Announcements

- Assignment 2 released
  - Due Wednesday Oct 23
  - Please start early!!

Crawling the web

Crawler Architecture

- If the frontier is a queue, the graph is traversed in breadth-first search (BFS) order.
- If the frontier is a stack, the graph is traversed in depth-first search (DFS) order.

Crawler performance

- Coverage
  - Can the crawler find every page?
- Freshness
  - How frequently can a crawler revisit?
- Trade-off!
  - Crawlers need to prioritize which pages to visit

Freshness

- Pages are constantly added, deleted, and modified
- Freshness is the percentage of pages for which the search engine has a current copy
  - Staleness is the % for which we have an outdated copy
  - Age is the # of days that an average page is out-of-date
  - Search engines try to maximize freshness
  - Crawlers revisit pages they have already crawled to see if they have changed

Freshness vs. Age

- Search engines try to maximize freshness.
- Crawlers revisit pages they have already crawled to see if they have changed.
**Age**

- Expected age of a page \( t \) days after it was last crawled:
  \[
  \text{Age}(\lambda, t) = \int_0^t P\text{(page changed at time } x \text{)} (t - x) dx
  \]

- Page updates generally follow a Poisson distribution
  - time until the next update is governed by an exponential distribution
  \[
  \text{Age}(\lambda, t) = \int_0^t \lambda e^{-\lambda x} (t - x) dx
  \]

**Estimated age**

- Given an estimate of how often a page changes (\( \lambda \)), we can estimate the current age of a page
  
  e.g. Expected age with \( \lambda = 1/7 \) (one change per week):

**Checking freshness**

- HTTP protocol has a special request type called HEAD that makes it easy to check for page changes
  - returns information about page, not page itself

**Other crawling strategies**

- We’ve seen two crawling strategies so far
  - which differ in the order that the web is crawled
  - but neither one looks at the content of pages
- Modern “smart” crawlers prioritize links based on a variety of factors
  - e.g. anchor text, text surrounding link, age of server, estimates of page change rate, etc.

**Smart crawlers**

- Modern search engines use sophisticated algorithms to crawl, to balance coverage and freshness
  - They don’t use simple BFS or DFS exploration anymore
- They do this by “biasing” the exploration, preferring some links to others
  - E.g. Bias towards popular pages, highest rate of change, closest to seeds, unknown servers, ...
- The exact techniques are closely guarded secrets
Crawler Architecture

Of all the URLs in the frontier, which do you choose?

Stacks vs. Queues

- **Stack**
- **Queue**
- **Priority Queue**
  - You put (item, priority) pairs into queue
  - You remove the highest-priority item from the queue

**Priority queue**

- Priority queue is a best-first data structure
  - When you add something to a PQ, you give an importance (priority number)
  - When you remove something, the PQ gives you the highest priority item in the queue
  - If multiple items have the same priority, it returns the one that was added first

**Priority queue examples**

- Hospital emergency room
  - Incoming patients see a triage nurse who assigns a priority to each patient. Highest need patients are seen first.
- Airplane boarding
  - First class, Business class, Coach
  - FIFO within each class
- Operating system scheduling
  - Important system jobs (memory management, etc) are given priority over user tasks

**Smart Crawler Architecture**

Store frontier in a priority queue, so that the highest priority link is chosen here.

**PQ implementation**

- Priority queues are typically implemented using a *heap*
  - A binary tree with a special property: The highest-priority element of any subtree is always at the root of the subtree.
  - Learn more in a data structures class...
InfoSpiders Algorithm

Foreach agent thread:
- Pick & follow link from local frontier
- Evaluate new links, merge frontier
- Adjust link estimator
  \( E := E + \text{payoff} - \text{cost} \)
  - If \( E < 0 \):
    Die
  - Elseif \( E > \text{Selection} \_ \text{Threshold} \):
    Clone offspring
    Split energy with offspring
    Split frontier with offspring
    Mutate offspring

Q-learning

- Compare estimated relevance of visited document with estimated relevance of link followed from previous page
- Teaching input:
  \( E(D) = \mu \max_{\lambda x} \lambda x \)
Indexing web documents

Indexing

Processing Text

- First step: converting each document to a set of **index terms**
- Why not simply make each word an index term?
  - Matching exact string of characters is too restrictive
    - a query for “regulation” should probably return documents with “regulations” and “regulated”
  - Not all words are of equal value in a search
  - Sometimes not clear where words begin and end
    - Not even clear what a word is in some languages

Statistics of text

- There’s a huge number & variety of words, but they exhibit some predictable patterns
  - Can be modeled using statistics
  - Search engines depend heavily on these pattern
- Q: Given a document, how would you choose the few most important words?
  - i.e. the words that best describe a document

Finding important words

- A document’s most descriptive words appear frequently in the document, but infrequently in the English language

Finding important words

- A document’s most descriptive words appear frequently in the document, but infrequently in the English language
**Finding important words**

- A document’s most descriptive words appear frequently in the document, but infrequently in the English language.

**Zipf’s Law**

- Distribution of word frequencies is very skewed — a few words occur often, most words rarely occur.

- Zipf’s “law” — observation that the frequency of a word in roughly proportional to 1/r, where r is its rank.
  - E.g. the most common word, “the”, occurs...
    - twice as often as the second most common word (“of”)
    - 3x as often as the third most common word (“and”)
    - 10x as often as the tenth most common word (“it”)

**Zipf’s Law**

- Plotted on regular (linear) axes:
- Plotted on log-log axes:

**Statistics of news articles (AP89)**

- 84,678 Total documents
- 39,749,179 Total word occurrences
- 198,763 Vocabulary size
- 70,064 Words occurring once
- 198,763 Vocabulary size
- 70,064 Words occurring once

**Power law distribution** $f(x) = ax^k$

- Plotted on regular (linear) axes:
- Plotted on log-log axes:

**Contrast with the Uniform Distribution** $f(x) = a$

**Contrast with the Gaussian (Normal) Distribution**
on the Moon

Solar flares

wars (1816-1980)

richest individuals 2003

US family names 1990

US cities 2003

Power law distribution

DistribuBon	
  of	
  human	
  heights

(of 16	
  year	
  old	
  males)

DistribuBon	
  of	
  human	
  IQ	
  scores

VelociBes	
  of	
  stars

[Schodel2009]