Plan for Today

- Recitation Qs
- Setting for MapReduce
- Design
- Failures

Logistics

* Schedule your proj presentation
* Design project updates released
* Volunteers for recitation Qs next week?
* Reminder: feedback@henrycg.com
Think in base 2:

1,024 = \(2^{10}\) = thousand (\(\text{KiB}\))

\(2^{20}\) = million (\(\text{MB}\))

\(2^{30}\) = billion (\(\text{GB}\))

\(2^{40}\) = trillion (\(\text{TB}\))

\(2^{80}\) = million billion (\(\text{PB}\))
Recitation Qs

1. What are perf goals of MapReduce?
   - Easier to process & gen big data

2. How was M.R. implemented?
   - Framework - user writes map & reduce fns.
   - Load balancing
   - Fault tolerance (cheap HV) - pings
   - Stragglers
   - Colocation of data & compute

3. Why was M.R. implemented this way?
   - Focus on fault tolerance
   - Parallelism
The Setting

Malware Finder
- Look at every page
- Run detection alg
- Output list of suspect sites

MalwareFinder (Safe Browsing)

- Search
- Maps
- Analytics
- Images
- Scholar
- Translate
- News
- Shopping (E-commerce)
- Ads
- Vast data (web)
- Thousands of machines
- Used for many applications

When you have $2^{60}$ bytes of data, simple things become hard.
Idea: Give programmers a simple API.

\[
\text{map}( \text{key}_1, \text{val}_1 ) \rightarrow [(\text{key}_2, \text{val}_2), \ldots ]
\]

\[
\text{reduce}( \text{key}_2, [\text{val}_2, \ldots ] ) \rightarrow [\text{val}_2, \ldots ]
\]

Example: PageRank

For each URL \( u \), how many pages link to \( u \)?

\[
\text{map}( \text{src-url}, \ldots ) \rightarrow
\]

\[
\text{reduce}( \ldots )
\]
map

webpage

src_url

url1

url2

url3

map

[(url1, 1), (url2, 2), (url3, 1)]
map(url, html) {
    // Find all links in html page
    // Append to list
    // Output the list.
}
reduce (key2, [val2, ...]) → [val2, ...]
1. One main server (as in GFS.)

2.
When not to use...
Plan for Today

- Recitation Qs
- Setting for MapReduce
- Design
- Failures

Logistics

- Schedule your proj presentation
- Design project updates released
- Volunteers for recitation Qs next week?
- Reminder: feedback.henry.cg.com
Recitation Qs

1. What are the perf goals of M.R?
   - big jobs...run in parallel?

2. How was M.R implemented to meet these goals?
   - library
   - split input, map input \rightarrow intermediate results, reduce \rightarrow output

3. Why was M.R implemented this way?
   - Abstract away the detail
   - User focuses on task at hand
   (load balancing, fault recovery)
-> Download web
-> Google
  -> Search, ranking
  -> Images
  -> News
  -> Scholar
  -> Ads
  -> Shopping
  -> Translation
  -> Analytics
  -> Llamas
-> Lots of data = $2^{60}$ bytes
-> Thousands of machines
-> Many applications

"Hadoop"
Idea: Give programmers a simple interface to a huge data set.

API:

\[
\text{Data set: } \{ (\text{key}_1, \text{val}_1), \ldots \} \\
\text{map (key}_1, \text{val}_1) \rightarrow [ (\text{key}_2, \text{val}_2), \ldots ] \\
\text{reduce (key}_2, [\text{val}_2, \ldots]) \rightarrow \text{val}_2
\]

Example: Find all domains containing "llamas".

\[
\text{Data: } \{ (\text{URL}, \text{html content}), \ldots \} \\
\text{map (URL, html)} \rightarrow [ (\text{hostname}, 1) \\
\rightarrow [ ] < \\
\text{reduce (hostname, [1]) } \leftarrow \text{almost trivial} \\
\text{Output: (key}_2, [\text{val}_2, \ldots])
\]
In Python:

```
map

\[
[a, b, c, d, \ldots] \rightarrow [f(a), f(b), f(c), \ldots, \ldots]
\]
```

Reduce

```
\[
[a, b, c, d, \ldots] \rightarrow \text{out} = g(g(g(a, b), c), d)
\]
```
MapReduce

User doesn't have to worry about:
* who runs the code
* fault tolerance
* storage
* locality
* strugglers
Example of map fn w/ many outputs?

\[
\text{map}(\text{key1, val1}) \rightarrow [(\text{key2, val2}), \ldots]
\]
Poll: MapReduce it?
Plan: MapReduce

* The setting
* System design
* MapReduce it?
* Failures

Logistics

* Schedule your project presentation!
* Design project updates released
The Setting

- Download pages in crawl
- Google
  - Images
  - Translate
  - Search index & rank
  - News
  - Training ML models
  - User data mgmt analytics
  - Shopping
    - Find all the pages hosting malware

The web

- Lots of data \( \approx \) 2^9 bytes
- 10,000s machines
- Many applications
Idea: Give programmers a simple way to interact with very large data set.

API: data set: \{(key, value), \ldots\}

map(key, val1) \rightarrow [(key2, val2), \ldots, \ldots]

reduce(key2, [val2, \ldots, \ldots]) \rightarrow val2

User implements map, reduce.
System handles: failures, scheduling, data placement, machines, load balancing.
Example: Page popularity

Q: For each URL u, how many pages link to URL u?

Let search

Data: \( \{(\text{page-name, page-contents}), \ldots\} \)

map (\text{page-name, page-contents}) \rightarrow [(\text{url1, 1}), (\text{url2, 1}), \ldots] 

// Find all outgoing links on page
// Write their URLs to output

reduce (\text{url}, [1, 1, 1, 1, 1]) \rightarrow 5 

// Count the # of 1s in input
Hash(url) → \{0, 1, 2, 3\}

MAP

\[ \{(url1, 1), (url3, 1)\} \]

Reduce

\((url1, 1)\)

\[ \{(url3, 1), (url4, 1)\} \]

Reduce

\(url3\)

\(url4\)

\[ \{(url3, 1), (url4, 1)\} \]

Reduce

\(url3\)

\(url4\)
Say you have $n$ machines in a data center. Each fails w.p. $\frac{1}{1000}$.

\[
\begin{align*}
\Pr[\text{main fail}] &= \frac{1}{1000} \\
\Pr[\text{worker fails}] &= 1 - \left(1 - \frac{1}{1000}\right)^n
\end{align*}
\]

$n \approx 1000$ \quad \frac{1}{2}