

Web Crawling

Introduction to Information Retrieval

INF 141/ CS 121

Donald J. Patterson

Content adapted from Hinrich Schütze

<http://www.informationretrieval.org>



Robots.txt - Exclusion

- Protocol for giving spiders (“robots”) limited access to a website
- Source: <http://www.robotstxt.org/wc/norobots.html>
- Website announces what is okay and not okay to crawl:
 - Located at <http://www.myurl.com/robots.txt>
 - This file holds the restrictions



Robots.txt Example

- <http://www.ics.uci.edu/robots.txt>

```
User-agent: MOMspider          # The Multi-Owner Maintenance Spider
Disallow: /cgi-bin/           # Script files
Disallow: /Admin/MOM/         # Local MOMspider output
Disallow: /~fielding/MOM/     # Local MOMspider output
Disallow: /TR/                # Dienst Technical Report Server
Disallow: /Server/            # Dienst Technical Report Server
Disallow: /Document/          # Dienst Technical Report Server
Disallow: /MetaServer/        # Dienst Technical Report Server
Disallow: /~eppstein/pubs/cites/ # Eppstein Database
Disallow: /~fiorello/pvt/     # Private pages

User-agent: *                  # All other spiders should avoid
Disallow: /cgi-bin/           # Script files
Disallow: /Test/              # The test area for web experimentation
Disallow: /Admin/             # Huge server statistic logs
Disallow: /TR/                # Dienst Technical Report Server
Disallow: /Server/            # Dienst Technical Report Server
Disallow: /Document/          # Dienst Technical Report Server
Disallow: /MetaServer/        # Dienst Technical Report Server
Disallow: /~fielding/MOM/     # Local MOMspider output
Disallow: /~kanderso/hidden   # Ken Anderson's stuff
Disallow: /~eppstein/pubs/cites/ # Eppstein Database
Disallow: /~fiorello/pvt/     # Private pages
Disallow: /~dean/
Disallow: /~wwwoffic/
Disallow: /~ucounsel/
Disallow: /~sao/
Disallow: /~support/
Disallow: /~icsdb/
Disallow: /bin/
```

Sitemaps - Inclusion

- <https://www.google.com/webmasters/tools/docs/en/protocol.html#sitemapXMLExample>

```
<?xml version="1.0" encoding="UTF-8"?>
<urlset xmlns="http://www.sitemaps.org/schemas/sitemap/0.9">

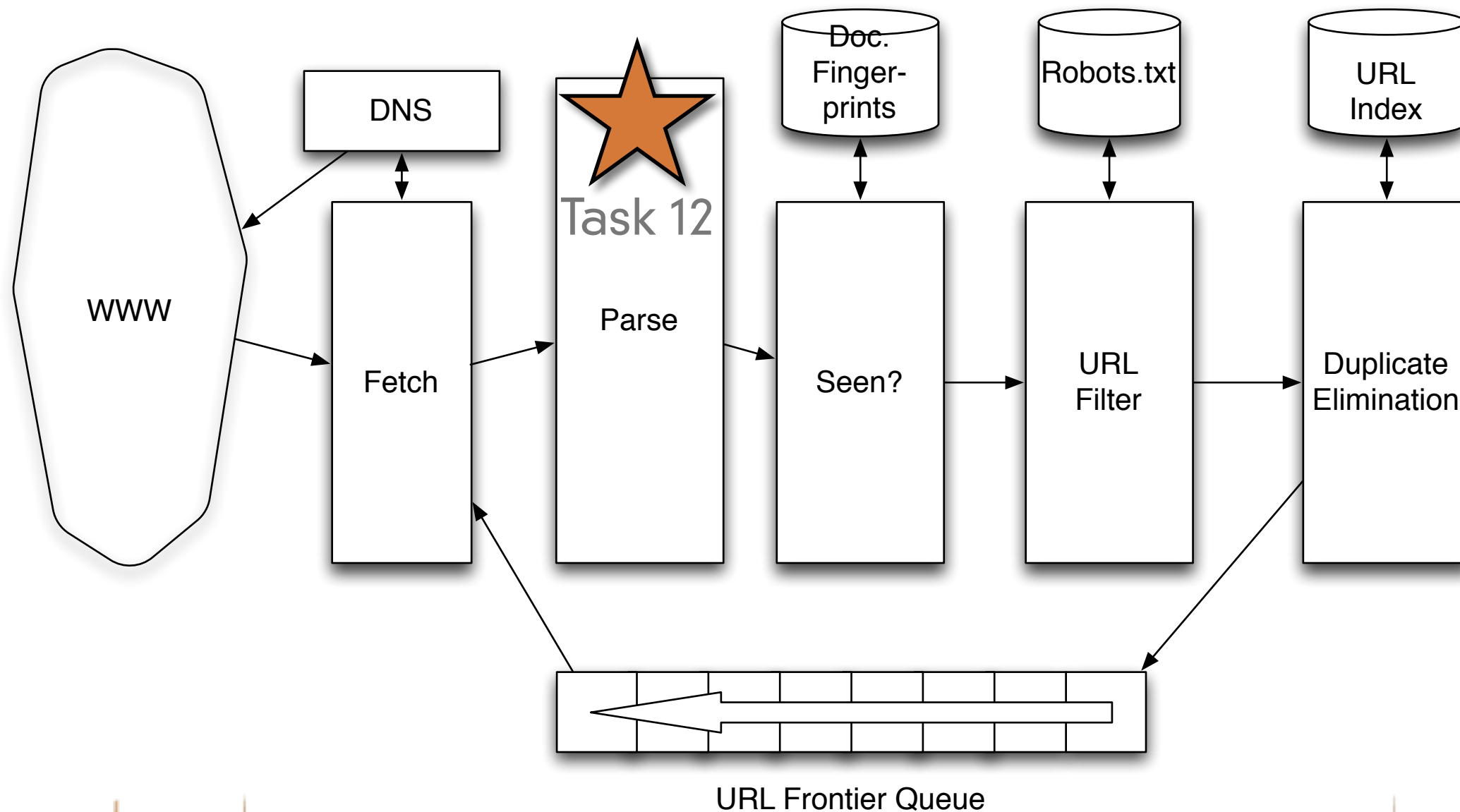
  <url>
    <loc>http://www.example.com/</loc>
    <lastmod>2005-01-01</lastmod>
    <changefreq>monthly</changefreq>
    <priority>0.8</priority>
  </url>
  <url>
    <loc>http://www.example.com/catalog?item=12&desc=vacation_hawaii</loc>
    <changefreq>weekly</changefreq>
  </url>
  <url>
    <loc>http://www.example.com/catalog?item=73&desc=vacation_new_zealand</loc>
    <lastmod>2004-12-23</lastmod>
    <changefreq>weekly</changefreq>
  </url>
  <url>
    <loc>http://www.example.com/catalog?item=74&desc=vacation_newfoundland</loc>
    <lastmod>2004-12-23T18:00:15+00:00</lastmod>
    <priority>0.3</priority>
  </url>
  <url>
    <loc>http://www.example.com/catalog?item=83&desc=vacation_usa</loc>
    <lastmod>2004-11-23</lastmod>
  </url>
</urlset>
```

Overview

- Introduction
- URL Frontier
- Robust Crawling
 - DNS



A Robust Crawl Architecture



Processing Steps in Crawling

- Pick a URL from the frontier (how to prioritize?)
- Fetch the document (DNS lookup)
- Parse the URL
 - Extract Links
- Check for duplicate content
 - If not add to index
- For each extracted link
 - Make sure it passes filter (robots.txt)
 - Make sure it isn't in the URL frontier



Domain Name Server

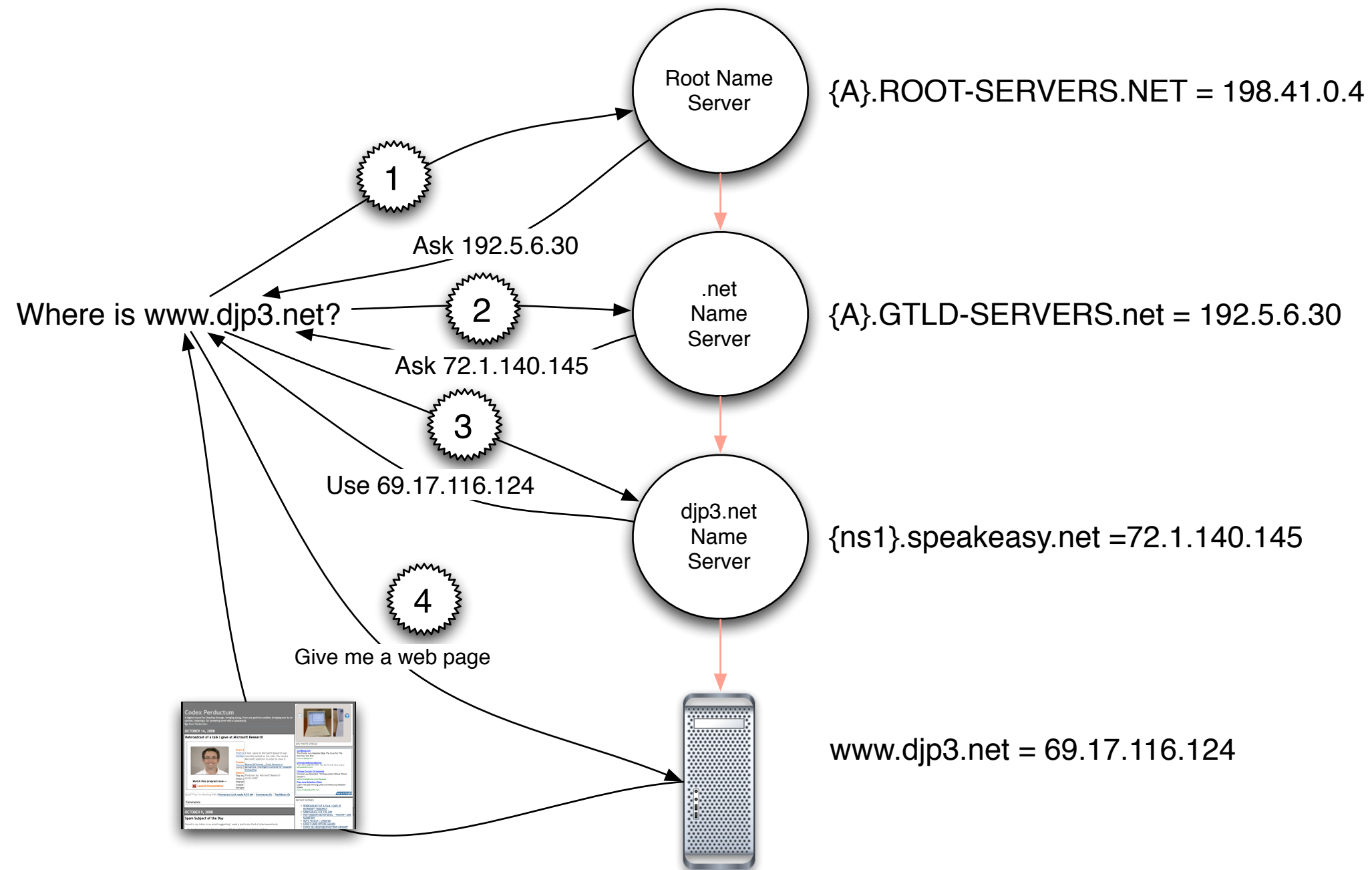
- A lookup service on the internet
 - Given a URL, retrieve its IP address
 - www.djp3.net -> 69.17.116.124
- This service is provided by a distributed set of servers
 - Latency can be high
 - Even seconds

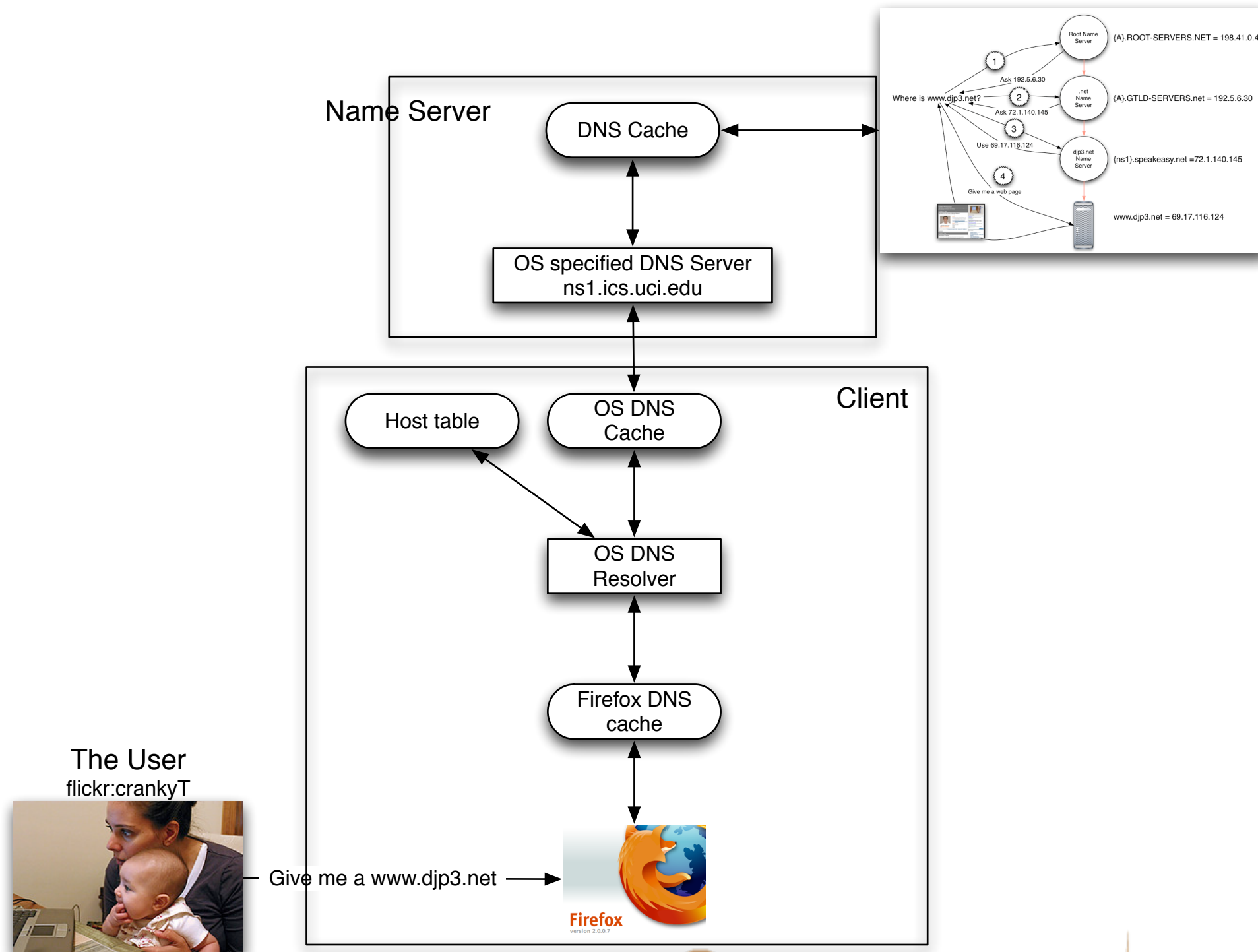


Domain Name Server

- Common OS implementations of DNS lookup are blocking
 - One request at a time
- Solution:
 - Caching
 - Batch requests
 - Custom resolvers





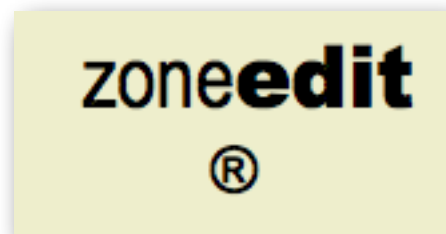
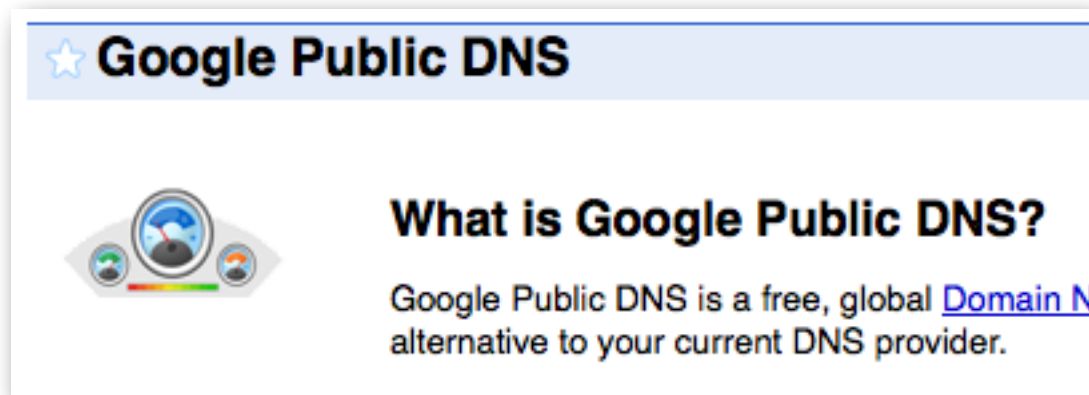


Class Exercise

- Calculate how long it would take to completely fill a DNS cache.
- How many active hosts are there?
- What is an average lookup time?
- Do the math.



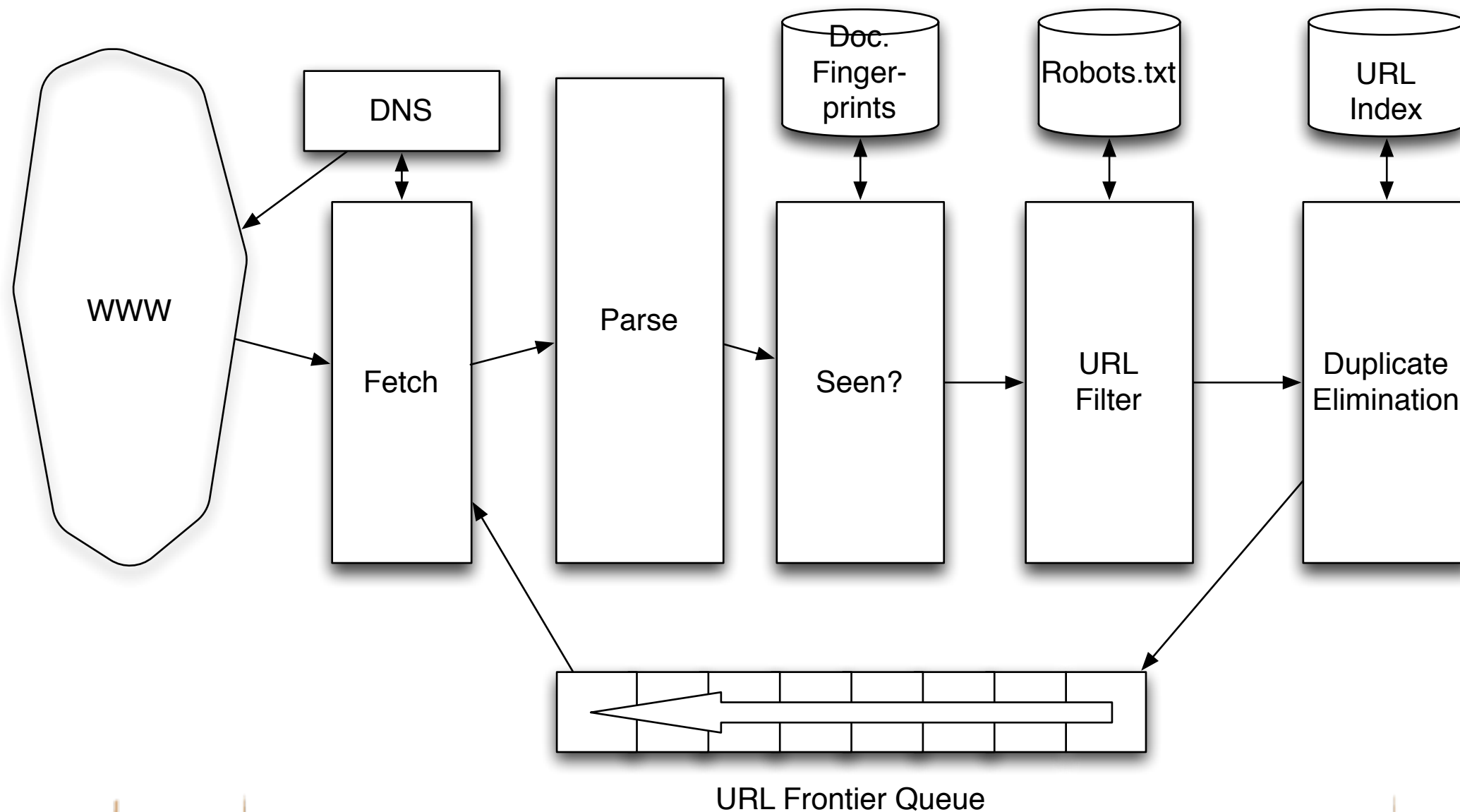
Public DNS Servers



- Why run a DNS lookup service?
- It's your administrative domain
- A public good
- It helps your other business
- You can make money on bad queries
- Mobile servers need special attention

<http://www.flickr.com/photos/lurie/298967218/>

A Robust Crawl Architecture



Parsing: URL normalization

- When a fetched document is parsed
 - some outlink URLs are **relative**
 - For example:
 - http://en.wikipedia.org/wiki/Main_Page
 - has a link to “/wiki/Special:Statistics”
 - which is the same as
 - <http://en.wikipedia.org/wiki/Special:Statistics>
 - Parsing involves normalizing (expanding) relative URLs

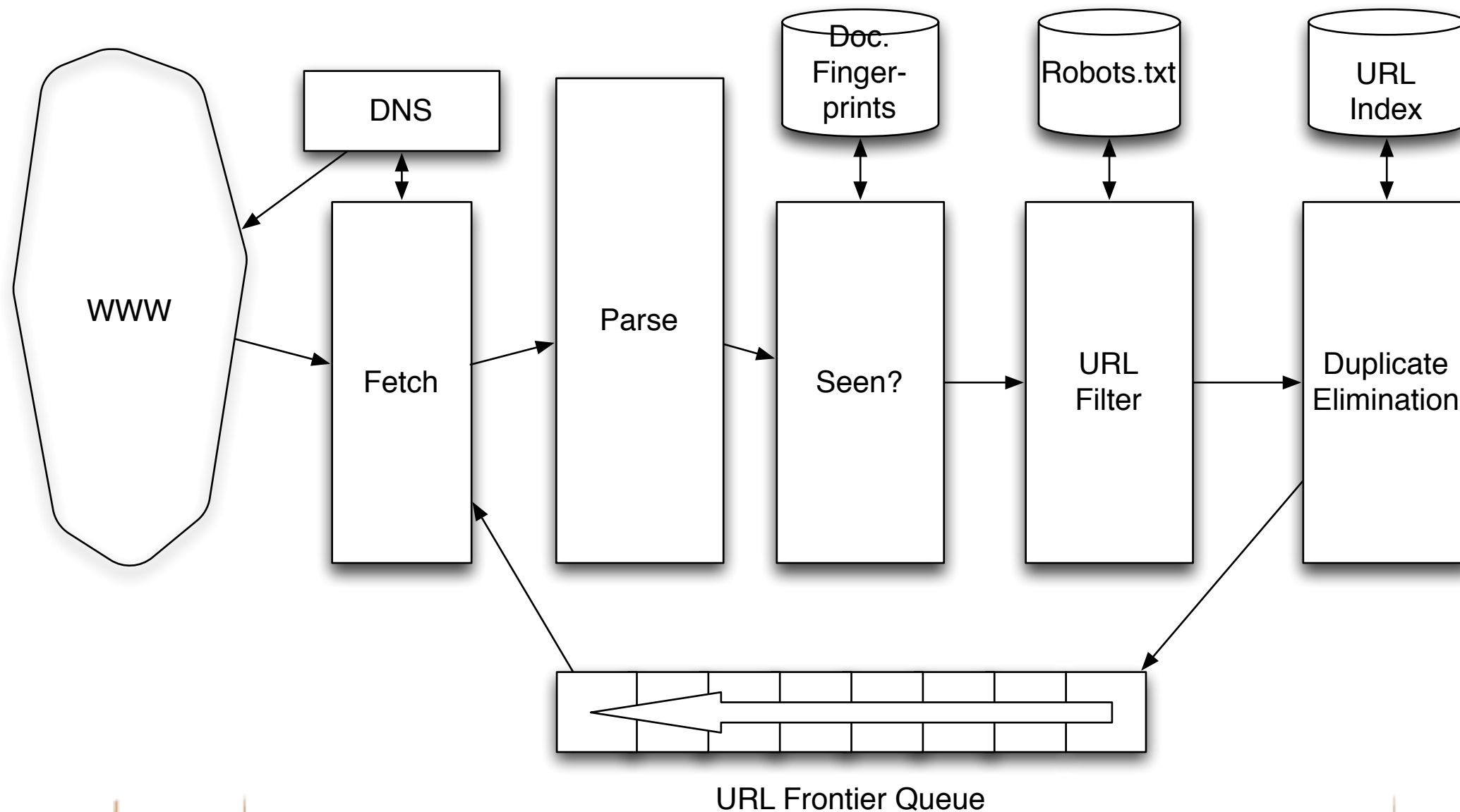


Parsing: URL normalization

- When a fetched document is parsed
 - some outlink URLs are **protocol-relative**
 - For example:
 - <http://www.starbucks.com/>
 - has a "`<script src="//cdn.optimizely.com/js/6558036.js"></script>`"
 - which matches the protocol used to load it
 - "`http:`" or "`https:`" or "`file:`" [//cdn.optimizely.com/js/6558036.js](http://cdn.optimizely.com/js/6558036.js)



A Robust Crawl Architecture

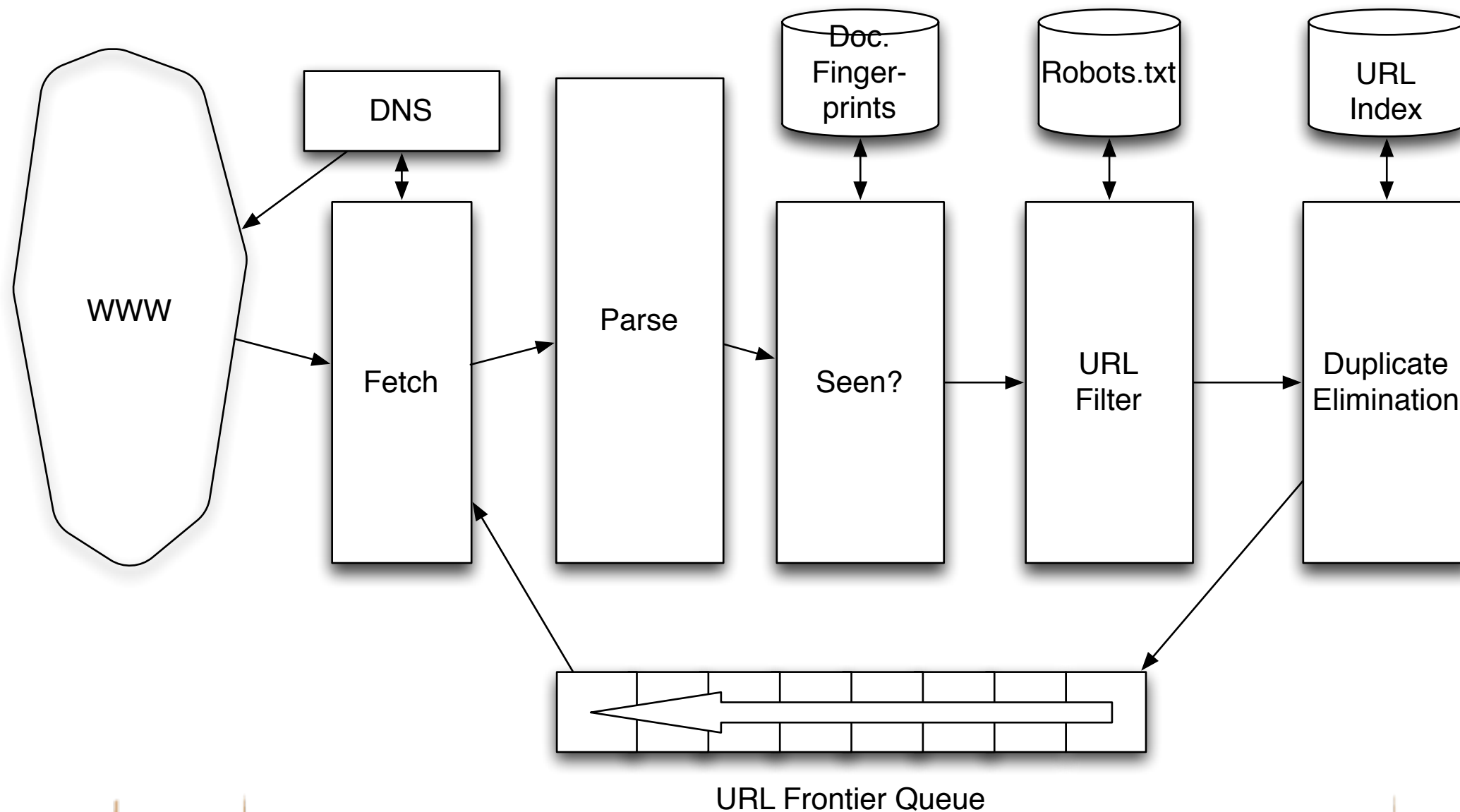


Content Seen?

- Duplication is widespread on the web
- If a page just fetched is already in the index, don't process it any further
- This can be done by using document **fingerprints**/shingles
 - A type of approximate hashing scheme
 - Similar to watermarking, SIFT features, etc.



A Robust Crawl Architecture

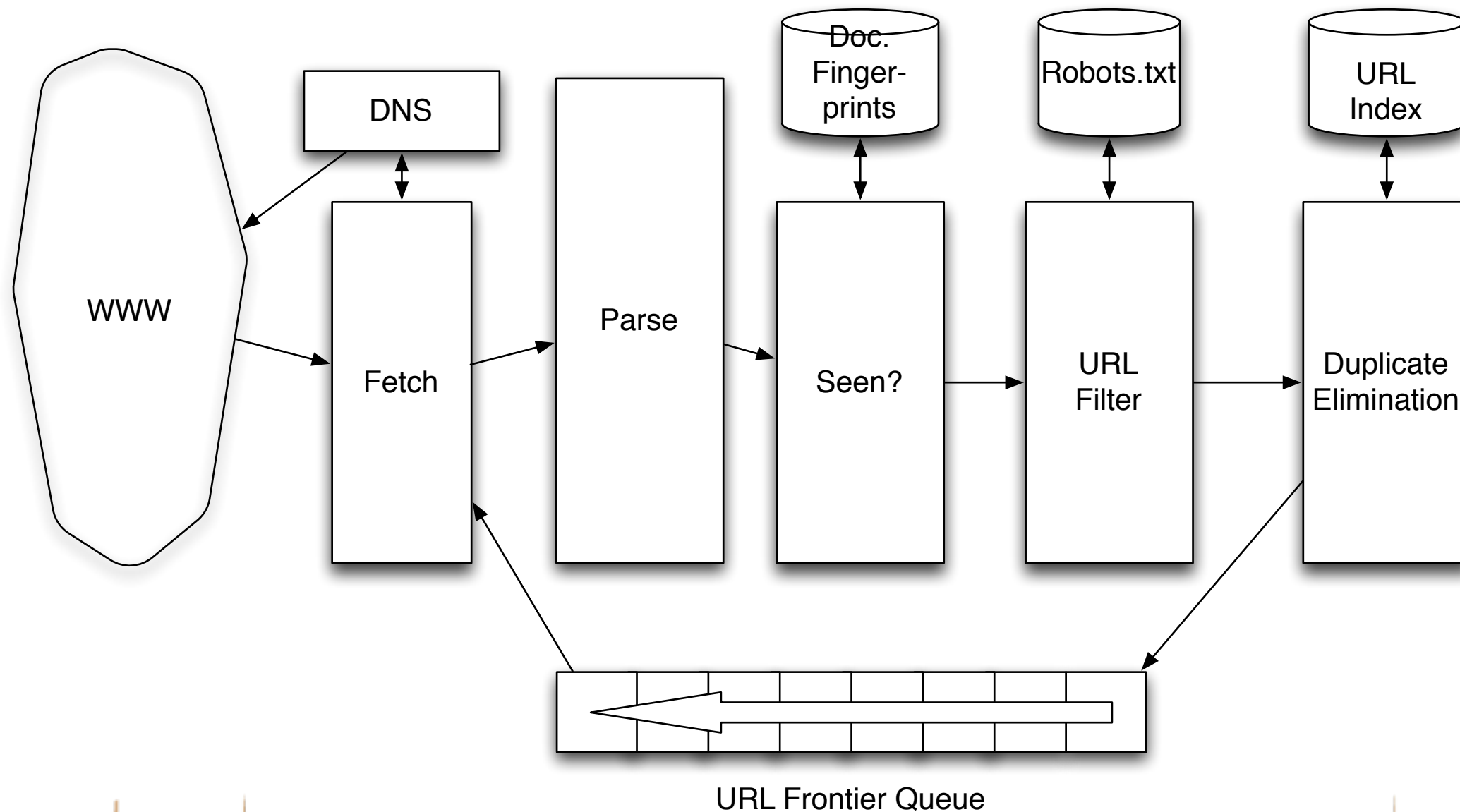


Compliance with webmasters wishes...

- Robots.txt
 - Filters is a regular expression for a URL to be excluded
 - How often do you check robots.txt?
 - Cache to avoid using bandwidth and loading web server
- Sitemaps
 - A mechanism to better manage the URL frontier



A Robust Crawl Architecture



Duplicate Elimination

- For a one-time crawl
 - Test to see if an extracted, parsed, filtered URL
 - has already been sent to the frontier.
 - has already been indexed.
- For a continuous crawl
 - See full frontier implementation:
 - Update the URL's priority
 - Based on staleness
 - Based on quality
 - Based on politeness



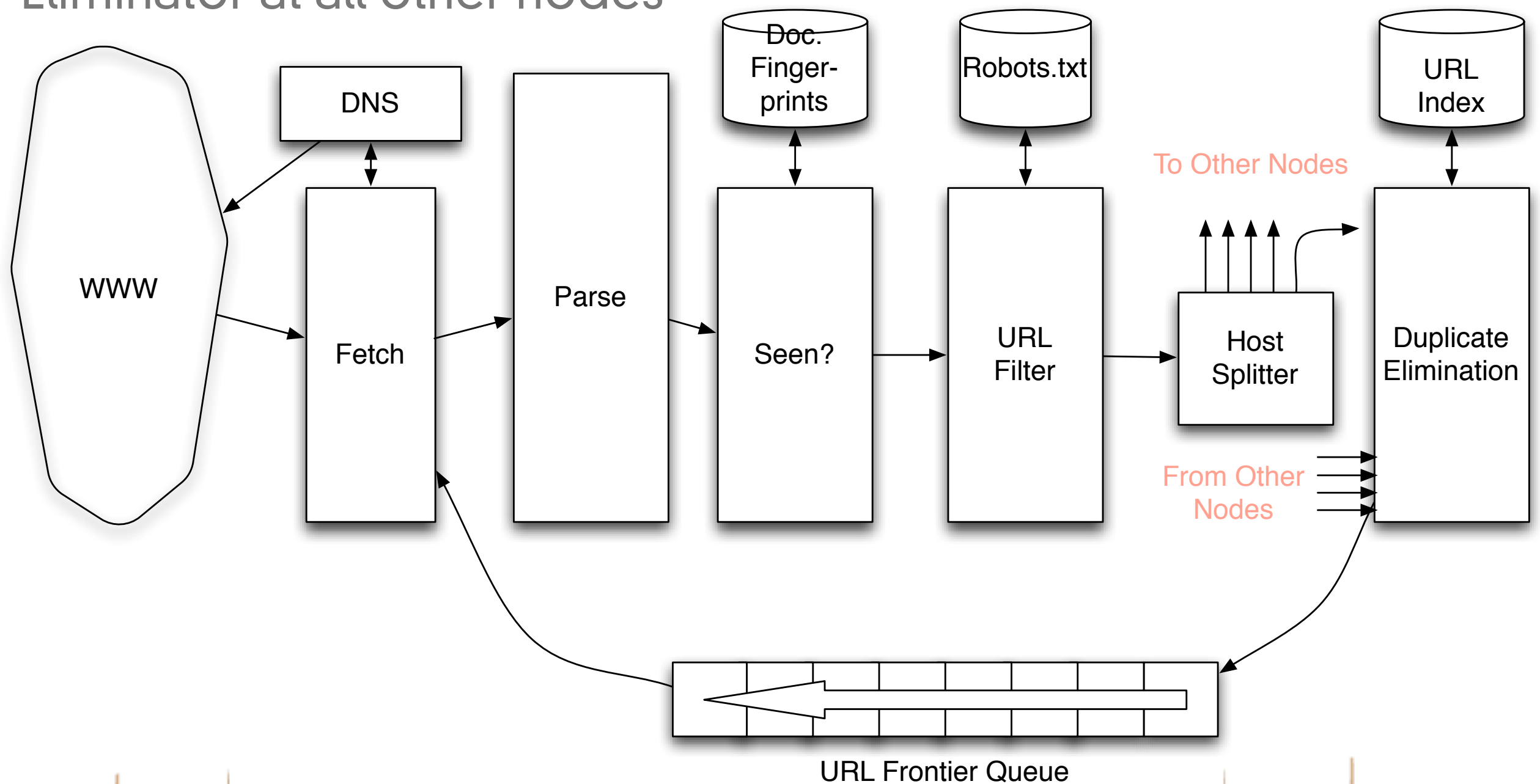
Distributing the crawl

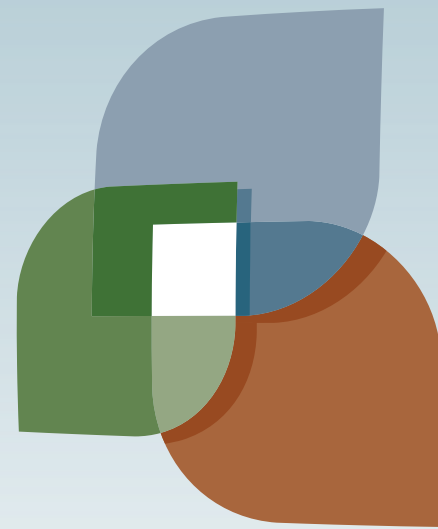
- The key goal for the architecture of a distributed crawl is **cache locality**
- We want multiple crawl threads in multiple processes at multiple nodes for robustness
 - Geographically distributed for speed
- Partition the hosts being crawled across nodes
 - Hash typically used for partition
- How do the nodes communicate?



Robust Crawling

The output of the URL Filter at each node is sent to the Duplicate Eliminator at all other nodes





L U C I

