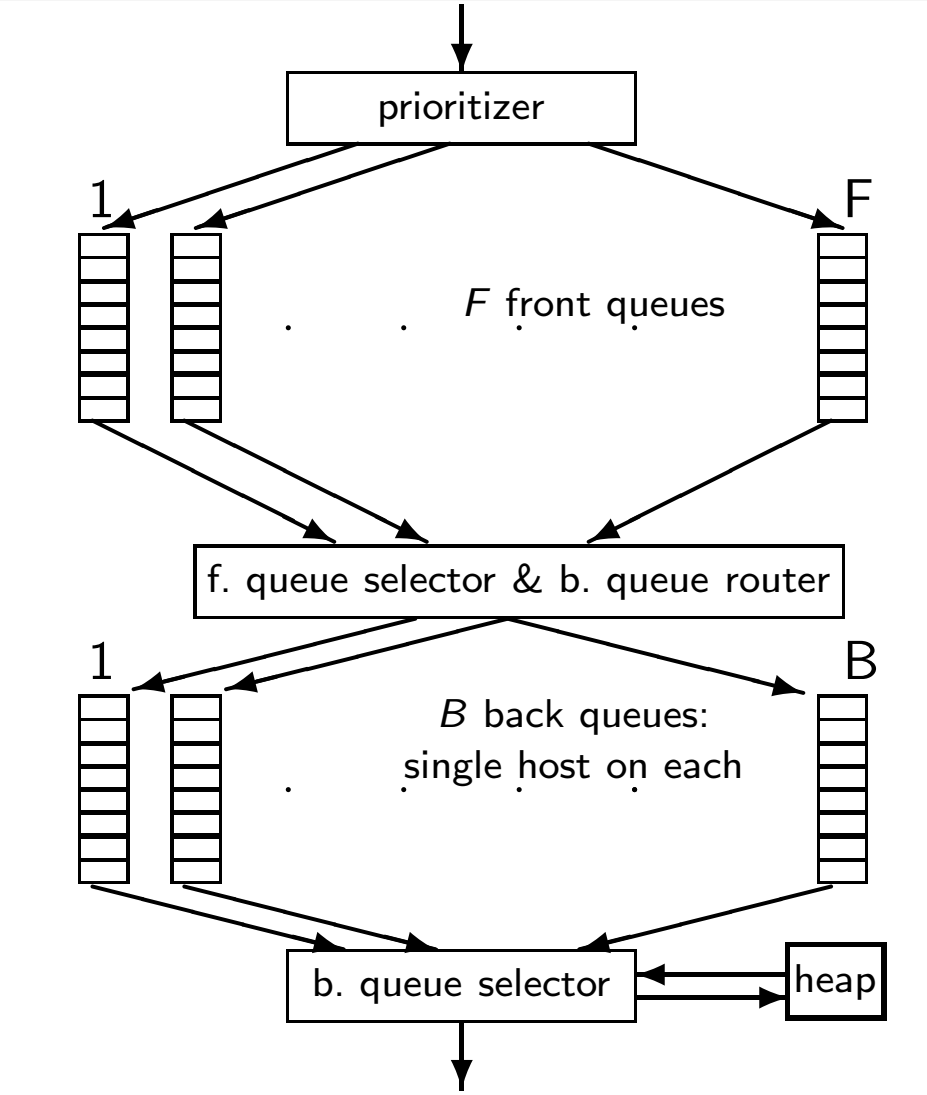
URL frontier

- The URL frontier is the data structure that holds and manages URLs we've seen, but that have not been crawled yet.
- Can include multiple pages from the same host
- Must avoid trying to fetch them all at the same time
- Must keep all crawling threads busy

URL frontier: Two main considerations

- Politeness: Don't hit a web server too frequently
  - E.g., insert a time gap between successive requests to the same server
- Freshness: Crawl some pages (e.g., news sites) more often than others
- Not an easy problem: simple priority queue fails.

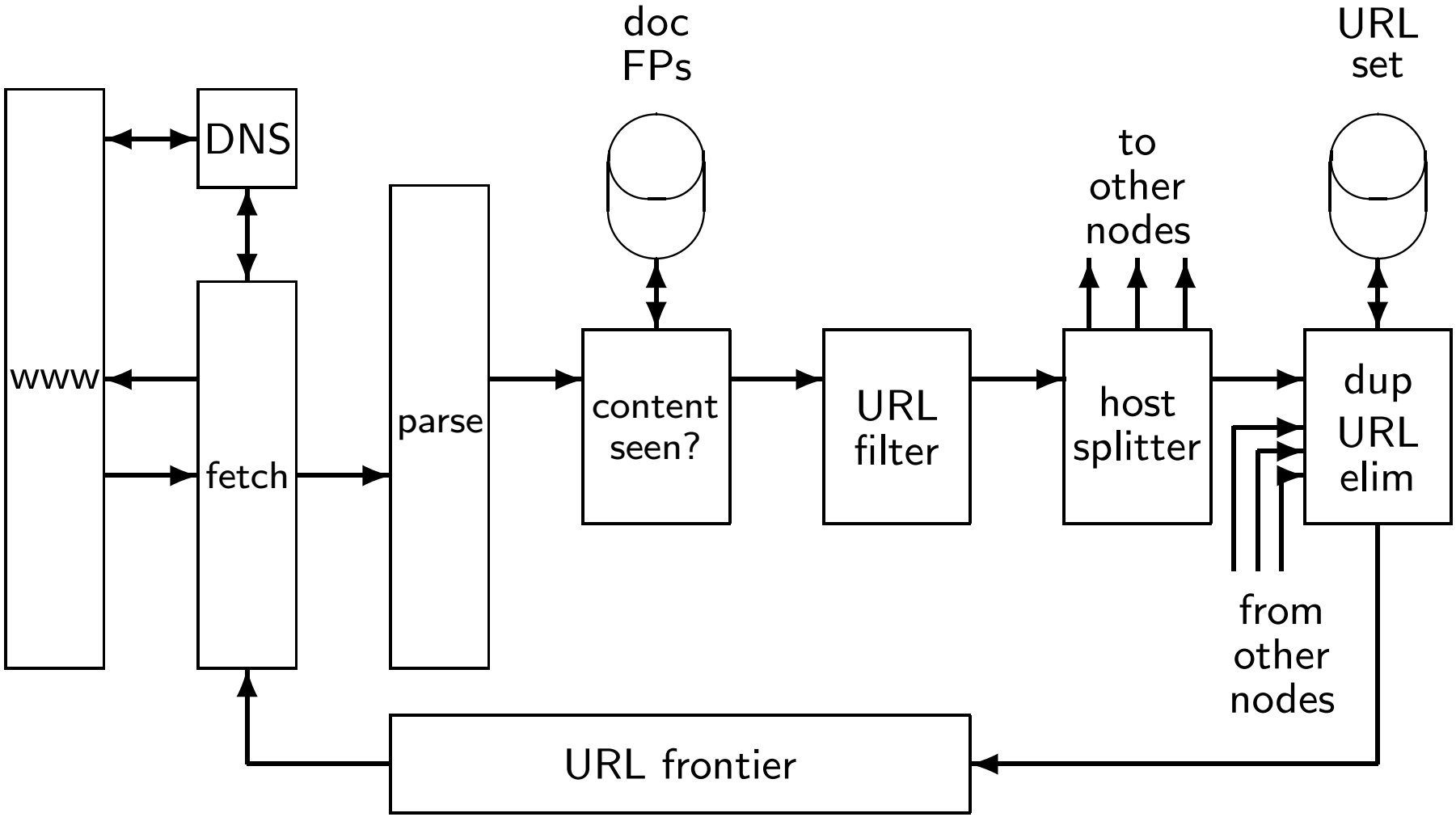
URL frontier



Distributing the crawler

1. Run multiple crawl threads, potentially at different nodes
2. Usually geographically distributed nodes
3. Partition hosts being crawled into nodes

Distributed crawl architecture



Reading

Reading

Please read chapters 19 and 20 of Information Retrieval Book.