

*introduction to*  
**WEB CRAWLING**  
*& extraction* *by Nate Murray*

**WHO AM I ?**

# ▶ **Nate Murray**

**AT&T Interactive (Yellowpages.com)**

**TB-scale data since 2009**

**Various crawlers since 2005**

*what is*  
**WEB CRAWLING?**

*definition:*

▶ **web crawler**

*a program that browses the web.*

*definition:*

# ▶ **web extraction**

*transforming **unstructured** web data into  
**structured** data*

*definition:*

# ▶ **web extraction**

*transforming **semistructured** web data into  
**structured** data*

# motivation

A screenshot of a Delicious bookmark page. The main title is "Scrapy | An open source web scraping framework for Python". Below the title, there's a "History" section showing a list of saved items with their dates, titles, and tags. On the right side, there's a "Tags" section with a list of tags and their counts, such as "python" (400), "web" (301), and "framework" (236).

A screenshot of the maru website. The header includes "CALIFORNIA CUISINE" and "DOWNTOWN CENTER DR. SUITE 100, VALLEJO, CA 94590". The main content area features the "maru" logo, contact information, location details, hours of operation, and a map showing the restaurant's location. The footer mentions "Website designed by o.fox" and a Facebook link.

A screenshot of a YouTube video player. The video is titled "Richard Feynman - Quantum Mechanics". The video player shows a man (Richard Feynman) speaking. Below the video, there are suggestions for other videos, a "Like" button, and a "Share" button. The video has 34,688 views.

A screenshot of a product page for "Combo Kits". The page shows a grid of various power tool kits, including Milwaukee and DeWalt brands. Each kit is displayed with its name, price, and a star rating. The page also includes a "Refine By" section with filters for Rating, Price, Brand, and Cordless.



# motivation: *bookmark buddies*

The screenshot shows a Delicious bookmark page for the URL 'scrapy.org'. The page title is 'Scrapy | An open source web scraping framework for Python'. The user is logged in as 'jashmenn'. The page displays a 'History' tab with a list of users who have bookmarked the page, including Ludovic Gasc, s41ted, philippwinkler, ideola, Eduardo.Lucas, Fabio Malini, Andrew Perry, Eddie Welker, Taras Shymbra, berg\_pe, and rafaelbco. Each entry shows the date, user name, and associated tags. A 'Tags' sidebar on the right lists the most common tags and their counts, such as 'python' (825), 'scraping' (454), and 'crawler' (405).

delicious Home Bookmarks People Tags Hi jashmenn Inbox Settings Help Sign Out

Everyone's Bookmarks for:  
**Scrapy** | An open source web scraping framework for Python  
scrapy.org/

**History** Notes

Saved 1019 times, first saved by Alex O'Connor on 30 Dec 08. [View Chart](#)

17 AUG 11 [Ludovic Gasc \(GMLudo\)](#) python web scraping library

16 AUG 11 [s41ted](#) software tools opensource web scraping internet data

[philippwinkler](#) programming data

15 AUG 11 [ideola](#) programming python

14 AUG 11 [Eduardo.Lucas](#) python framework scraping opensource opendata

[Fabio Malini](#) crawler

11 AUG 11 Scrapy is a fast high-level screen scraping and web crawling framework, used to crawl websites and extract structured data from their pages. It can be used for a wide range of purposes, from data mining to monitoring and automated testing.

[Andrew Perry](#) python django scraping web spider software tools

09 AUG 11 [Eddie Welker](#) programming python django web development crawler

08 AUG 11 [Taras Shymbra](#) opensource development web analysis nlp

07 AUG 11 [berg\\_pe](#) python web crawler

04 AUG 11 [rafaelbco](#) python crawler scraping spider

**Tags**

Top Tags

python	825
scraping	454
crawler	405
web	381
spider	285
framework	236
programming	195
screenscraping	194
webscraping	187
tools	162
webdev	115
opensource	114
scrapy	83
search	68
crawling	65
library	61
software	60
datamining	55
html	47
scraper	39
tool	35
scrape	29
code	27
webcrawler	21
data	21
internet	19

# motivation: *bookmark buddies*

The screenshot shows the Delicious website interface. At the top, there are navigation tabs for Home, Bookmarks, People, and Tags. A search bar is visible on the right. The main content area displays a bookmark for 'Scrapy' with the title 'An open source web scraping framework for Python' and the URL 'scrapy.org/'. Below this, a 'History' section shows a list of users who have saved the bookmark, including Ludovic Gasc, s41ted, philippwinkler, ideola, Eduardo.Lucas, Fabio Malini, Andrew Perry, Eddie Welker, Taras Shymbra, berg\_pe, and rafaelbco. Red arrows point from the text 'URL Title' to the bookmark's title and URL, and from 'Users' to the list of users. A red box highlights the 'History' section.

delicious

Home Bookmarks People Tags

Search Delicious Search

Everyone's Bookmarks for:

**Scrapy** | An open source web scraping framework for Python  
scrapy.org/

Save this bookmark  
Share this bookmark  
Look up another URL

Tags

Top Tags

Tag	Count
python	825
scraping	454
crawler	405
web	381
spider	285
framework	236
programming	195
screenscraping	194
webscraping	187
tools	162
webdev	115
opensource	114
scrapy	83
search	68
crawling	65
library	61
software	60
datamining	55
html	47
scraper	39
tool	35
scrape	29
code	27
webcrawler	21
data	21
internet	19

History

Saved 1019 times, first saved by Alex O'Connor on Dec 08. View Chart

17 AUG 11 Ludovic Gasc (GMLudo)

16 AUG 11 s41ted software tools opensource web scraping internet data

philippwinkler programming data

15 AUG 11 ideola programming python

14 AUG 11 Eduardo.Lucas python framework scraping opensource opendata

Fabio Malini crawler

11 AUG 11 Scrapy is a fast high-level screen scraping and web crawling framework, used to crawl websites and extract structured data from their pages. It can be used for a wide range of purposes, from data mining to monitoring and automated testing.  
Andrew Perry python django scraping web spider software tools

09 AUG 11 Eddie Welker programming python django web development crawler

08 AUG 11 Taras Shymbra opensource development web analysis nlp

07 AUG 11 berg\_pe python web crawler

04 AUG 11 rafaelbco python crawler scraping spider

URL Title

Users

# motivation:

A screenshot of a Delicious bookmark page. The main title is "Scrapy | An open source web scraping framework for Python". The page shows a list of bookmarks with columns for date, title, and tags. A "Tags" sidebar on the right lists various tags like "python", "web", "scraping", and "framework" with their respective counts. The top navigation bar includes "Home", "Bookmarks", "People", and "Tags".

A screenshot of the Maru restaurant website. The header includes "CALIFORNIA CUISINE" and the address "34250 Town Center Dr, Suite 100, Valencia, CA 91355". The main content area is divided into "About Maru", "Location", "Hours", and "Contact". A map on the right shows the restaurant's location. The footer includes "Website designed by o.fox" and a Facebook link.

A screenshot of a YouTube video player. The video shows Richard Feynman speaking. The video title is "Richard Feynman - Quantum Mechanics". The video has 34,688 views. The page includes a list of suggestions for other videos by Feynman, such as "Richard Feynman on Dual", "Feynman on EPR's paradox", and "What keeps a train on the track?".

A screenshot of a product page for "Combo Kits". The page features a grid of product images and descriptions. The products include "Milwaukee 2801-22 18V Cordless 2-1/2 Ton Compressor Kit", "DEWALT DCR260L 18V Cordless Compact Lithium-Ion 2-Ton Combo Kit", "Black & Decker 9C168-09 18V 10-Piece Home Project Kit", and "Makita LXT316 18V Cordless LXT Lithium-Ion 3-Piece Combo Kit". The page also includes filters for "Rating", "Price", "Brand", and "Cordless".

# motivation:

# business hours

CALIFORNIA CUISINE

24250 TOWN CENTER DR. SUITE 180, VALENCIA, CA 91355  
(661) 290-2595

## maru

About Maru  
Menu  
Reservations  
Press  
[Contact](#)

### CONTACT

Maru is located in the heart of Valencia on Town Center Drive.

#### Location

24250 Town Center Dr. Suite 180  
Valencia, CA 91355  
Tele: (661) 290-2595  
[contact@maruvalencia.com](mailto:contact@maruvalencia.com)

#### Hours

Closed on Mondays.

##### Tues - Friday

Lunch	11:30AM - 2:30PM
Dinner	5:30 - 10PM

##### Saturday

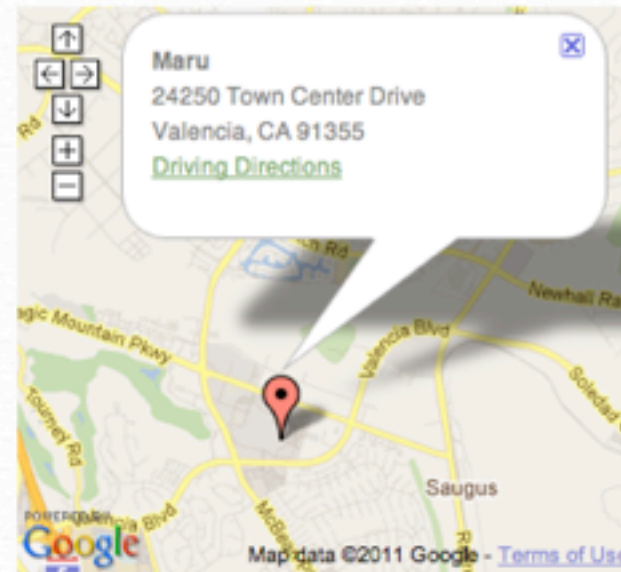
Lunch	12 - 2:30PM
Dinner	5 - 10PM

##### Sunday

Dinner	5 - 9PM
--------	---------

#### Credit Cards

We accept American Express, Visa, and Mastercard.



Website designed by *electiv*

facebook



# motivation:

# business hours

CALIFORNIA CUISINE

24250 TOWN CENTER DR. SUITE 180, VALENCIA, CA 91355  
(661) 290-2595

## maru

About Maru  
Menu  
Reservations  
Press  
Contact

**CONTACT**

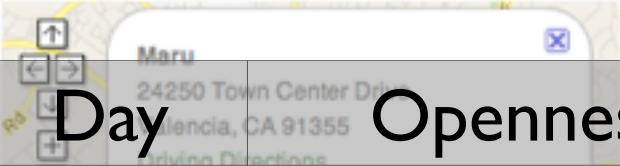
Maru is located in the heart of Valencia on Town Center Drive.

**Location**  
24250 Town Center Dr. Suite 180  
Valencia, CA 91355  
Tele: (661) 290-2595  
[contact@maruvalencia.com](mailto:contact@maruvalencia.com)

**Hours**

Closed on Mondays.	
<i>Tues - Friday</i>	
Lunch	11:30AM - 2:30PM
Dinner	5:30 - 10PM
<i>Saturday</i>	
Lunch	12 - 2:30PM
Dinner	5 - 10PM
<i>Sunday</i>	
Dinner	5 - 9PM

**Credit Cards**  
We accept American Express, Visa, and Mastercard



Day	Openness	
Mon	Closed	
Tue	11:30-14:30	17:30-22:00
Wed	11:30-14:30	17:30-22:00
Thur	11:30-14:30	17:30-22:00
Fri	11:30-14:30	17:30-22:00
Sat	12:00-14:30	17:00-22:00
Sun	-	17:00-21:00

Website designed by *electiv*

facebook

# motivation:

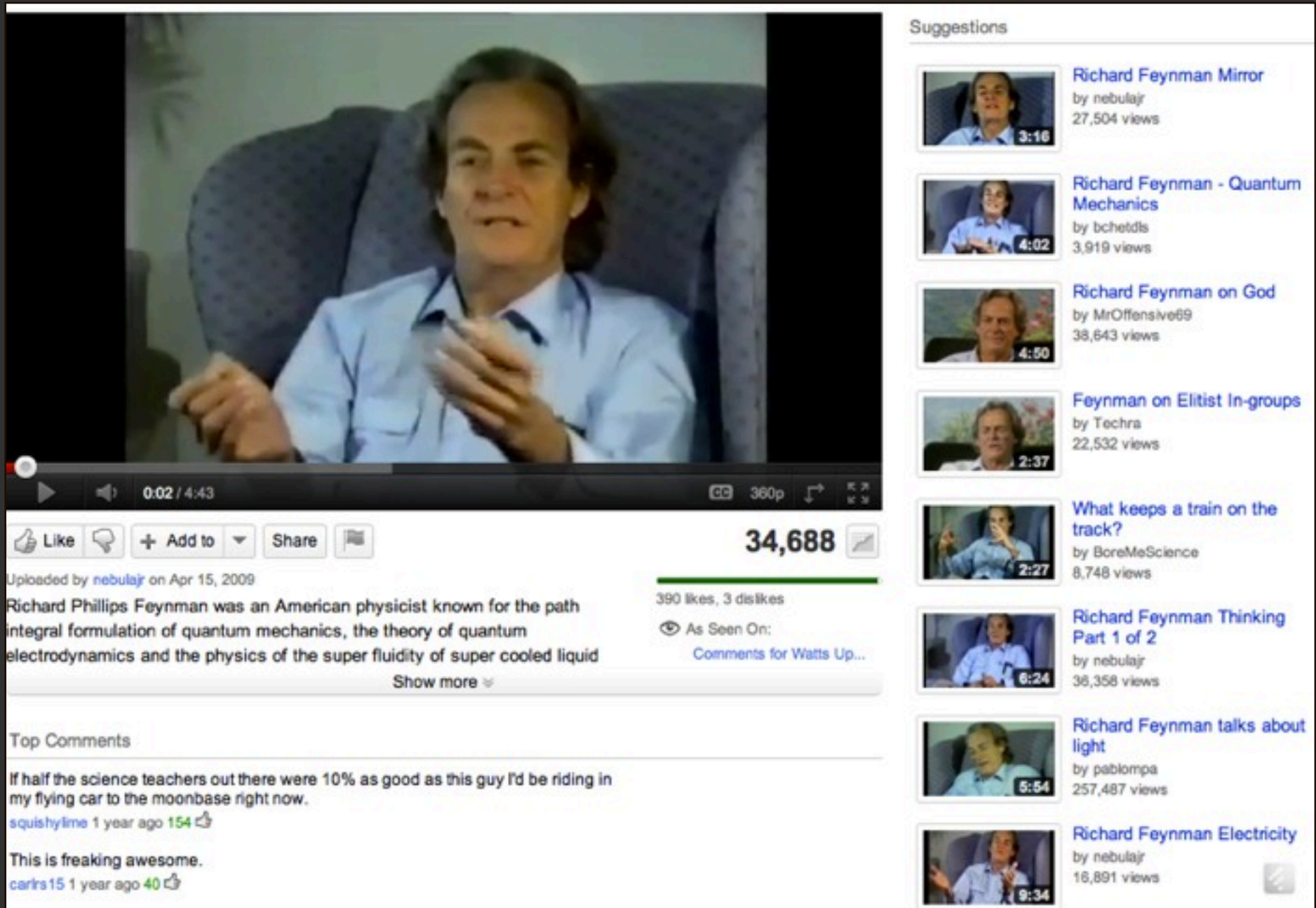
A screenshot of a Delicious bookmark page. The main heading is "Scrapy | An open source web scraping framework for Python". Below this, there is a "History" section showing a list of saved items with their titles, authors, and tags. A "Tags" section on the right lists various tags like "python", "web", "scraping", and "framework" with their respective counts. The page is clean and organized, with a clear focus on the bookmarked content.

A screenshot of the Maru restaurant website. The page features a clean, modern design with a white background and orange accents. The header includes the restaurant's name "maru" and its location in Valencia, CA. A "CONTACT" section provides details about the restaurant, including its address, phone number, and email. A map shows the restaurant's location. The page is well-structured and easy to navigate.

A screenshot of a YouTube video player. The video is titled "Richard Feynman - Quantum Mechanics" and is part of a series. The video player shows a man in a blue shirt speaking. To the right of the video, there are several suggestions for related videos, each with a thumbnail and a brief description. The page is clean and easy to navigate.

A screenshot of a product page for "Combo Kits". The page features a grid of product images and descriptions. Each product is accompanied by a price tag and a rating. The page is well-organized and easy to navigate. The products are arranged in a grid, and the information is presented in a clear and concise manner.

# motivation: *recommend videos*



Richard Phillips Feynman was an American physicist known for the path integral formulation of quantum mechanics, the theory of quantum electrodynamics and the physics of the super fluidity of super cooled liquid

34,688

390 likes, 3 dislikes

As Seen On:  
[Comments for Watts Up...](#)

Uploaded by [nebulajr](#) on Apr 15, 2009

Like Add to Share

Top Comments

If half the science teachers out there were 10% as good as this guy I'd be riding in my flying car to the moonbase right now.  
[squishylime](#) 1 year ago 154

This is freaking awesome.  
[carlrs15](#) 1 year ago 40

Suggestions

- [Richard Feynman Mirror](#)  
by [nebulajr](#)  
27,504 views  
3:16
- [Richard Feynman - Quantum Mechanics](#)  
by [bchelds](#)  
3,919 views  
4:02
- [Richard Feynman on God](#)  
by [MrOffensive69](#)  
38,643 views  
4:50
- [Feynman on Elitist In-groups](#)  
by [Techra](#)  
22,532 views  
2:37
- [What keeps a train on the track?](#)  
by [BoreMeScience](#)  
8,748 views  
2:27
- [Richard Feynman Thinking Part 1 of 2](#)  
by [nebulajr](#)  
36,358 views  
6:24
- [Richard Feynman talks about light](#)  
by [pablomp](#)  
257,487 views  
5:54
- [Richard Feynman Electricity](#)  
by [nebulajr](#)  
16,891 views  
9:34



# motivation: *recommend videos*

Richard Phillips Feynman was an American physicist known for the path integral formulation of quantum mechanics, the theory of quantum electrodynamics and the physics of the super fluidity of super cooled liquid

34,688

390 likes, 3 dislikes

As Seen On:  
Comments for Watts Up...

Top Comments

If half the science teachers out there were 10% as good as this guy I'd be riding in my flying car to the moonbase right now.

**squishylime** 1 year ago 154

**carls15** 1 year ago 40

**Users**

Suggestions

- Richard Feynman Mirror by nebulajr 27,504 views 3:16
- Richard Feynman - Quantum Mechanics by bchelds 3,919 views 4:02
- Richard Feynman on God by MrOffensive69 38,643 views 4:50
- Feynman on Elitist In-groups by Techra 22,532 views 2:37
- What keeps a train on the track? by BoreMeScience 8,748 views 2:27
- Richard Feynman Thinking Part 1 of 2 by nebulajr 36,358 views 6:24
- Richard Feynman talks about light by pablompa 257,487 views 5:54
- Richard Feynman Electricity by nebulajr 16,891 views 9:34



# motivation:

A screenshot of a Delicious bookmark page. The main title is "Scrapy | An open source web scraping framework for Python". Below the title, there's a "History" section showing a list of saved items with their titles, authors, and tags. On the right side, there's a "Tags" section with a list of tags and their counts, such as "python" (400), "web" (301), and "framework" (236).

A screenshot of the Maru restaurant website. The page features the restaurant's name "maru" in a large, stylized font. Below the name, there's a "CONTACT" section with the address "24250 Town Center Drive, Valencia, CA 91355", phone number, and email. A map is embedded on the right side, showing the location. The website is designed with a clean, modern layout and includes a Facebook link at the bottom.

A screenshot of a YouTube video page. The video player shows Richard Feynman speaking. Below the video, there's a "Suggestions" section with several video thumbnails and titles, such as "Richard Feynman - Minor", "Richard Feynman - Quantum Mechanics", and "Richard Feynman on God". The video has 34,688 views and is from the channel "PBS".

A screenshot of a product page for "Combo Kits". The page displays a grid of various power tool kits, including Milwaukee, DeWalt, and Makita. Each kit is shown with its name, price, and a star rating. The page also includes a "Refine By" section on the left with filters for Rating, Price, Brand, and Cordless. The overall layout is clean and organized, with clear product information and navigation options.

# motivation:

# vertical search

Home > Shop By Category > Combo Kits

Refine By:

## Combo Kits

Showing items 1 - 12 of 53 Results | Sort By: Most Popular | Items per page 12 | 1 2 3 4 5

**Rating**

- ★★★★★
- ★★★★★ & Up
- ★★★★★ & Up
- ★★★★★ & Up

**Price**


- \$50 - \$99.99
- \$100 - \$199.99
- \$200 - \$299.99
- \$300 - \$399.99
- \$400 - \$599.99
- \$600 - \$799.99
- \$800 - \$999.99
- \$1,000 - \$1,499.99
- \$1,500 and Up

**Brand**


- Black & Decker
- Bosch
- Bostitch
- DEWALT
- Hitachi
- Makita
- Milwaukee
- Porter-Cable

**Cordless**


- Cordless




MILN2691-22  
**Milwaukee 2691-22 18V Cordless M18 Lithium-Ion 2-Tool Combo Kit with Case**  
~~\$329.00~~ **\$199.00**  
★★★★★




DEWINDCK265L  
**DEWALT DCK265L 18V Cordless Compact Li-Ion 2-Tool Combo Kit**  
~~\$349.00~~ **\$279.00**  
★★★★★




BCKNGC18B-59  
**Black & Decker GC18B-59 18V 59-Piece Home Project Kit**  
**\$64.99**




MKTNLXT218  
**Makita LXT218 18V Cordless LXT Lithium-Ion 2-Piece Combo Kit**  
~~\$429.00~~ **\$379.99**  
★★★★★




BSTNROOFKIT2  
**Bostitch ROOFKIT2 1-3/4-In Roofing Nailer and 18 Gauge Cap Stapler Combo Kit**  
~~\$349.99~~ **\$319.99**



MKTNLXT405  
**Makita LXT405 18V Cordless LXT Lithium-Ion 4-Piece Combo Kit**  
~~\$599.99~~ **\$469.99**  
★★★★★



MKTNLXT601  
**Makita LXT601 18V Cordless LXT Lithium-Ion 6-Piece Combo Kit**  
~~\$759.99~~ **\$699.99**  
★★★★★



MKTNLXT407  
**Makita LXT407 18V Cordless LXT Lithium-Ion 4-Piece Combo Kit**  
~~\$599.99~~ **\$469.99**  
★★★★★

# motivation:

# vertical search

Home > Shop By Category > Combo Kits

Refine By: **Combo Kits**

Showing items 1 - 12 of 53 Results | Sort By: Most Popular | Items per page 12 | 1 2 3 4 5

**Rating**

★★★★★  
 ★★★★☆  
 ★★★☆☆  
 ★★☆☆☆  
 ★☆☆☆☆

**Price**









\$50 - \$99.99  
 \$100 - \$199.99  
 \$200 - \$299.99  
 \$300 - \$399.99  
 \$400 - \$599.99  
 \$600 - \$799.99  
 \$800 - \$999.99  
 \$1,000 - \$1,499.99  
 \$1,500 and up

**Brand**

Black & Decker  
 Bosch  
 Bostitch  
 DEWALT  
 Hitachi  
 Makita  
 Milwaukee  
 Porter-Cable

**Cordless**

Cordless

Image	SKU	Name	Price	Rating
	T1LN2691-22	Milwaukee 2691-22 18V Cordless Compact Li-Ion 2-Tool Combo Kit with Case	329.99 <del>\$329.99</del> \$199.00	★★★★★
	DEWALDCK265L	DEWALT DCK265L 18V Cordless Compact Li-Ion 2-Tool Combo Kit	349.99 <del>\$349.99</del> \$279.00	★★★★★
	B0KNGC18B-59	Black & Decker GC18B-59 18V 59-Piece Home Project Kit	\$64.99	
	MKTNLXT218	Makita LXT218 18V Cordless LXT Lithium-Ion 2-Piece Combo Kit	6429.99 <del>\$6429.99</del> \$379.99	★★★★★
	BSTNROOFKIT2	Bostitch ROOFKIT2 1-3/4-In Roofing Nailer and 18 Gauge Cap Stapler Combo Kit	5349.99 <del>\$5349.99</del> \$319.99	
	MKTNLXT405	Makita LXT405 18V Cordless LXT Lithium-Ion 4-Piece Combo Kit	5599.99 <del>\$5599.99</del> \$469.99	★★★★★
	MKTNLXT601	Makita LXT601 18V Cordless LXT Lithium-Ion 6-Piece Combo Kit	5759.99 <del>\$5759.99</del> \$699.99	★★★★★
	MKTNLXT407	Makita LXT407 18V Cordless LXT Lithium-Ion 4-Piece Combo Kit	5599.99 <del>\$5599.99</del> \$469.99	★★★★★

Image

SKU

Name

Price

Rating



DEWALDCK265L  
DEWALT DCK265L 18V  
Cordless Compact Li-Ion 2-  
Tool Combo Kit

349.99 ~~\$349.99~~ \$279.00  
★★★★★

# motivation:

A screenshot of a Delicious bookmark page. The main title is "Scrapy | An open source web scraping framework for Python". Below the title, there's a "History" section showing a list of saved items with their titles, dates, and tags. On the right side, there's a "Tags" section with a list of tags and their counts, such as "python" (400), "web" (301), and "framework" (236).

A screenshot of the Maru restaurant website. The header includes the name "maru" and the address "34280 Town Center Drive, Valencia, CA 91355". The page is divided into sections for "About Maru", "Location", "Hours", and "Contact". There is also a map showing the restaurant's location and a "Facebook" link at the bottom right.

A screenshot of a YouTube video player. The video is titled "Richard Feynman - Quantum Mechanics". The video player shows a man (Richard Feynman) speaking. Below the video, there are "Suggestions" for other videos related to Feynman and quantum mechanics. The video has 34,688 views and is from the channel "PBS".

A screenshot of a product page for Dewalt Combo Kits. The page features a grid of product images and descriptions. The products include various power tool kits such as "Miter Saw, Drill, and 18V Cordless Compact Lithium-Ion 2-Tool Combo Kit with Case". The page includes filters for "Rating", "Price", and "Brand", and a "Refine By" section.

# DESIRED PROPERTIES

**DESIRED PROPERTIES**

**SPEED**

# CONSTRAINTS



# CONSTRAINTS

- Politeness



# CONSTRAINTS

- Politeness
- Distributed

# CONSTRAINTS

- Politeness
- Distributed
  - Linear Scalability

# CONSTRAINTS

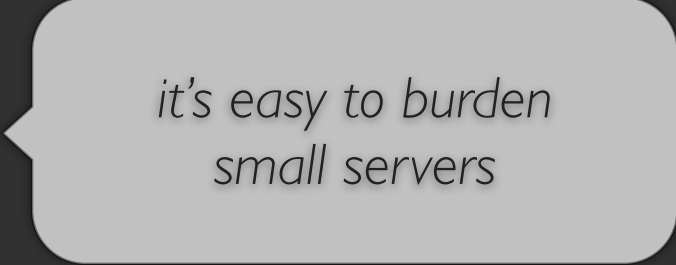
- Politeness
- Distributed
  - Linear Scalability
  - Even partitioning

# CONSTRAINTS

- Politeness
- Distributed
  - Linear Scalability
  - Even partitioning
  - Minimum overlap

# CONSTRAINTS

- Politeness
- Distributed
  - Linear Scalability
  - Even partitioning
  - Minimum overlap



*it's easy to burden  
small servers*

# CONSTRAINTS

- Politeness
- Distributed
  - Linear Scalability
  - Even partitioning
  - Minimum overlap

*(for any significant  
crawl)*

# CONSTRAINTS

- Politeness
- Distributed
  - Linear Scalability
  - Even partitioning
  - Minimum overlap

*n machines =  
n\*m pages-per-second*

# CONSTRAINTS

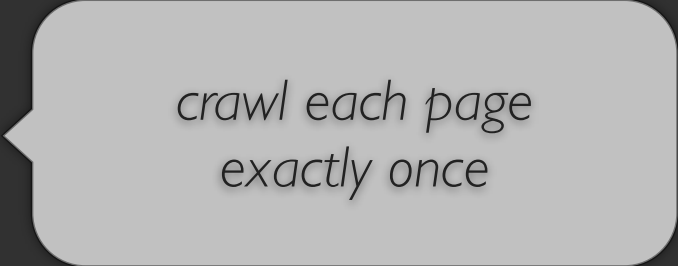
- Politeness
- Distributed
  - Linear Scalability
  - Even partitioning
  - Minimum overlap

*every machine should  
perform equal work*



# CONSTRAINTS

- Politeness
- Distributed
  - Linear Scalability
  - Even partitioning
  - Minimum overlap



*crawl each page  
exactly once*

# CONSTRAINTS

- Politeness
- Distributed
  - Linear Scalability
  - Even partitioning
  - Minimum overlap

# BASIC ALGORITHM

Initialize:

```
UrlsDone = null
```

```
UrlFrontier = { 'google.com/index.html', .. }
```

Repeat

```
url = UrlFrontier.getNext()
```

```
ip = DNSlookup(url.getHostname())
```

```
html = DownloadPage(ip, url.getPath())
```

```
UrlsDone.insert(url)
```

```
newUrls = parseForLinks(html)
```

```
For each newUrl
```

```
    If not UrlsDone.contains(newUrl)
```

```
        then UrlsTodo.insert(newUrl)
```

## Initialize:

```
UrlsDone = null
```

```
UrlFrontier = { 'google.com/index.html', .. }
```

## Repeat

```
url = UrlFrontier.getNext()
```

```
ip = DNSlookup(url.getHostname())
```

```
html = DownloadPage(ip, url.getPath())
```

```
UrlsDone.insert(url)
```

```
newUrls = parseForLinks(html)
```

```
For each newUrl
```

```
    If not UrlsDone.contains(newUrl)
```

```
    then UrlsTodo.insert(newUrl)
```

Initialize:

```
UrlsDone = null
```

```
UrlFrontier = { 'google.com/index.html', .. }
```

Repeat

```
url = UrlFrontier.getNext()
```

```
ip = DNSlookup(url.getHostname())
```

```
html = DownloadPage(ip, url.getPath())
```

```
UrlsDone.insert(url)
```

```
newUrls = parseForLinks(html)
```

```
For each newUrl
```

```
    If not UrlsDone.contains(newUrl)
```

```
    then UrlsTodo.insert(newUrl)
```

Initialize:

```
UrlsDone = null
```

```
UrlFrontier = { 'google.com/index.html', .. }
```

Repeat

```
url = UrlFrontier.getNext()
```

```
ip = DNSlookup(url.getHostname())
```

```
html = DownloadPage(ip, url.getPath())
```

```
UrlsDone.insert(url)
```

```
newUrls = parseForLinks(html)
```

```
For each newUrl
```

```
    If not UrlsDone.contains(newUrl)
```

```
    then UrlsTodo.insert(newUrl)
```

Initialize:

```
UrlsDone = null
```

```
UrlFrontier = { 'google.com/index.html', .. }
```

Repeat

```
url = UrlFrontier.getNext()
```

```
ip = DNSlookup(url.getHostname())
```

```
html = DownloadPage(ip, url.getPath())
```

```
UrlsDone.insert(url)
```

```
newUrls = parseForLinks(html)
```

```
For each newUrl
```

```
    If not UrlsDone.contains(newUrl)
```

```
    then UrlsTodo.insert(newUrl)
```



Initialize:

```
UrlsDone = null
```

```
UrlFrontier = { 'google.com/index.html', .. }
```

Repeat

```
url = UrlFrontier.getNext()
```

```
ip = DNSlookup(url.getHostname())
```

```
html = DownloadPage(ip, url.getPath())
```

```
UrlsDone.insert(url)
```

```
newUrls = parseForLinks(html)
```

```
For each newUrl
```

```
    If not UrlsDone.contains(newUrl)
```

```
    then UrlsTodo.insert(newUrl)
```

Initialize:

```
UrlsDone = null
```

```
UrlFrontier = { 'google.com/index.html', .. }
```

Repeat

```
url = UrlFrontier.getNext()
```

```
ip = DNSlookup(url.getHostname())
```

```
html = DownloadPage(ip, url.getPath())
```

```
UrlsDone.insert(url)
```

```
newUrls = parseForLinks(html)
```

```
For each newUrl
```

```
    If not UrlsDone.contains(newUrl)
```

```
    then UrlsTodo.insert(newUrl)
```

Initialize:

```
UrlsDone = null
```

```
UrlFrontier = { 'google.com/index.html', .. }
```

Repeat

```
url = UrlFrontier.getNext()
```

```
ip = DNSlookup(url.getHostname())
```

```
html = DownloadPage(ip, url.getPath())
```

```
UrlsDone.insert(url)
```

```
newUrls = parseForLinks(html)
```

```
For each newUrl
```

```
    If not UrlsDone.contains(newUrl)
```

```
    then UrlsTodo.insert(newUrl)
```

Initialize:

```
UrlsDone = null
```

```
UrlFrontier = { 'google.com/index.html', .. }
```

Repeat

```
url = UrlFrontier.getNext()
```

```
ip = DNSlookup(url.getHostname())
```

```
html = DownloadPage(ip, url.getPath())
```

```
UrlsDone.insert(url)
```

```
newUrls = parseForLinks(html)
```

```
For each newUrl
```

```
    If not UrlsDone.contains(newUrl)
```

```
    then UrlsTodo.insert(newUrl)
```

Initialize:

```
UrlsDone = null
```

```
UrlFrontier = { 'google.com/index.html', .. }
```

Repeat

```
url = UrlFrontier.getNext()
```

```
ip = DNSlookup(url.getHostname())
```

```
html = DownloadPage(ip, url.getPath())
```

```
UrlsDone.insert(url)
```

```
newUrls = parseForLinks(html)
```

```
For each newUrl
```

```
    If not UrlsDone.contains(newUrl)
```

```
    then UrlsTodo.insert(newUrl)
```

Initialize:

```
UrlsDone = null
```

```
UrlFrontier = { 'google.com/index.html', .. }
```

Repeat

```
url = UrlFrontier.getNext()
```

```
ip = DNSlookup(url.getHostname())
```

```
html = DownloadPage(ip, url.getPath())
```

```
UrlsDone.insert(url)
```

```
newUrls = parseForLinks(html)
```

For each newUrl

```
  If not UrlsDone.contains(newUrl)
```

```
  then UrlsTodo.insert(newUrl)
```

Initialize:

```
UrlsDone = null
```

```
UrlFrontier = { 'google.com/index.html', .. }
```

Repeat

```
url = UrlFrontier.getNext()
```

```
ip = DNSlookup(url.getHostname())
```

```
html = DownloadPage(ip, url.getPath())
```

```
UrlsDone.insert(url)
```

```
newUrls = parseForLinks(html)
```

```
For each newUrl
```

```
    If not UrlsDone.contains(newUrl)
```

```
    then UrlsTodo.insert(newUrl)
```

Initialize:

```
UrlsDone = null
```

```
UrlFrontier = { 'google.com/index.html', .. }
```

Repeat

```
url = UrlFrontier.getNext()
```

```
ip = DNSlookup(url.getHostname())
```

```
html = DownloadPage(ip, url.getPath())
```

```
UrlsDone.insert(url)
```

```
newUrls = parseForLinks(html)
```

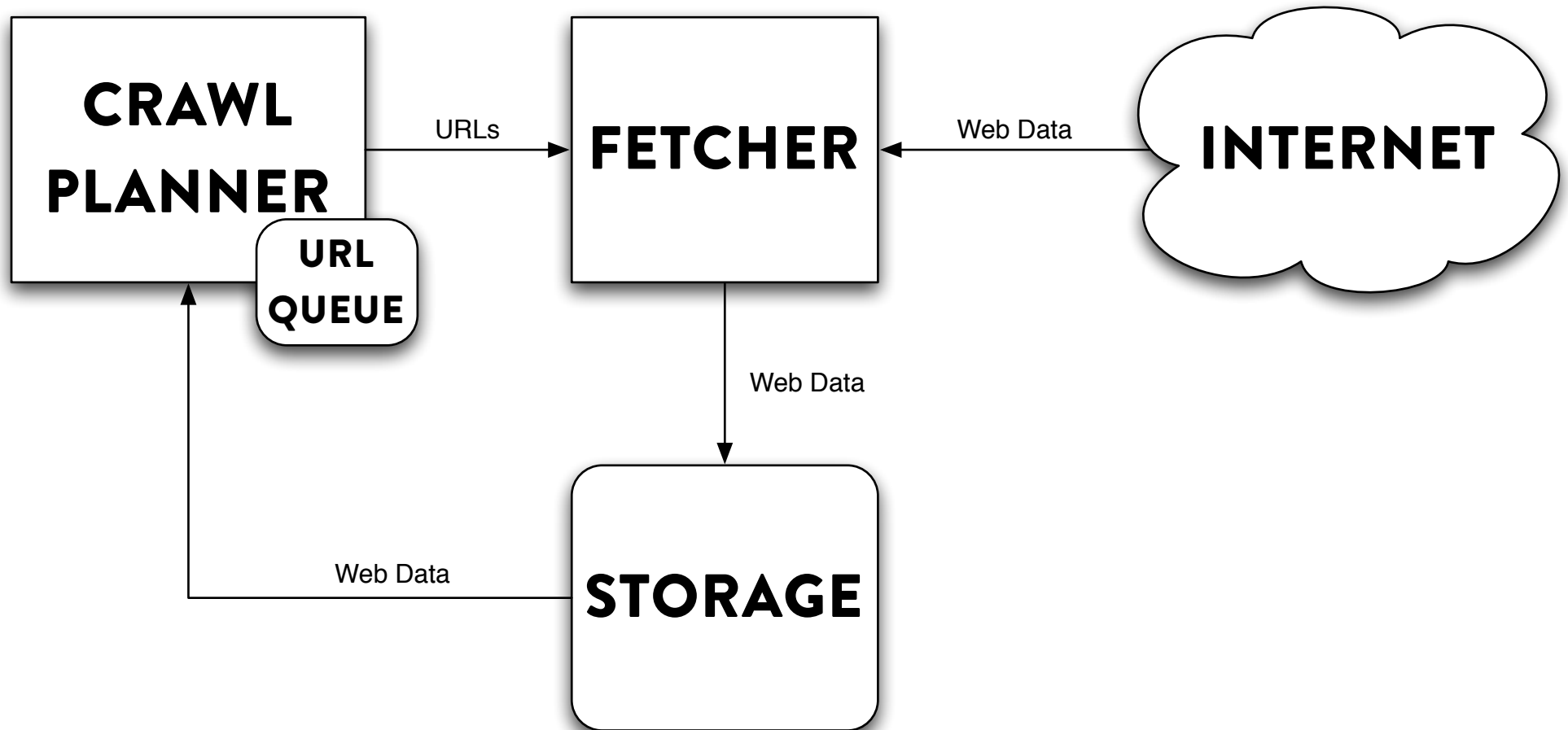
```
For each newUrl
```

```
    If not UrlsDone.contains(newUrl)
```

```
        then UrlsTodo.insert(newUrl)
```



# architecture overview



# CHALLENGES

*challenges:*

**depends on your ambitions**

*challenges:*

## **Google's Index Size:**

**1998 - 26 million**

**2005 - 8 billion**

**2008 - 1 trillion**

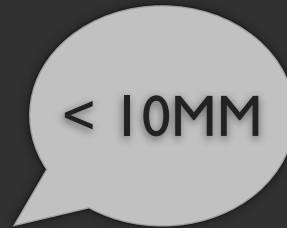
<http://www.nytimes.com/2005/08/15/technology/15search.html>

<http://googleblog.blogspot.com/2008/07/we-knew-web-was-big.html>

*challenges:*

**small crawls are easy**

*challenges:*



**small crawls are easy**

*challenges:*

large crawls are interesting

*challenges:*



*challenges:*

# DNS Lookup

*challenges:*

**DNS Lookup**  
**URLs Crawled**

*challenges:*

**DNS Lookup**  
**URLs Crawled**  
**Politeness**

*challenges:*

**DNS Lookup**  
**URLs Crawled**  
**Politeness**  
**URL Frontier**

*challenges:*

**DNS Lookup**  
**URLs Crawled**  
**Politeness**  
**URL Frontier**  
**Queueing URLs**

*challenges:*

**DNS Lookup**

**URLs Crawled**

**Politeness**

**URL Frontier**

**Queueing URLs**

**Extracting URLs**

*challenges:*

# DNS LOOKUP

Initialize:

```
UrlsDone = null
```

```
UrlFrontier = { 'google.com/index.html', .. }
```

Repeat

```
url = UrlFrontier.getNext()
```

```
ip = DNSlookup(url.getHostname())
```

```
html = DownloadPage(ip, url.getPath())
```

```
UrlsDone.insert(url)
```

```
newUrls = parseForLinks(html)
```

```
For each newUrl
```

```
    If not UrlsDone.contains(newUrl)
```

```
    then UrlsTodo.insert(newUrl)
```



*challenges:*

# ▶ DNS LOOKUP

can easily be a bottleneck

*challenges:*

# ▶ DNS LOOKUP

- consider running your own DNS servers
  - djbdns
  - PowerDNS
  - etc.

*challenges:*

# ▶ DNS LOOKUP

- **be aware of software limitations**
  - **gethostbyaddr is synchronized**
  - **same with many “default” DNS clients**

*challenges:*

# ▶ DNS LOOKUP

*You'll know when you need it*

*challenges:*

**URLs CRAWLED**

Initialize:

```
UrlsDone = null
```

```
UrlFrontier = { 'google.com/index.html', .. }
```

Repeat

```
url = UrlFrontier.getNext()
```

```
ip = DNSlookup(url.getHostname())
```

```
html = DownloadPage(ip, url.getPath())
```

```
UrlsDone.insert(url)
```

```
newUrls = parseForLinks(html)
```

```
For each newUrl
```

```
    If not UrlsDone.contains(newUrl)
```

```
    then UrlsTodo.insert(newUrl)
```

*challenges:*

# ▶ **URLs CRAWLED**

1 machine, store in memory

*challenges:*

# ► **URLs CRAWLED**

1 machine, store in memory

**NAPKIN CALCULATION**



challenges:

# ► URLs CRAWLED

1 machine, store in memory

## NAPKIN CALCULATION

~50 bytes per URL

e.g. <http://wiki.apache.org/cassandra/ArticlesAndPresentations>

challenges:

# ► URLs CRAWLED

1 machine, store in memory

## NAPKIN CALCULATION

~50 bytes per URL

e.g. `http://wiki.apache.org/cassandra/ArticlesAndPresentations`

+8 bytes for time-last-crawled

as long e.g. `System.currentTimeMillis() -> 1314392455712`

challenges:

# ► URLs CRAWLED

1 machine, store in memory

## NAPKIN CALCULATION

~50 bytes per URL

e.g. <http://wiki.apache.org/cassandra/ArticlesAndPresentations>

+8 bytes for time-last-crawled

as long e.g. `System.currentTimeMillis()` -> 1314392455712

x 100 million

challenges:

# ► URLs CRAWLED

1 machine, store in memory

## NAPKIN CALCULATION

~50 bytes per URL

e.g. <http://wiki.apache.org/cassandra/ArticlesAndPresentations>

+8 bytes for time-last-crawled

as long e.g. `System.currentTimeMillis()` -> 1314392455712

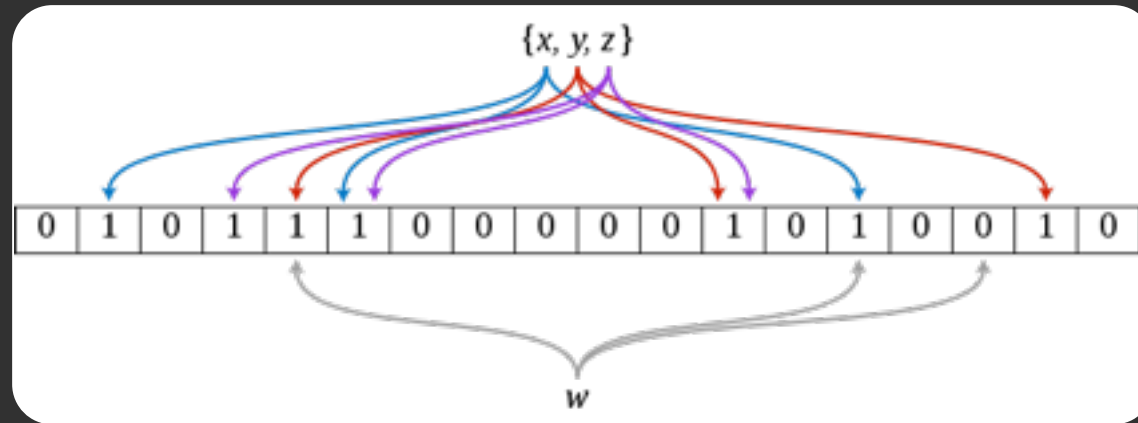
x 100 million

=~ 5.4 gigabytes

*can we do better?*

# BLOOM FILTERS

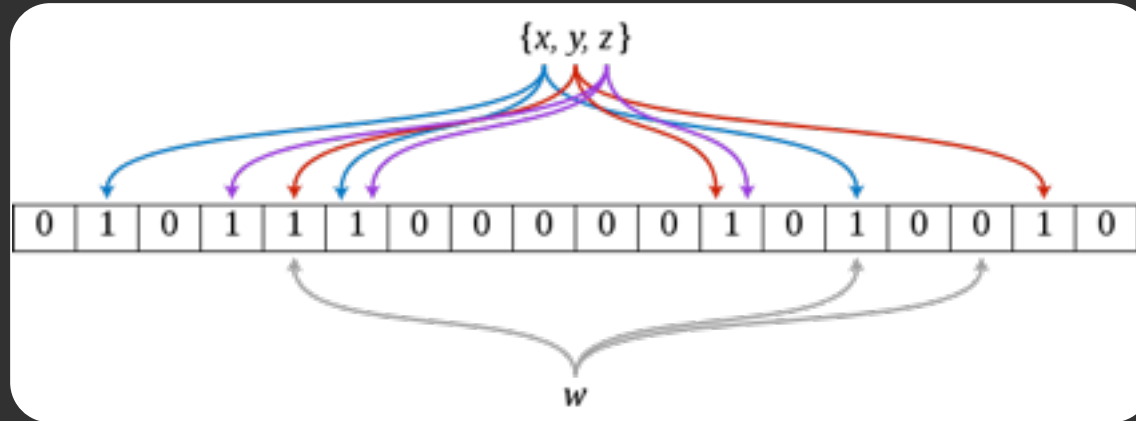
# BLOOM FILTERS



*answers the question:*

**is this item in the set?**

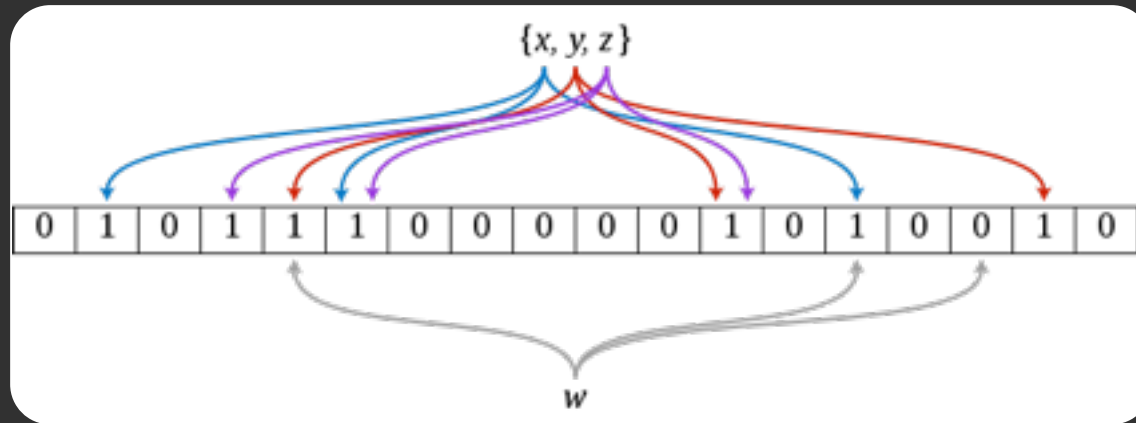
# BLOOM FILTERS



*answers either:*



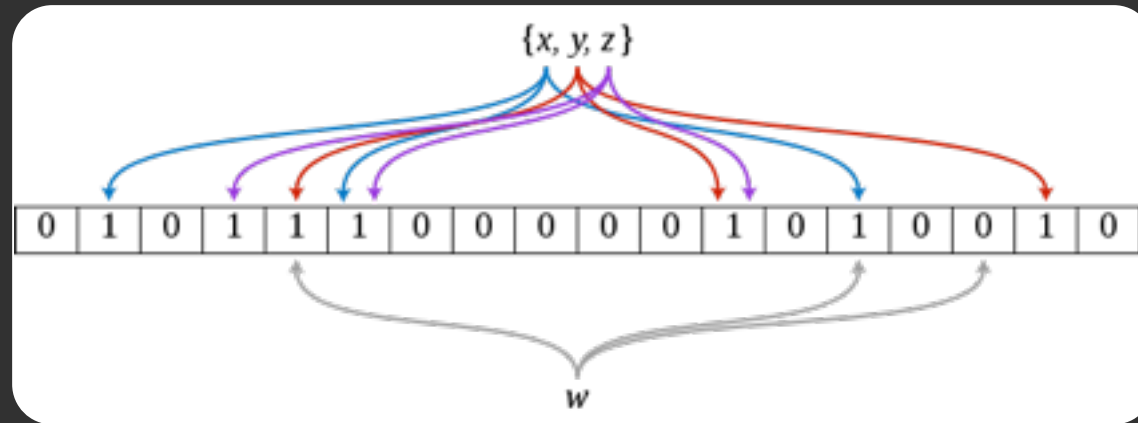
# BLOOM FILTERS



*answers either:*

- **yes, probably**

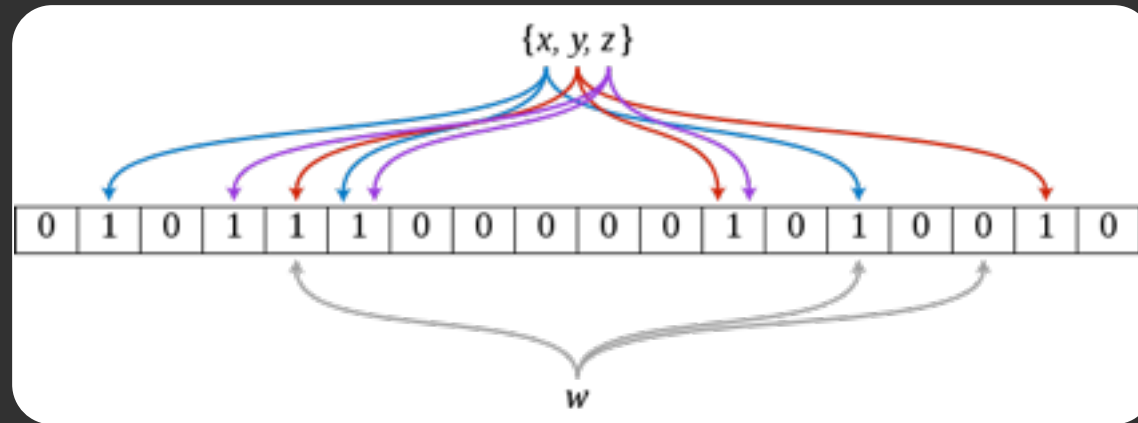
# BLOOM FILTERS



*answers either:*

- yes, probably
- definitely not

# BLOOM FILTERS

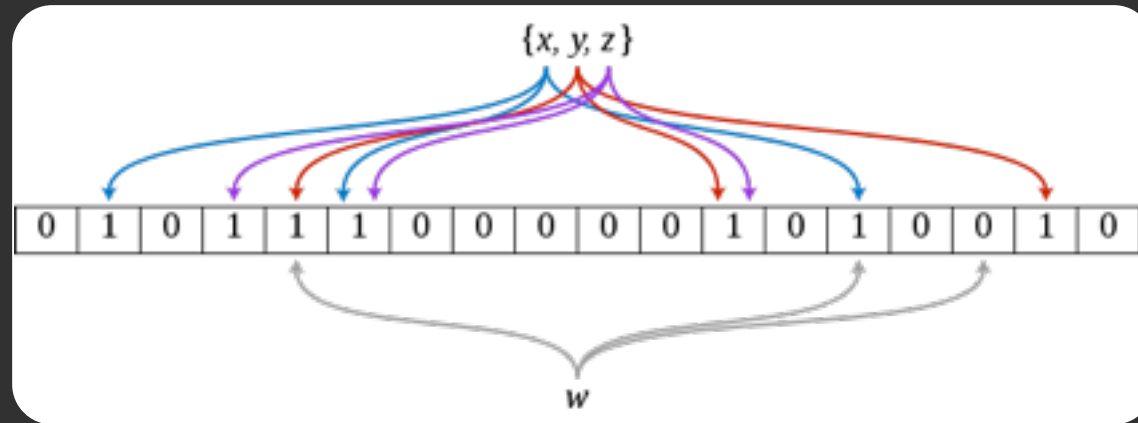


Have we crawled: <http://www.xcombinator.com>?

*answers either:*

- yes, probably
- definitely not

# BLOOM FILTERS



Have we crawled: <http://www.xcombinator.com>?

*answers either:*

- yes, probably
- definitely not



challenges:

# ► URLs CRAWLED

**1 machine, bloom filter**

100 million URLs

1 in 100 million chance  
of false positive

see: <http://hur.st/bloomfilter?n=100000000&p=1.0E-8>

challenges:

# ► URLs CRAWLED

1 machine, bloom filter

## NAPKIN CALCULATION

100 million URLs

1 in 100 million chance

of false positive

see: <http://hur.st/bloomfilter?n=100000000&p=1.0E-8>

challenges:

# ► URLs CRAWLED

1 machine, bloom filter

## NAPKIN CALCULATION

100 million URLs

1 in 100 million chance

of false positive

=~ 457 megabytes

see: <http://hur.st/bloomfilter?n=100000000&p=1.0E-8>

# ▶ BLOOM FILTER



# ▶ BLOOM FILTER

**drawbacks**

# ▶ BLOOM FILTER

## drawbacks

- probabilistic - occasional errors

# ▶ BLOOM FILTER

## drawbacks

- probabilistic - occasional errors
- estimate # of items ahead of time

# ▶ BLOOM FILTER

## drawbacks

- probabilistic - occasional errors
- estimate # of items ahead of time
- can't delete

# ▶ BLOOM FILTER

## drawbacks

- probabilistic - occasional errors
- estimate # of items ahead of time
- can't delete

## solutions

# ▶ BLOOM FILTER

## drawbacks

- probabilistic - occasional errors
  - acceptable
- estimate # of items ahead of time
- can't delete

## solutions

# ▶ BLOOM FILTER

## drawbacks

- probabilistic - occasional errors
  - acceptable
- estimate # of items ahead of time
  - not hard, see Dynamic BFs
- can't delete

## solutions

# ▶ BLOOM FILTER

## drawbacks

- probabilistic - occasional errors
  - acceptable
- estimate # of items ahead of time
- can't delete

## solutions

- not hard, see Dynamic BFs
- pick granularity (days)



# ► BLOOM FILTER

## drawbacks

- probabilistic - occasional errors
  - acceptable
- estimate # of items ahead of time
- can't delete

## solutions

- not hard, see Dynamic BFs
- pick granularity (days)
- cascade them

# BLOOM FILTERS

*references:*

[http://en.wikipedia.org/wiki/Bloom\\_filter](http://en.wikipedia.org/wiki/Bloom_filter)

<http://spyced.blogspot.com/2009/01/all-you-ever-wanted-to-know-about.html>

<http://www.igvita.com/2010/01/06/flow-analysis-time-based-bloom-filters/>

*challenges:*

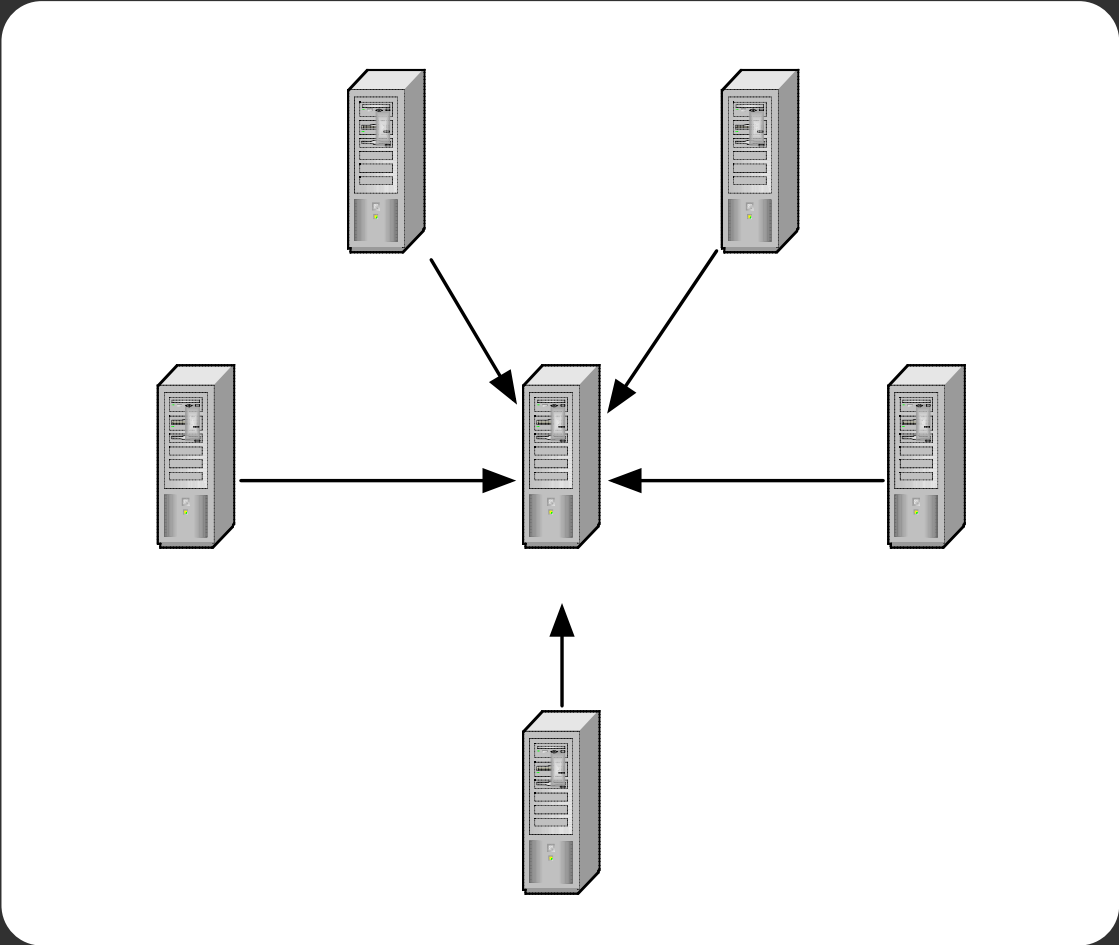
**POLITENESS**

`obey robots . txt`

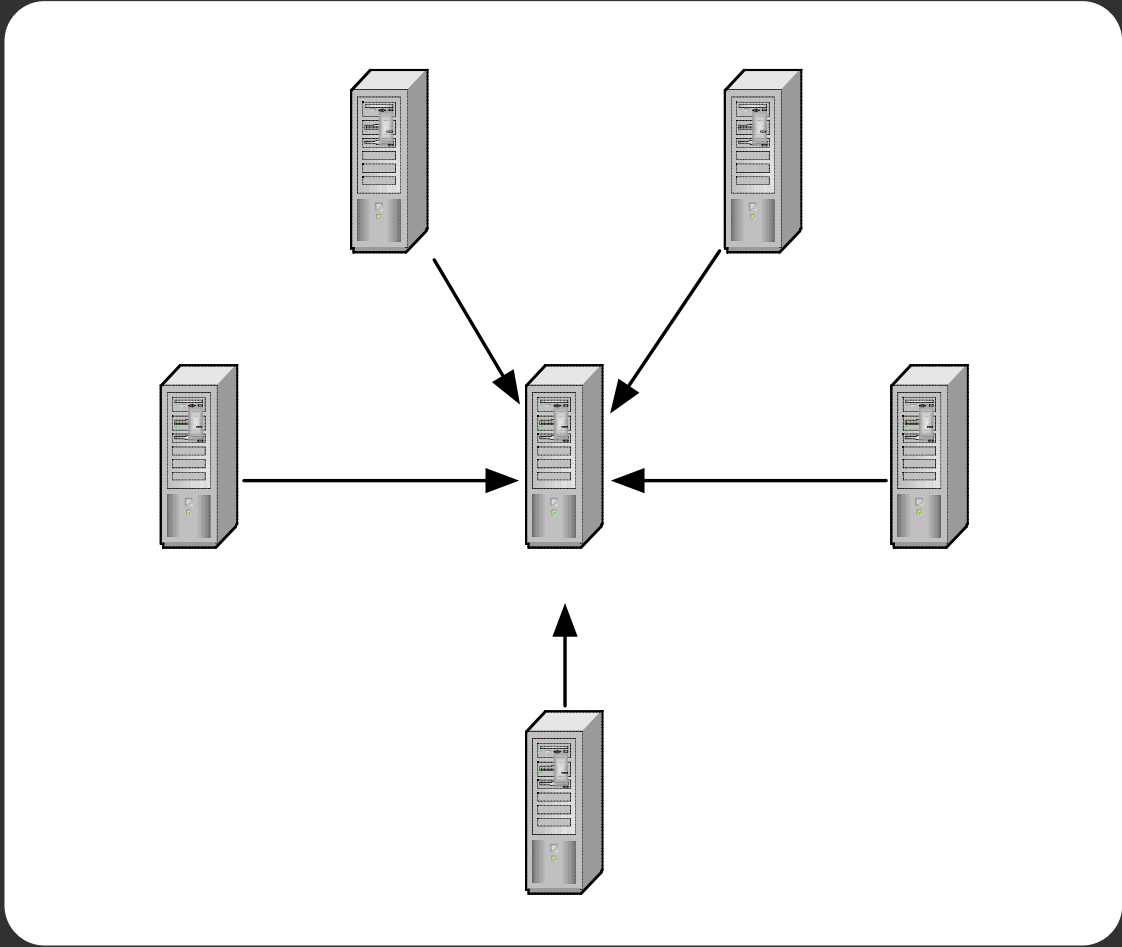
*rule of thumb:*

**wait 2 seconds (w.r.t. ip)**

*centralized politeness*

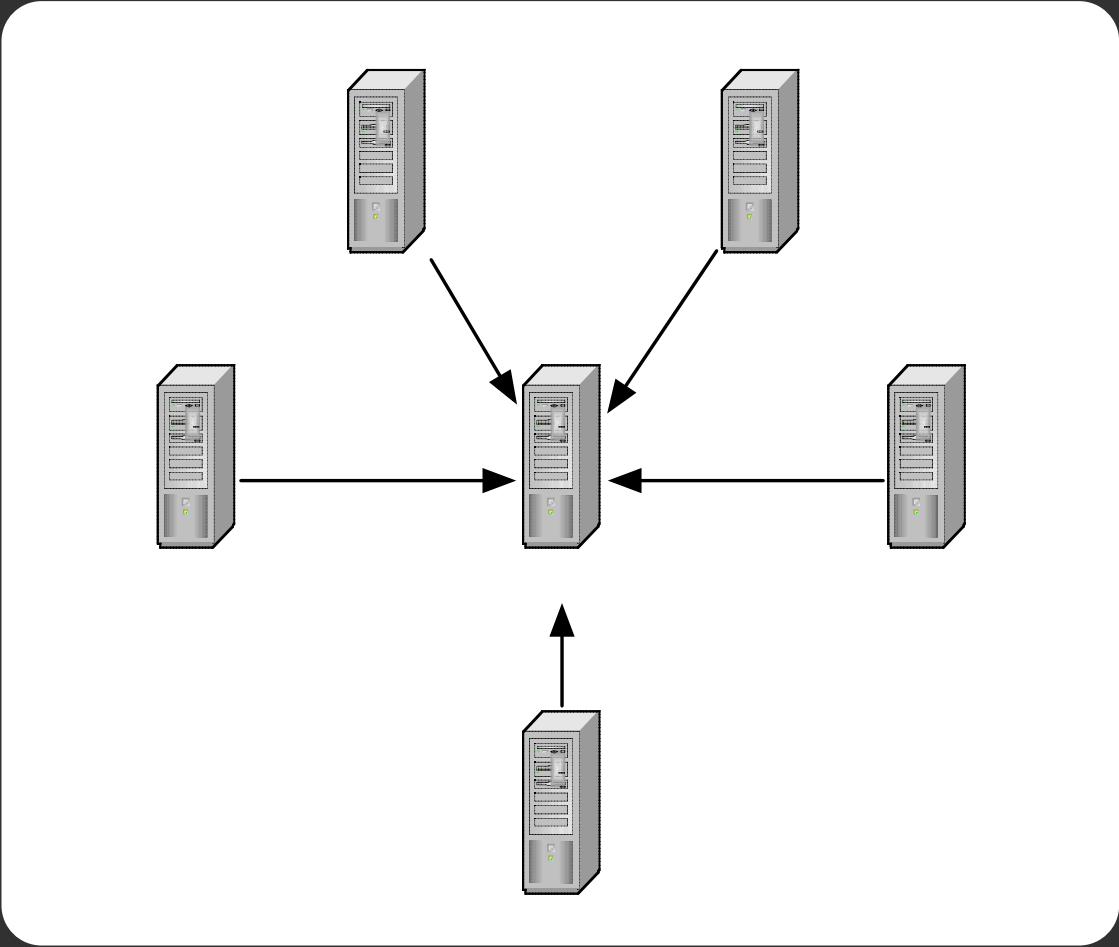


*centralized politeness*



**SPOF**

*centralized politeness*



**SPOF  
contention**



*challenges:*

# ▶ **POLITENESS**

*challenges:*

# ▶ POLITENESS

- Options:

*challenges:*

# ▶ POLITENESS

- Options:
  - central database

*challenges:*

# ▶ POLITENESS

- Options:
  - central database
  - distributed locks (paxos/sigma/zookeeper)

*challenges:*

# ▶ POLITENESS

- Options:
  - central database
  - distributed locks (paxos/sigma/zookeeper)
  - controlled URL distribution

*challenges:*

# ► POLITENESS

- Options:
  - central database
  - distributed locks (paxos/sigma/zookeeper)
  - controlled URL distribution

[http://en.wikipedia.org/wiki/Paxos\\_\(computer\\_science\)](http://en.wikipedia.org/wiki/Paxos_(computer_science))

*challenges:*

# ► POLITENESS

- Options:
  - central database
  - distributed locks (paxos/sigma/zookeeper)
  - controlled URL distribution

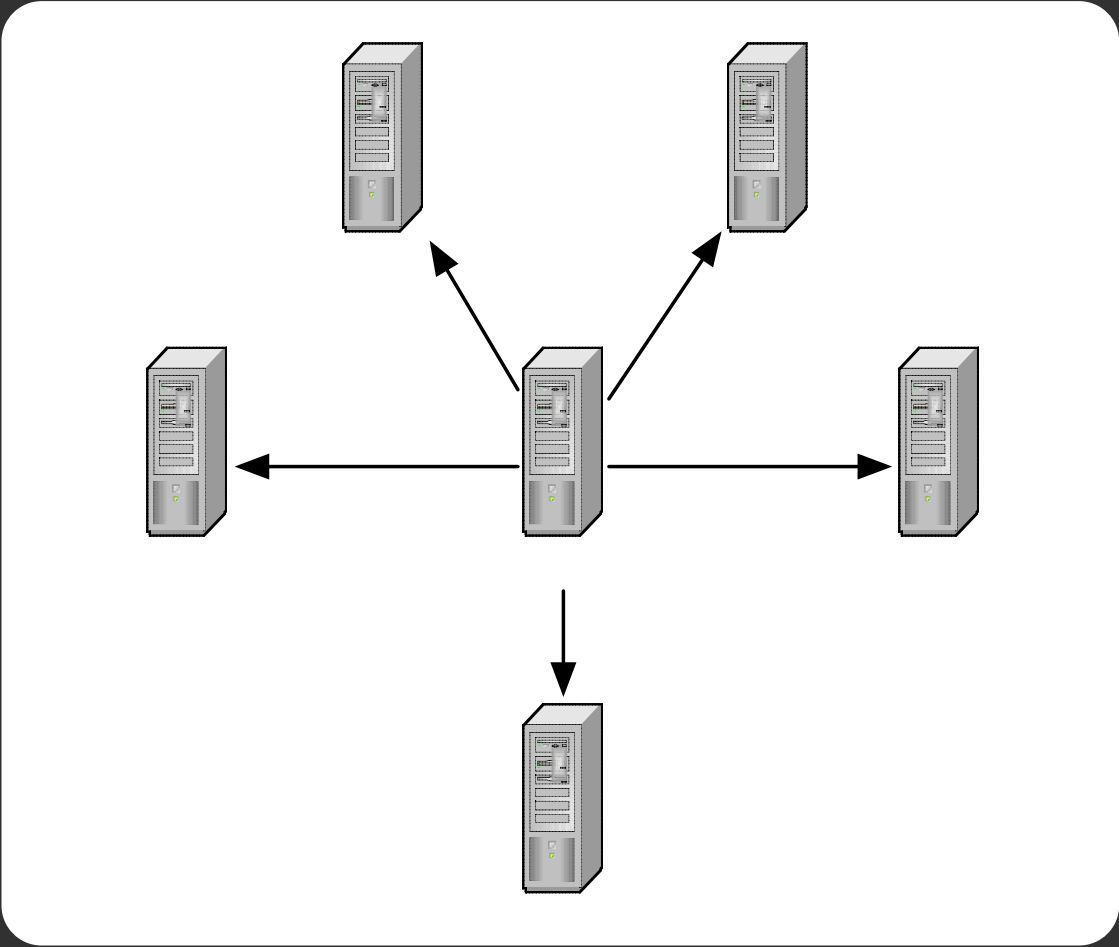
[http://en.wikipedia.org/wiki/Paxos\\_\(computer\\_science\)](http://en.wikipedia.org/wiki/Paxos_(computer_science))

<http://zookeeper.apache.org/>

*challenges:*

# URL FRONTIER





*idea:*

**consistently distribute URLs based on IP**

# modulo

IP	SHA-1	bucket (mod 5)
174.132.225.106	4dd14b0b...	2
74.125.224.115	cf4b7594...	1
157.166.255.19	0ac4d141...	4
69.22.138.129	6c1584fa...	4
98.139.50.166	327252c5...	3

*benefits:*

**same IP *always* goes to same machine**

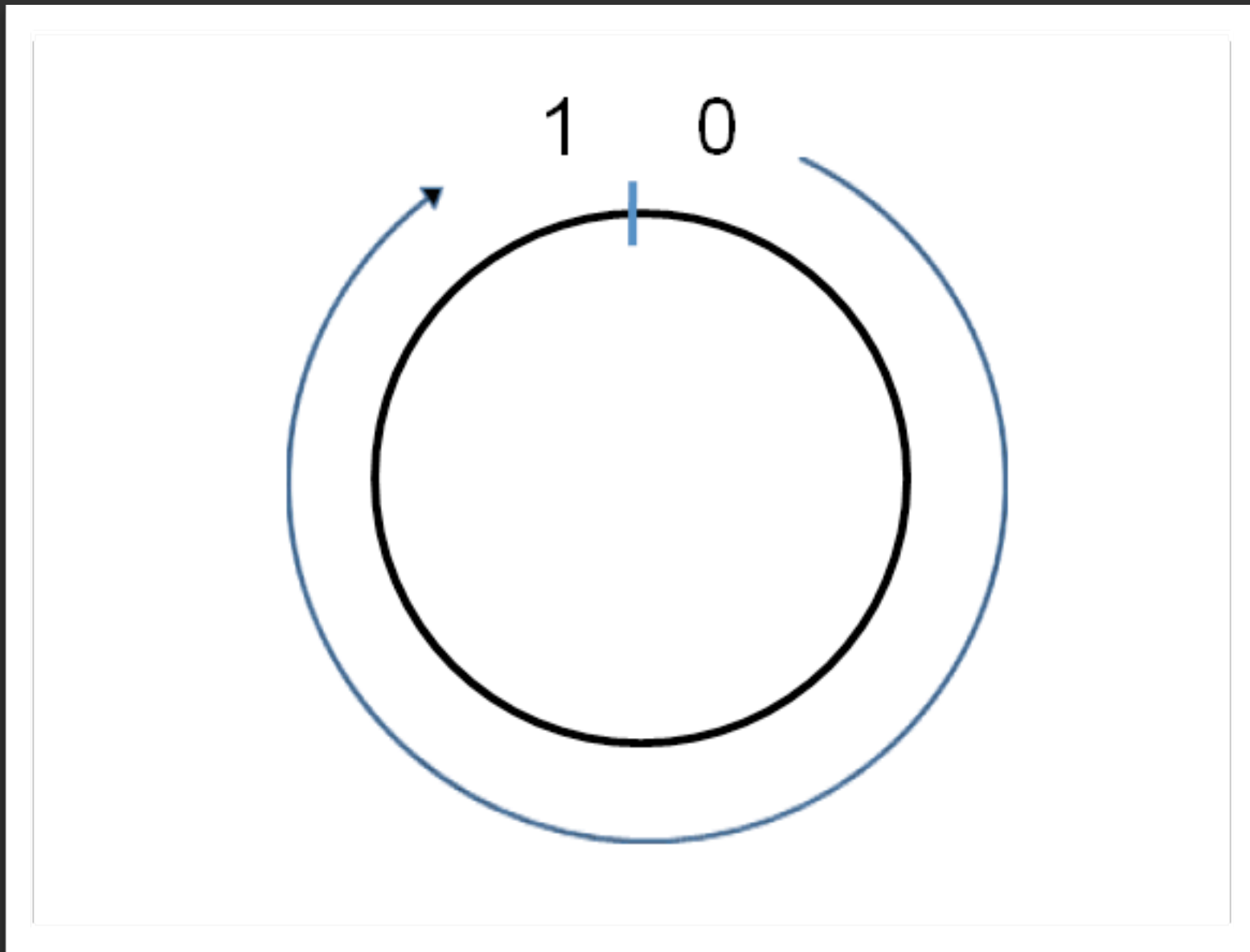
**simple**

*drawbacks:*

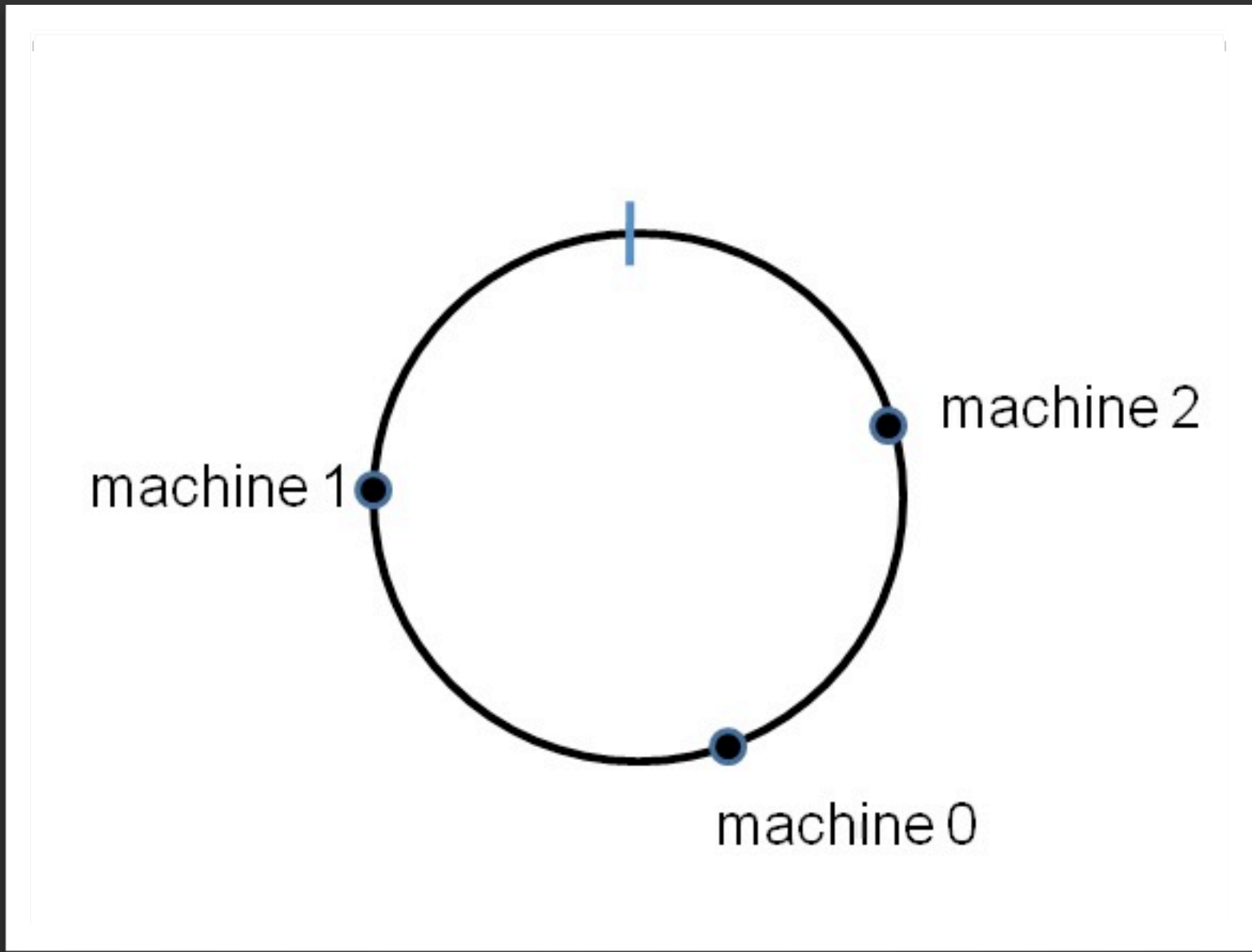
**susceptible to skew**

**can't add / remove nodes without pain**

# consistent hashing

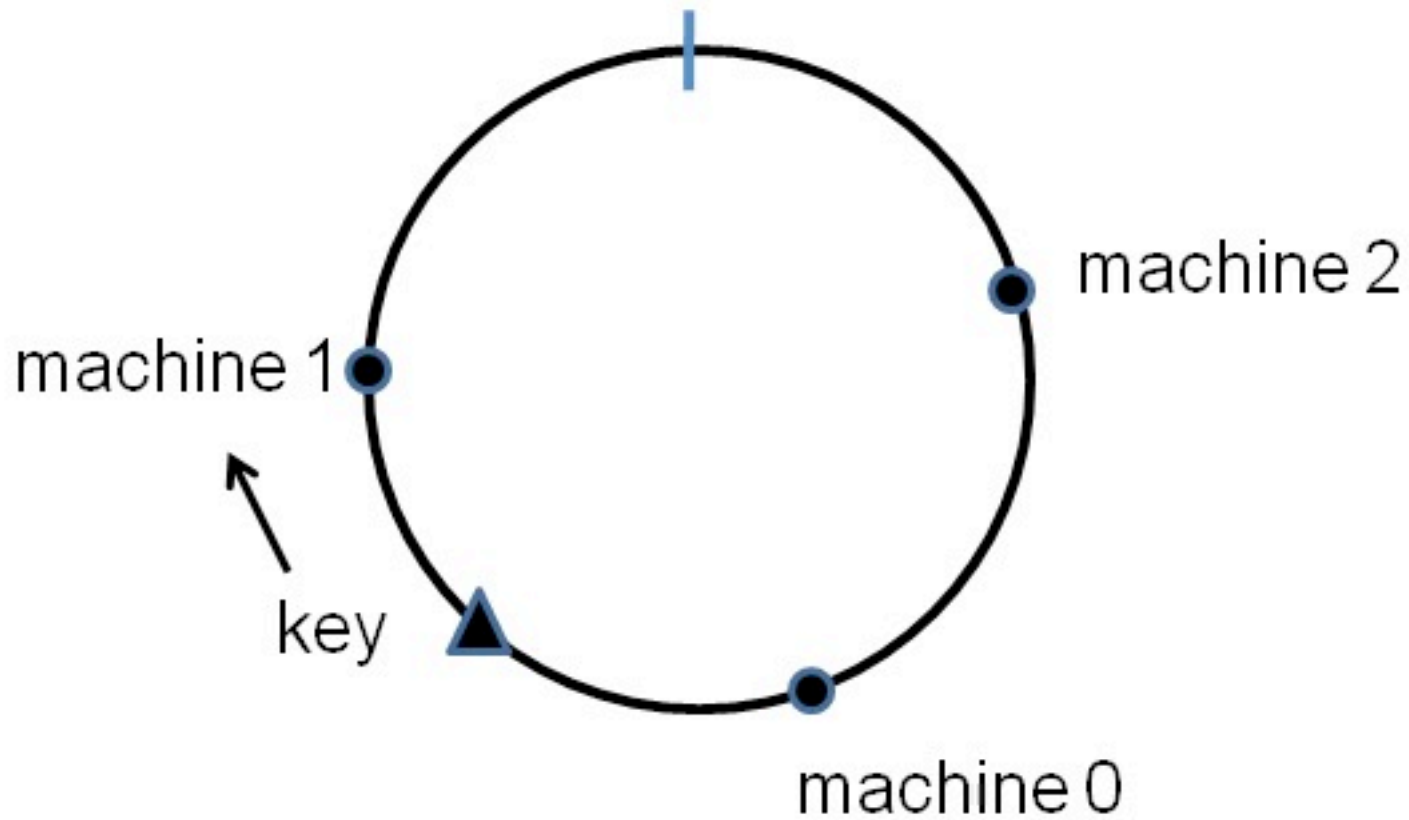


source: <http://michaelnielsen.org/blog/consistent-hashing/>

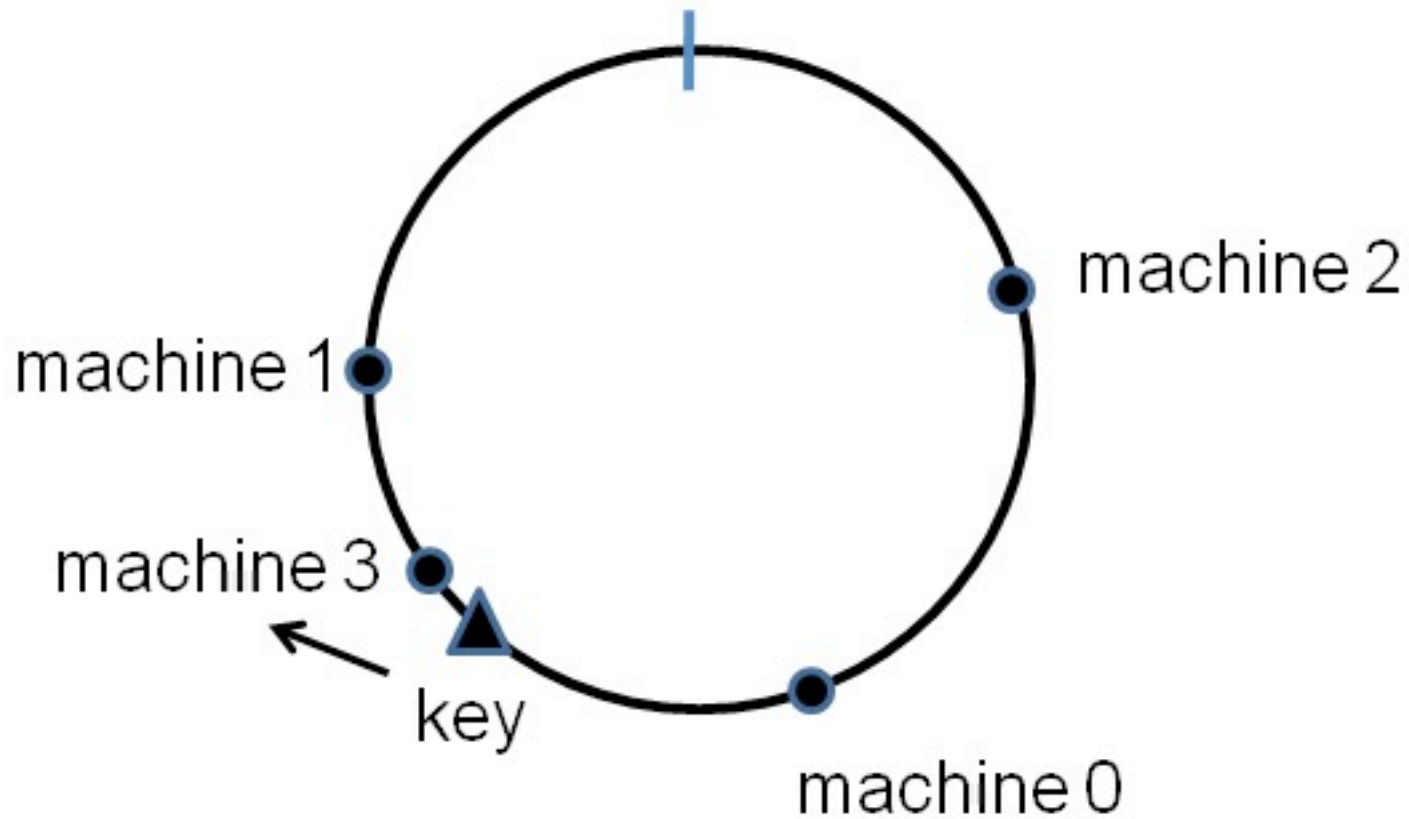


source: <http://michaelnielsen.org/blog/consistent-hashing/>





source: <http://michaelnielsen.org/blog/consistent-hashing/>



source: <http://michaelnielsen.org/blog/consistent-hashing/>

*benefits:*

**~  $1/(n+1)$  URLs move on add/remove**  
**virtual nodes help skew**  
**robust (no SOP)**

*drawbacks:*

**naive solution won't work for large sites**

## *further reading:*

**Chord: A Scalable Peer-to-Peer Lookup Protocol for Internet Applications (2001)** Stoica et al.

**Dynamo: Amazon's Highly Available Key-value Store, SOSP 2007**

**Tapestry: A Resilient Global-Scale Overlay for Service Deployment (2004)** Zhao et al.

*challenges:*

# QUEUEING URLS

***situation:***

*situation:*

**URL**



*situation:*

**URL**

**not recently crawled**

*situation:*

**URL**

**not recently crawled**

**allowed by robots.txt**

*situation:*

**URL**

**not recently crawled**

**allowed by robots.txt**

**polite**

**how to you order them?**

**(within a single machine)**

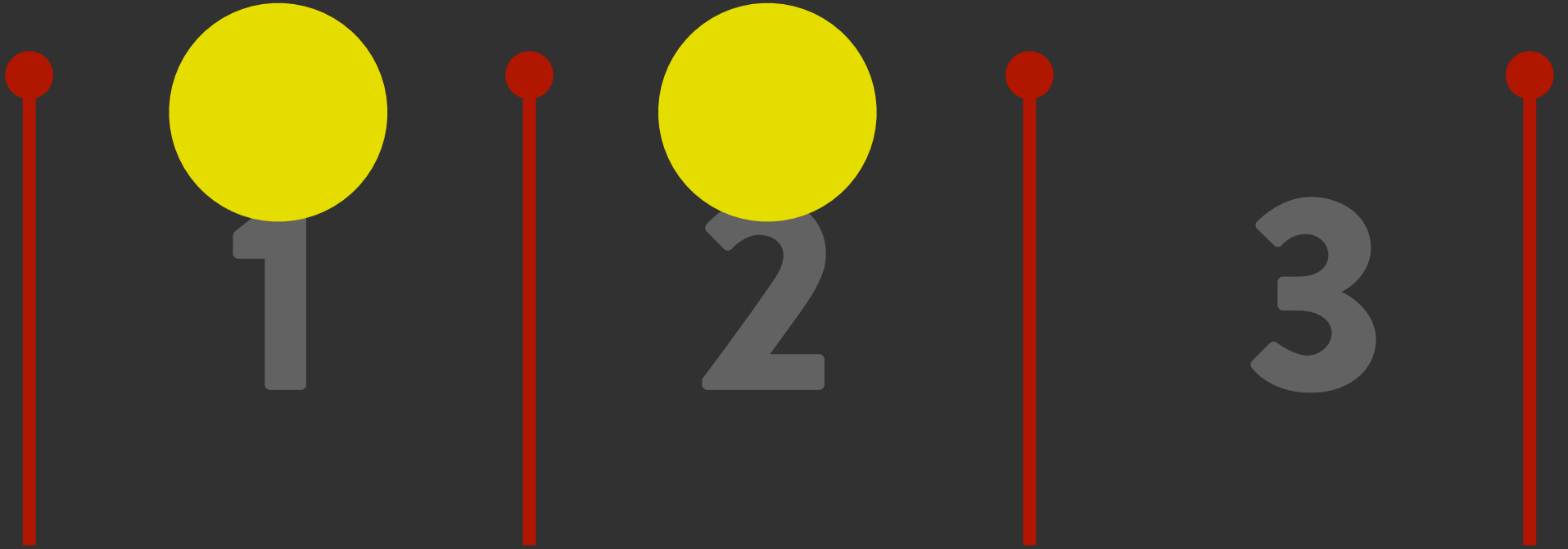
*hash each lane:*



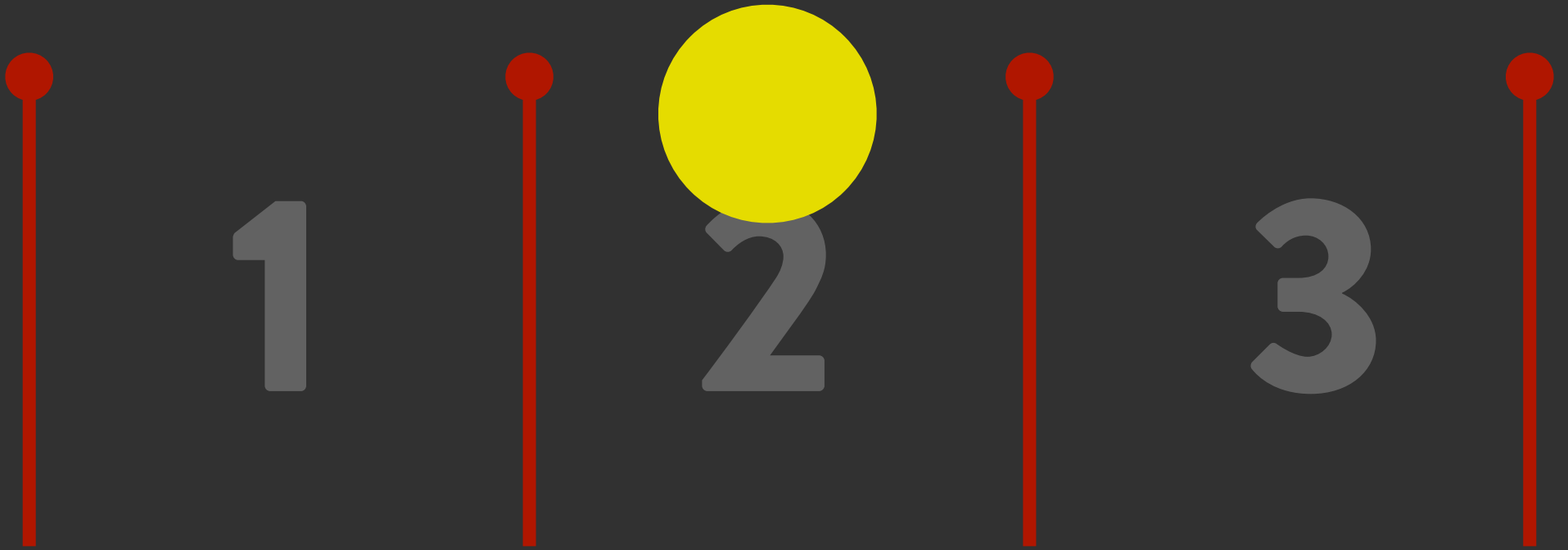
```
http://yachtmaintenanceco.com/  
http://www.amsterdamports.nl/  
http://www.4s-dawn.com/  
http://www.embassysuiteslittlerock.com/  
http://members.tripod.com/airfields\_freeman/NM/Airfields\_NM\_NW.htm  
http://mdgroover.iweb.bsu.edu  
http://music.imbc.com/  
http://www.robertjbradshaw.com  
http://www.kerkattenhoven.be  
http://www.escolania.org/
```

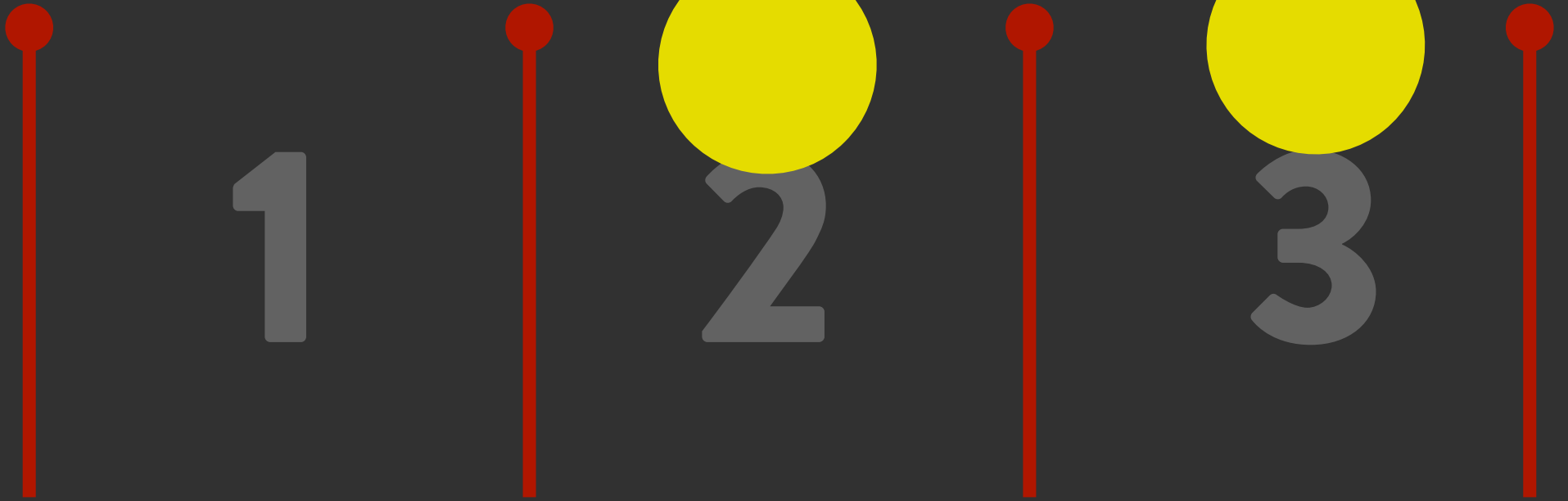














1

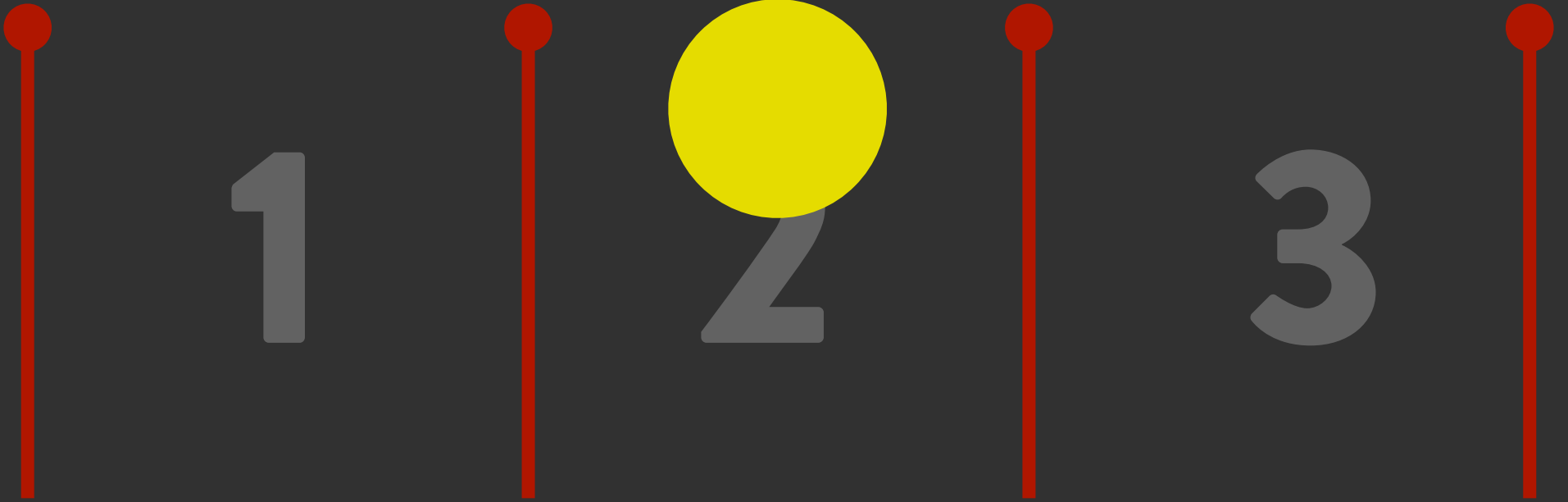
2

3



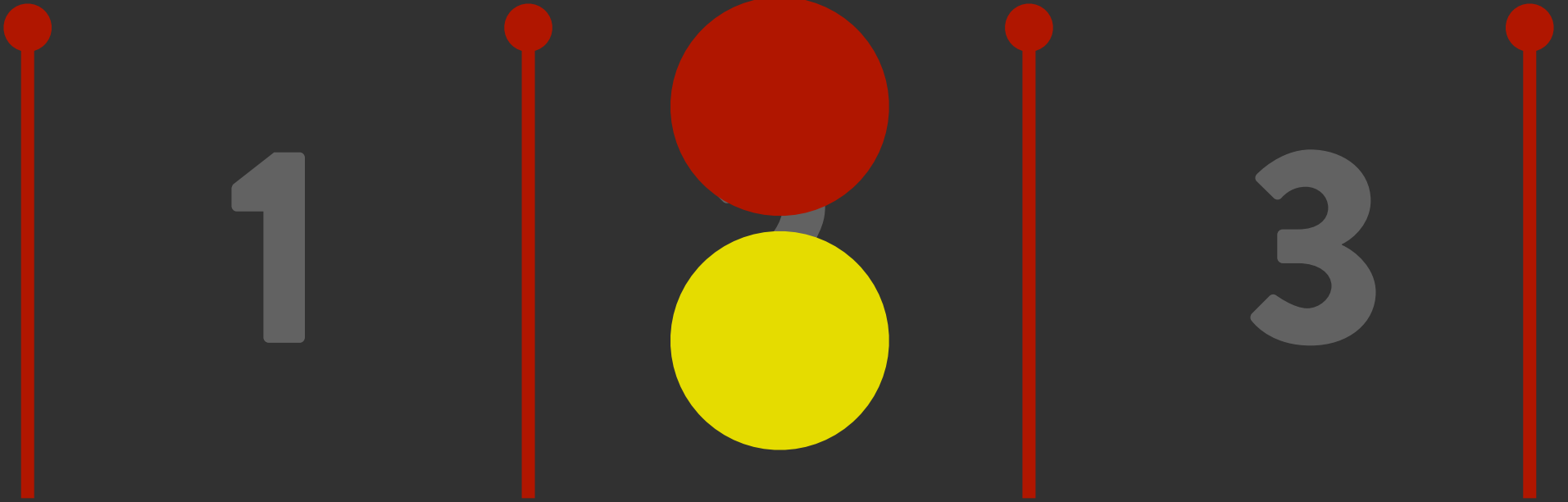


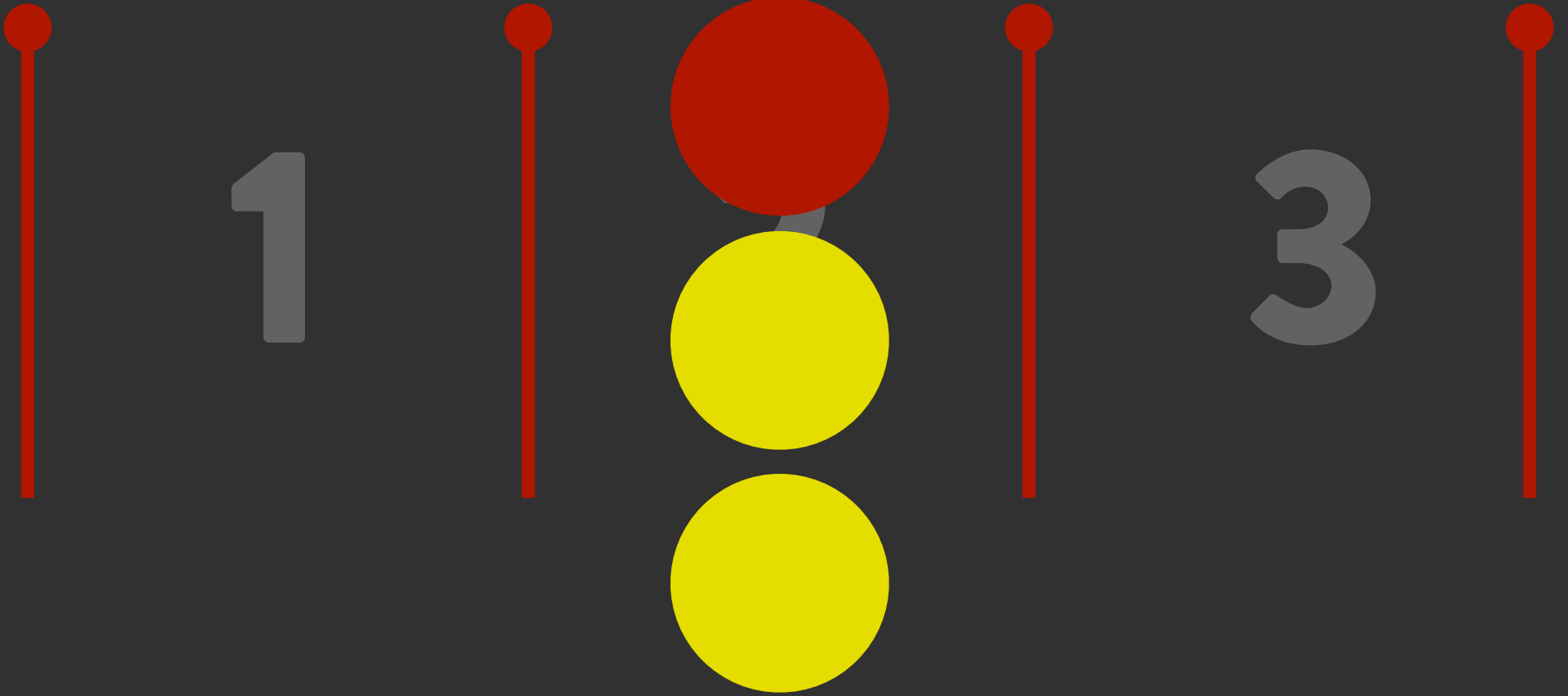


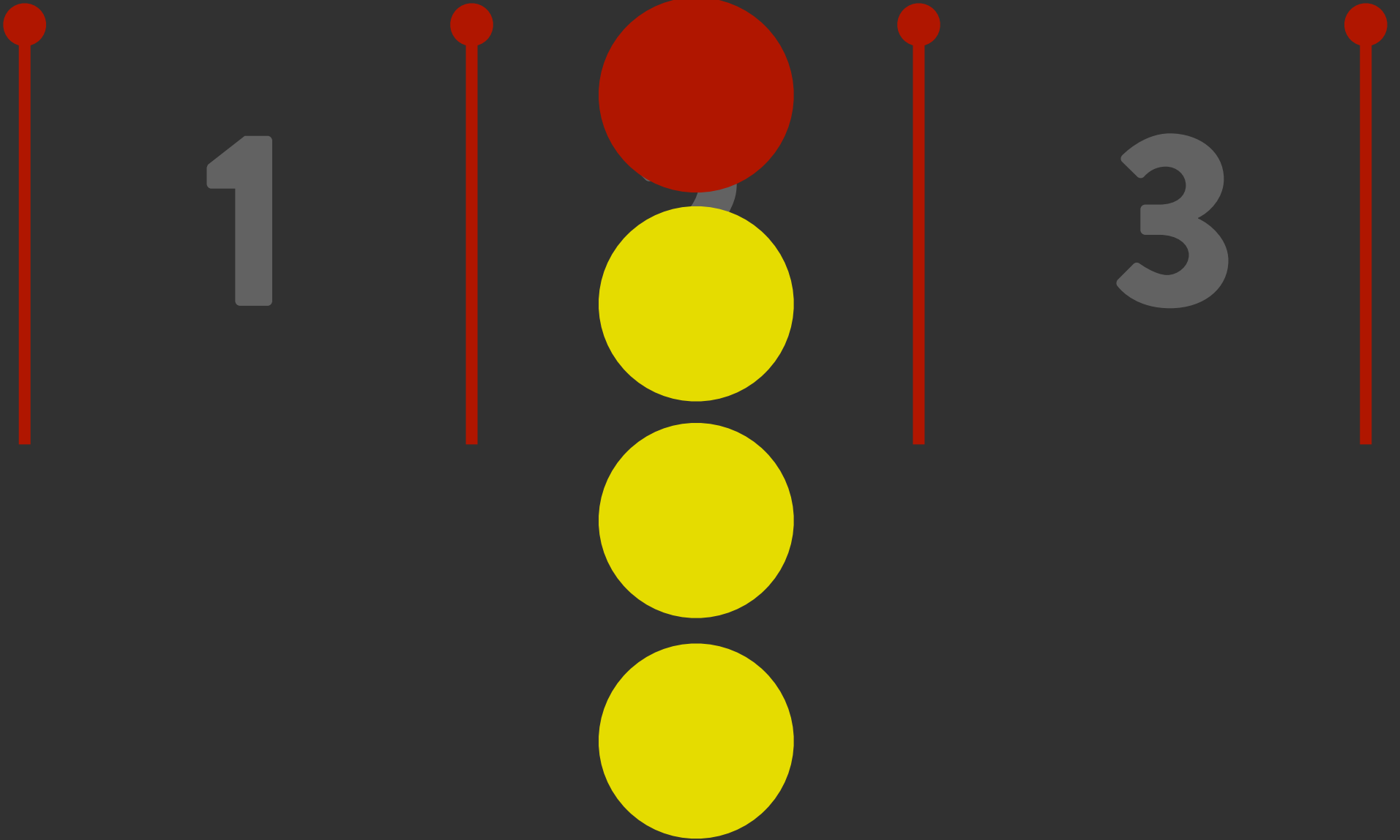


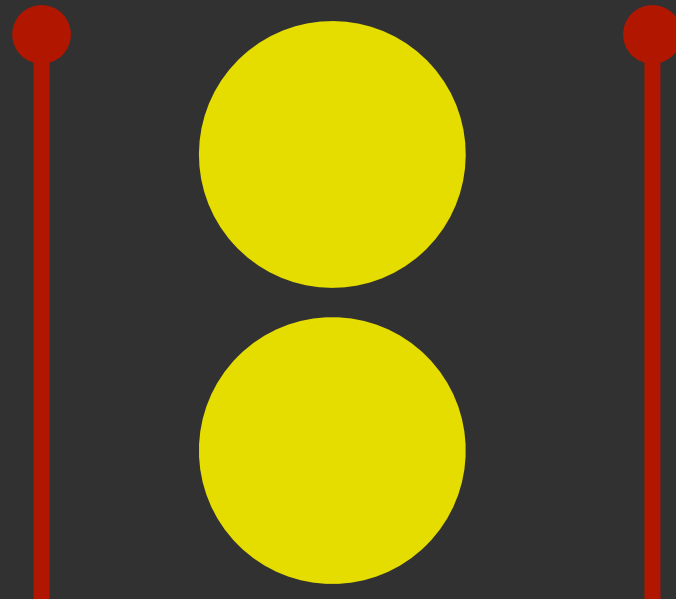


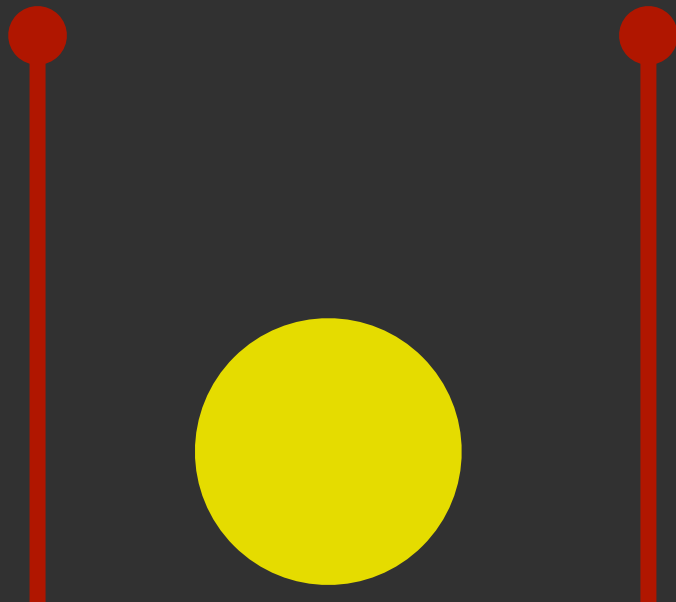


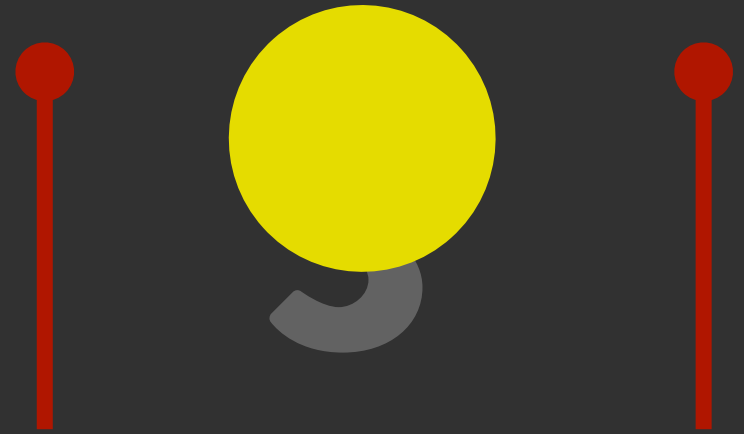
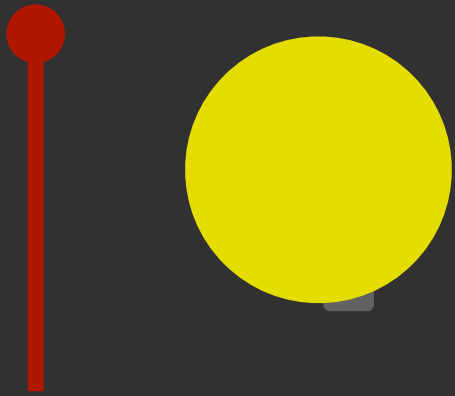


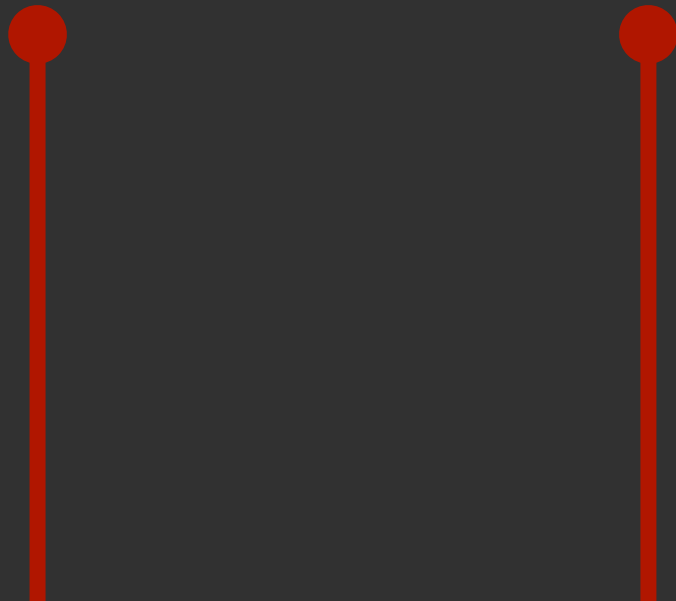


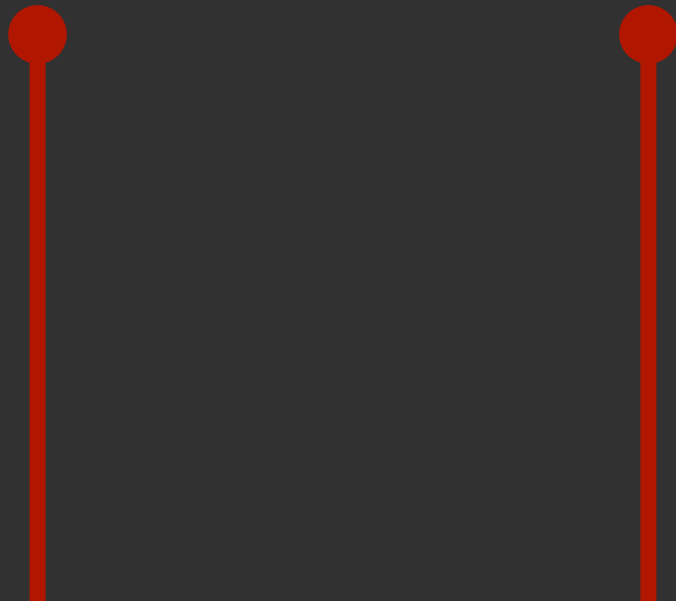
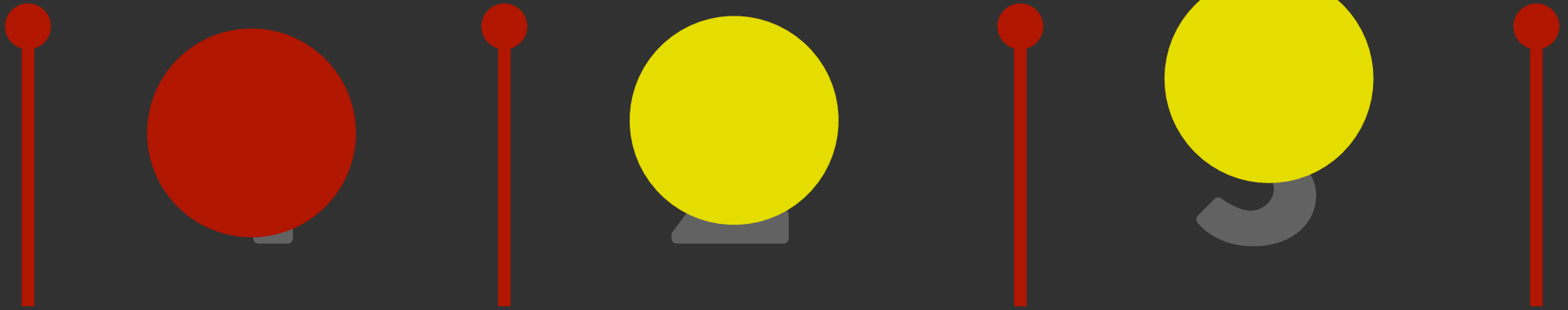




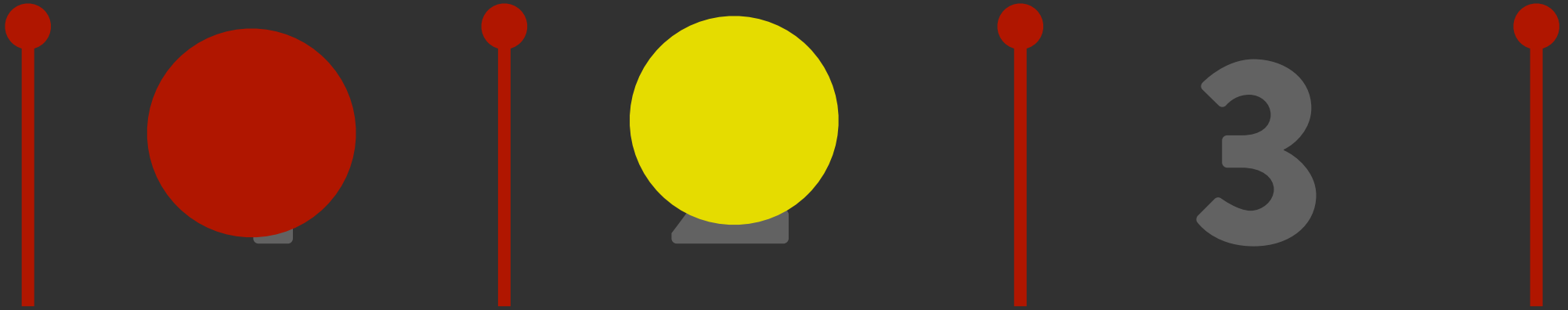


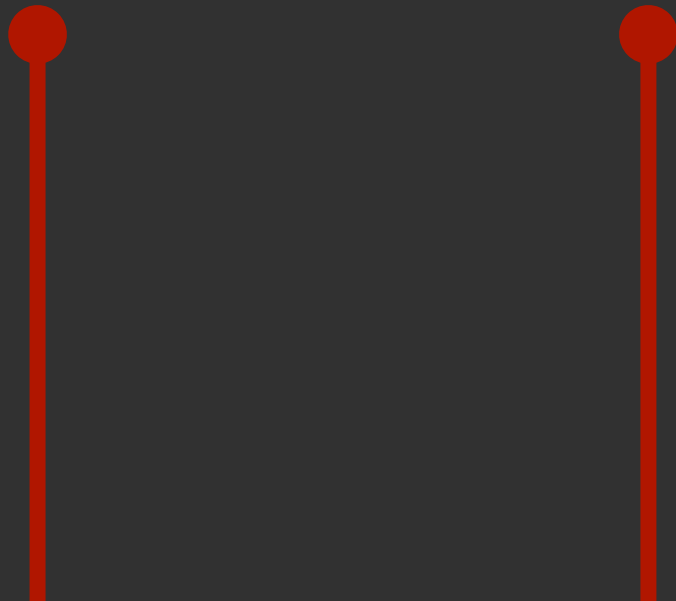












# ERLANG



*lookup: erlang B / C / engset*

**as many threads as possible**

**don't sort input URLs**

<http://abcnews.go.com/>  
<http://abcnews.go.com/2020/ABCNEWSspecial/>  
<http://abcnews.go.com/2020/story?id=207269&page=1>  
<http://abcnews.go.com/2020/story?id=207269&page=1>  
<http://abcnews.go.com/GMA/JoelSiegel/story?id=1734395>  
<http://abcnews.go.com/International/News/story?id=203089&page=1>  
<http://abcnews.go.com/International/Pope/>  
<http://abcnews.go.com/International/story?id=81417&page=1>



<http://abcnews.go.com/>  
<http://abcnews.go.com/2020/ABCNEWSspecial/>  
<http://abcnews.go.com/2020/story?id=207269&page=1>  
<http://abcnews.go.com/2020/story?id=207269&page=1>  
<http://abcnews.go.com/GMA/JoelSiegel/story?id=1734395>  
<http://abcnews.go.com/International/News/story?id=203089&page=1>  
<http://abcnews.go.com/International/Pope/>  
<http://abcnews.go.com/International/story?id=81417&page=1>

fetch



<http://abcnews.go.com/>  
<http://abcnews.go.com/2020/ABCNEWSspecial/>  
<http://abcnews.go.com/2020/story?id=207269&page=1>  
<http://abcnews.go.com/2020/story?id=207269&page=1>  
<http://abcnews.go.com/GMA/JoelSiegel/story?id=1734395>  
<http://abcnews.go.com/International/News/story?id=203089&page=1>  
<http://abcnews.go.com/International/Pope/>  
<http://abcnews.go.com/International/story?id=81417&page=1>

wait





<http://abcnews.go.com/>  
<http://abcnews.go.com/2020/ABCNEWSspecial/>  
<http://abcnews.go.com/2020/story?id=207269&page=1>  
<http://abcnews.go.com/2020/story?id=207269&page=1>  
<http://abcnews.go.com/GMA/JoelSiegel/story?id=1734395>  
<http://abcnews.go.com/International/News/story?id=203089&page=1>  
<http://abcnews.go.com/International/Pope/>  
<http://abcnews.go.com/International/story?id=81417&page=1>

fetch



<http://abcnews.go.com/>  
<http://abcnews.go.com/2020/ABCNEWSspecial/>  
<http://abcnews.go.com/2020/story?id=207269&page=1>  
<http://abcnews.go.com/2020/story?id=207269&page=1>  
<http://abcnews.go.com/GMA/JoelSiegel/story?id=1734395>  
<http://abcnews.go.com/International/News/story?id=203089&page=1>  
<http://abcnews.go.com/International/Pope/>  
<http://abcnews.go.com/International/story?id=81417&page=1>

wait



<http://abcnews.go.com/>  
<http://abcnews.go.com/2020/ABCNEWSspecial/>  
<http://abcnews.go.com/2020/story?id=207269&page=1>  
<http://abcnews.go.com/2020/story?id=207269&page=1>  
<http://abcnews.go.com/GMA/JoelSiegel/story?id=1734395>  
<http://abcnews.go.com/International/News/story?id=203089&page=1>  
<http://abcnews.go.com/International/Pope/>  
<http://abcnews.go.com/International/story?id=81417&page=1>

fetch



<http://abcnews.go.com/>  
<http://abcnews.go.com/2020/ABCNEWSspecial/>  
<http://abcnews.go.com/2020/story?id=207269&page=1>  
<http://abcnews.go.com/2020/story?id=207269&page=1>  
<http://abcnews.go.com/GMA/JoelSiegel/story?id=1734395>  
<http://abcnews.go.com/International/News/story?id=203089&page=1>  
<http://abcnews.go.com/International/Pope/>  
<http://abcnews.go.com/International/story?id=81417&page=1>

wait

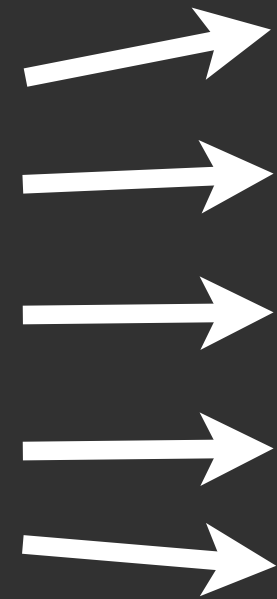


<http://abcnews.go.com/>  
<http://abcnews.go.com/2020/ABCNEWSspecial/>  
<http://abcnews.go.com/2020/story?id=207269&page=1>  
<http://abcnews.go.com/2020/story?id=207269&page=1>  
<http://abcnews.go.com/GMA/JoelSiegel/story?id=1734395>  
<http://abcnews.go.com/International/News/story?id=203089&page=1>  
<http://abcnews.go.com/International/Pope/>  
<http://abcnews.go.com/International/story?id=81417&page=1>



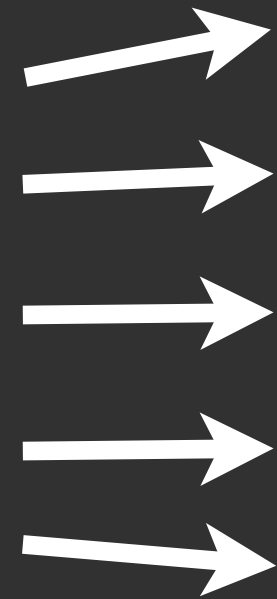
<http://yachtmaintenanceco.com/>  
<http://www.amsterdamports.nl/>  
<http://www.4s-dawn.com/>  
<http://www.embassysuiteslittlerock.com/>  
[http://members.tripod.com/airfields\\_freeman/NM/Airfields\\_NM\\_NW.htm](http://members.tripod.com/airfields_freeman/NM/Airfields_NM_NW.htm)  
<http://mdgroover.iweb.bsu.edu>  
<http://music.imbc.com/>  
<http://www.robertjbradshaw.com>  
<http://www.kerkattenhoven.be>  
<http://www.escolania.org/>  
<http://www.musiciansdfw.org/>  
<http://www.ariana.org/>

<http://yachtmaintenanceco.com/>  
<http://www.amsterdamports.nl/>  
<http://www.4s-dawn.com/>  
<http://www.embassysuiteslittlerock.com/>  
[http://members.tripod.com/airfields\\_freeman/NM/Airfields\\_NM\\_NW.htm](http://members.tripod.com/airfields_freeman/NM/Airfields_NM_NW.htm)  
<http://mdgroover.iweb.bsu.edu>  
<http://music.imbc.com/>  
<http://www.robertjbradshaw.com>  
<http://www.kerkattenhoven.be>  
<http://www.escolania.org/>  
<http://www.musiciansdfw.org/>  
<http://www.ariana.org/>



**no waiting!**

<http://yachtmaintenanceco.com/>  
<http://www.amsterdamports.nl/>  
<http://www.4s-dawn.com/>  
<http://www.embassysuiteslittlerock.com/>  
[http://members.tripod.com/airfields\\_freeman/NM/Airfields\\_NM\\_NW.htm](http://members.tripod.com/airfields_freeman/NM/Airfields_NM_NW.htm)  
<http://mdgroover.iweb.bsu.edu>  
<http://music.imbc.com/>  
<http://www.robertjbradshaw.com>  
<http://www.kerkattenhoven.be>  
<http://www.escolania.org/>  
<http://www.musiciansdfw.org/>  
<http://www.ariana.org/>





*challenges:*

# EXTRACTING URLS

*challenges:*

# ▶ EXTRACTING URLS

the internet is full of garbage

*challenges:*

# ▶ EXTRACTING URLS

*challenges:*

# ▶ **EXTRACTING URLS**

enormous pages

*challenges:*

# ▶ **EXTRACTING URLS**

enormous pages

terrible markup

*challenges:*

# ▶ **EXTRACTING URLS**

enormous pages

terrible markup

ridiculous urls

*challenges:*

# ▶ EXTRACTING URLS

enormous pages

terrible markup

ridiculous urls

 .net/

*challenges:*

# ▶ EXTRACTING URLS

enormous pages

terrible markup

ridiculous urls

 .net/

**“unicode snowman dot net”**



*challenges:*

# ▶ **EXTRACTING URLS**

**be prepared:**

*challenges:*

# ▶ **EXTRACTING URLS**

**be prepared:**

use a streaming XML parser

*challenges:*

# ▶ **EXTRACTING URLS**

**be prepared:**

use a streaming XML parser

use a library that handle's bad markup

*challenges:*

# ▶ **EXTRACTING URLS**

**be prepared:**

use a streaming XML parser

use a library that handle's bad markup

be aware that URLs aren't ASCII

*challenges:*

# ▶ **EXTRACTING URLS**

**be prepared:**

use a streaming XML parser

use a library that handle's bad markup

be aware that URLs aren't ASCII

use a URL normalizer

# SOFTWARE

# software advice:

# software advice:

- goals determine scale



# software advice:

- goals determine scale
- someone else has already done it

# 2 second crawler:

```
function wgetspider() {  
    wget --html-extension --convert-links --mirror \  
        --page-requisites --progress=bar --level=5 \  
        --no-parent --no-verbose \  
        --no-check-certificate "$@";  
}  
  
$ wgetspider http://www.ischool.berkeley.edu/
```

# java crawlers:

# java crawlers:

- Heritrix (Internet Archive)

# java crawlers:

- Heritrix (Internet Archive)
- Nutch (Lucene)

# java crawlers:

- Heritrix (Internet Archive)
- Nutch (Lucene)
- Bixo (Hadoop / Cascading)

# java crawlers:

- Heritrix (Internet Archive)
- Nutch (Lucene)
- Bixo (Hadoop / Cascading)

<http://crawler.archive.org/>

<http://nutch.apache.org/>

<http://bixo.101tec.com/>

# extraction packages:



# extraction packages:

- mechanize

# extraction packages:

- mechanize
- BeautifulSoup & urllib2

# extraction packages:

- mechanize
- BeautifulSoup & urllib2
- Scrapy

# extraction packages:

- mechanize
- BeautifulSoup & urllib2
- Scrapy

<http://wwwsearch.sourceforge.net/mechanize/>

<http://www.crummy.com/software/BeautifulSoup/>

<http://scrapy.org/>

# wrapper induction(ish)

# wrapper induction(ish)

- Ariel

# wrapper induction(ish)

- Ariel
- RoadRunner

# wrapper induction(ish)

- Ariel
- RoadRunner
- TemplateMaker



# wrapper induction(ish)

- Ariel
- RoadRunner
- TemplateMaker
- scrubyt

# wrapper induction(ish)

- Ariel
- RoadRunner
- TemplateMaker
- scrubyt

<http://ariel.rubyforge.org/index.html>

<http://www.dia.uniroma3.it/db/roadRunner/>

<http://code.google.com/p/templatemaker/>

<http://scrubyt.rubyforge.org/files/README.html>

# SCRAPING TUTORIAL (code)



**QUESTIONS?**

**FEEDBACK:**

**nate@xcombinator.com**

**www.xcombinator.com**

**@xcombinator**