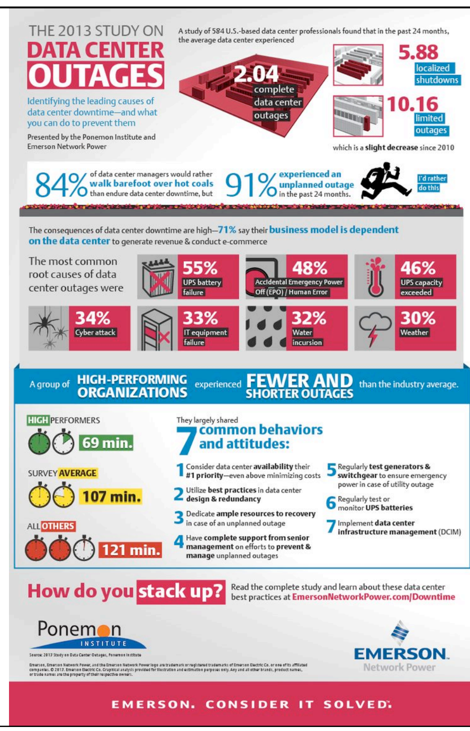
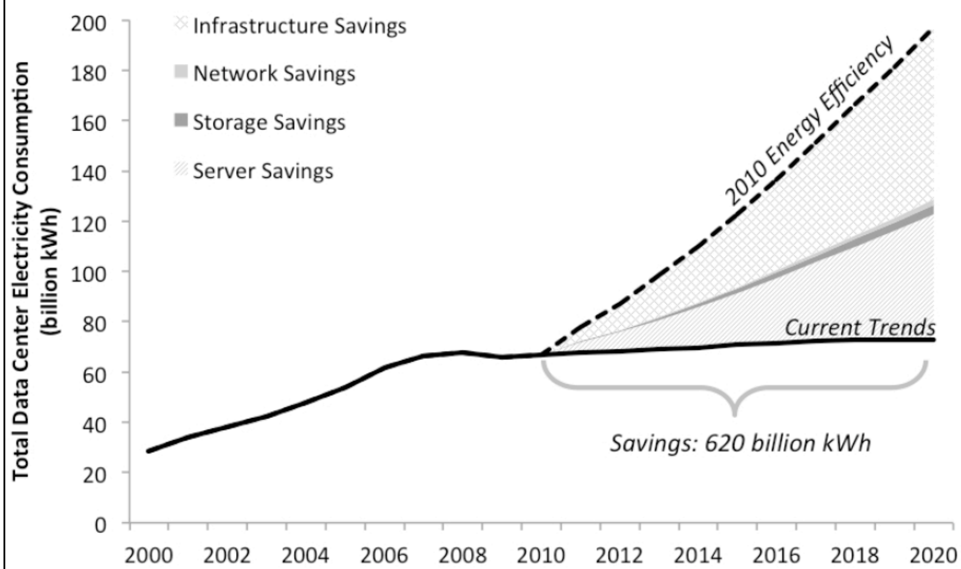
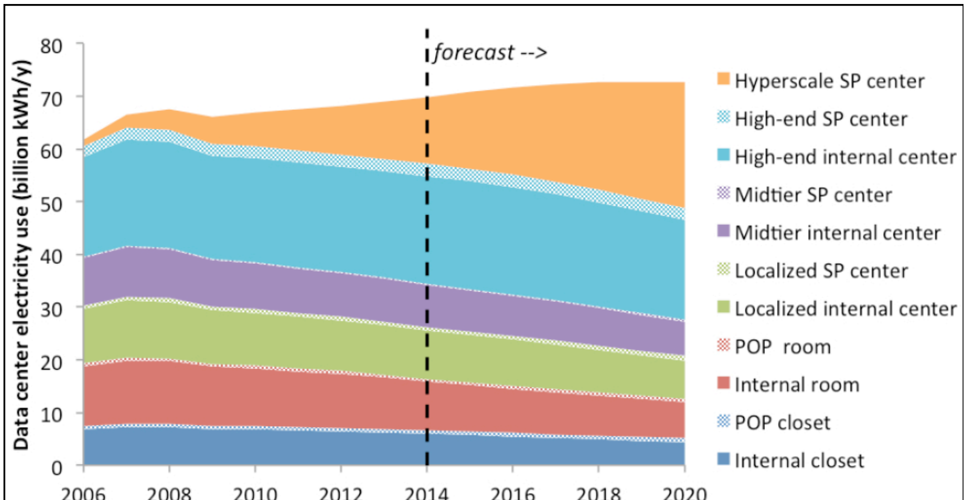
Today's Objectives

- MapReduce
- Hadoop
- Amazon Web Services



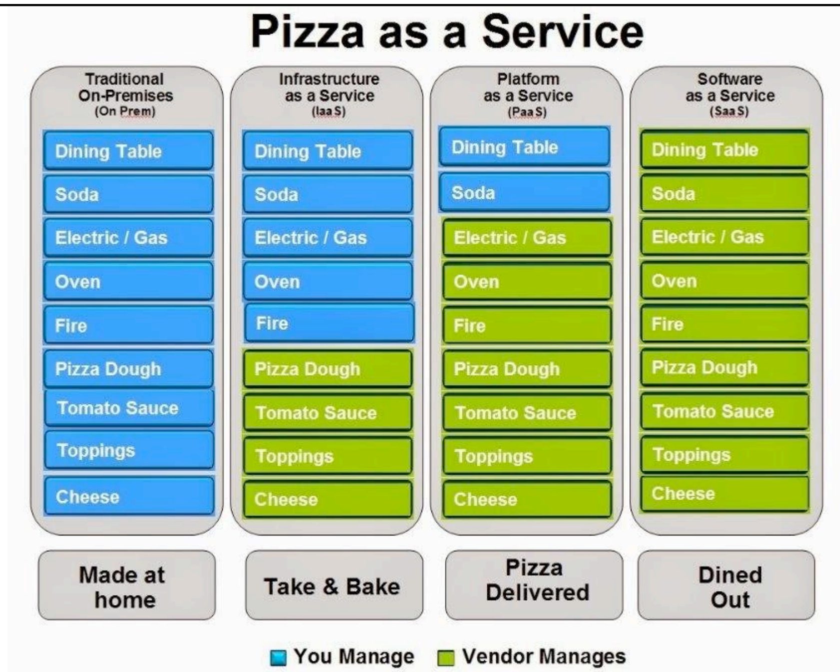


http://www.datacenterknowledge.com/archives/2016/06/27/heres-how-much-energy-all-us-data-centers-consume



SP: service providers

http://www.datacenterknowledge.com/archives/2016/06/27/heres-how-much-energy-all-us-data-centers-consume



SaaS and PaaS

- **SaaS: Software as a Service**
  - an application is hosted as a service provided to customers across the Internet
  - Saas alleviates the burden of software maintenance/ support
  - but users relinquish control over software versions and requirements
- **PaaS: Platform as a Service**
  - provides a computing platform and a solution stack as a service
  - Consumer creates the software using tools and/or libraries from the provider
  - Consumer controls software deployment and configuration settings.
  - Provider provides the networks, servers, storage and other services

IaaS: Infrastructure as a Service

- IaaS providers offer virtual machines, virtual-machine image libraries, raw (block) and file-based storage, firewalls, load balancers, IP addresses, virtual local area networks (VLANs), and software bundles.
- Pools of hypervisors can scale services up and down according to customers' varying requirements
- All infrastructure is provided on-demand

MAPREDUCE

MapReduce

- What were your main takeaways?
  - Why?
  - What?
  - Who?
  - How?

Oct 25, 2017 Sprenkle - CSCI325 9

Discussion

- What are the motivation, challenges, and goals for MapReduce?

Oct 25, 2017 Sprenkle - CSCI325 10

Motivation: Large-Scale Data Processing

- Want to process lots of data (> 1 TB)
- Want to parallelize across hundreds/thousands of CPUs
  - Probably more in reality…
- And we want to make this *easy*
  - Programming for distributed systems is complex

Discussion

- MapReduce: what applications have they used it for?

Sample Applications

- Distributed grep
- Count of URL access frequency
- Reverse Web-link graph
- Inverted index
- Distributed sort

Discussion

- What features does MapReduce provide?

MapReduce

- Automatic parallelization & distribution of large-scale computations
- Fault-tolerant
  - handles machine failures gracefully
- Provides status and monitoring tools
- Clean abstraction for programmers

Programming Model

- Borrows from functional programming
- Input & Output: each a set of key/value pairs
- Users implement interface of two functions:

```
map(in_key, in_value) ->
     (out_key, intermediate_value) list

reduce(out_key, intermediate_value list) ->
     out_value list
```

Who has used functional languages?

Programming Model: map

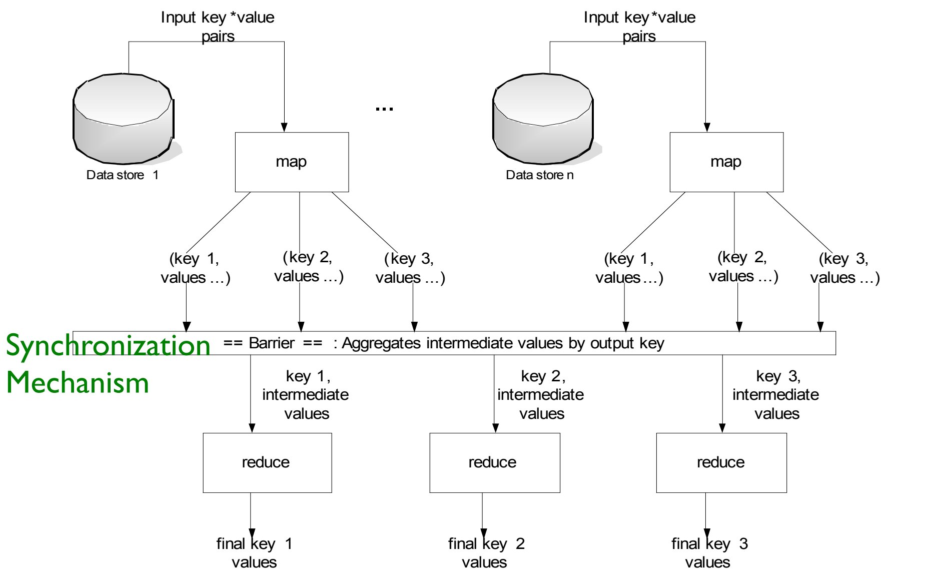
```
map(in_key, in_value) ->
     (out_key, intermediate_value) list
```

- ***Records*** from the data source (lines out of files, rows of a database, etc) are fed into the map function as key*value pairs: e.g., (filename, line)
- `map()` produces one or more intermediate values along with an output key from the input

Programming Model: reduce

```
reduce(out_key, intermediate_value list) ->
     out_value list
```

- After the map phase, all intermediate values for a particular output key are combined together into a list
- `reduce()` combines those intermediate values into one or more *final values* for that same output key
  - in practice, usually only one final value per key



Parallelism

- `map()` functions run in parallel, creating different intermediate values from different input data sets
- `reduce()` functions also run in parallel, each working on a different output key
- All values are processed *independently*
- Bottleneck: reduce phase can't start until map phase is completely finished.

Example: Count word occurrences

```
map(String input_key, String input_value):
    // input_key: document name
    // input_value: document contents
    for each word w in input_value:
       EmitIntermediate(w, "1");

reduce(String output_key, Iterator intermediate_values):
    // output_key: a word
    // output_values: a list of counts
    int result = 0;
    for each v in intermediate_values:
        result += ParseInt(v);
    Emit(AsString(result));
```


Example vs. Actual Source Code

- Example is written in pseudo-code
- Actual implementation is in C++, using a MapReduce library
- Bindings for Python and Java exist via interfaces
- True code is somewhat more involved (defines how the input key/values are divided up and accessed, etc.)

Discussion

- What are some optimizations that MapReduce utilizes?

Locality

- Master program divides tasks based on location of data: tries to have `map()` tasks on same machine as physical file data or at least same rack
- `map()` task inputs are divided into 64 MB blocks: same size as Google File System chunks

Fault Tolerance

- Master detects worker failures
  - Re-executes completed & in-progress `map()` tasks
  - Re-executes in-progress `reduce()` tasks
- Master notices particular input key/values cause crashes in `map()` and skips those values on re-execution
  - Effect: Can work around bugs in third-party libraries!

Optimizations

- No `reduce` can start until `map` is complete:
  - A single slow disk controller can rate-limit the whole process
- Master redundantly executes "slow-moving" map tasks; uses results of first copy to finish
  - This is the "stragglers" problem

Why is it safe to redundantly execute map tasks? Wouldn't this mess up the total computation?

Optimizations

- "Combiner" functions can run on same machine as a mapper
- Causes a mini-reduce phase to occur before the real reduce phase, to save bandwidth

MapReduce Conclusions

- MapReduce has proven to be a useful abstraction
- Greatly simplifies large-scale computations at Google
- Functional programming paradigm can be applied to large-scale applications
- Fun to use: focus on problem, let library deal w/ messy details

Discussion

- What do you think about MapReduce?
- Any problems or limitations?
- Lessons learned?
- Your questions?

Does Google Use MapReduce?

- Google Dumps MapReduce in Favor of New Hyper-Scale Analytics System
  - Cloud Dataflow
  - http://www.datacenterknowledge.com/archives/2014/06/25/google-dumps-mapreduce-favor-new-hyper-scale-analytics-system

APACHE hadoop

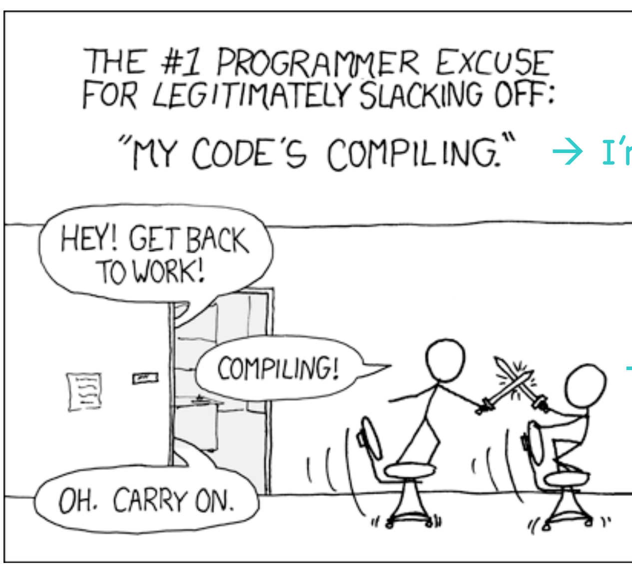
APACHE hadoop

- Framework written in Java for running applications on large clusters of commodity hardware
- Incorporates features similar to those of Google File System and of MapReduce
- Used to break complicated problems apart, spreading them across many computers
- Open-source implementation of MapReduce, and its own filesystem HDFS (Hadoop distributed file system)

The Hadoop Ecosystem

Hadoop Common	• Contains Libraries and other modules
HDFS	• Hadoop Distributed File System
Hadoop YARN	• (Yet Another Resource Negotiator) • Job scheduling and resource manager
Hadoop MapReduce	• A programming model for large scale data processing

Evolution of Comics



→ I'm MapReducing!

→ MapReducing!

WordCount Mapper in Java

```
public static class TokenizerMapper
      extends Mapper<Object, Text, Text, IntWritable> {

    private final static IntWritable one = new IntWritable(1);
    private Text word = new Text();

    public void map(Object key, Text value, Context context)
              throws IOException, InterruptedException {
        StringTokenizer itr = new
StringTokenizer(value.toString());
        while (itr.hasMoreTokens()) {
            word.set(itr.nextToken());
            context.write(word, one);
         }
    }
}
```


WordCount Reducer in Java

```
public static class IntSumReducer
    extends Reducer<Text, IntWritable, Text, IntWritable> {
    private IntWritable result = new IntWritable();

    public void reduce(Text key, Iterable<IntWritable>
values, Context context)
              throws IOException, InterruptedException {
       int sum = 0;
       for (IntWritable val : values) {
          sum += val.get();
       }
       result.set(sum);
       context.write(key, result);
    }
}
```

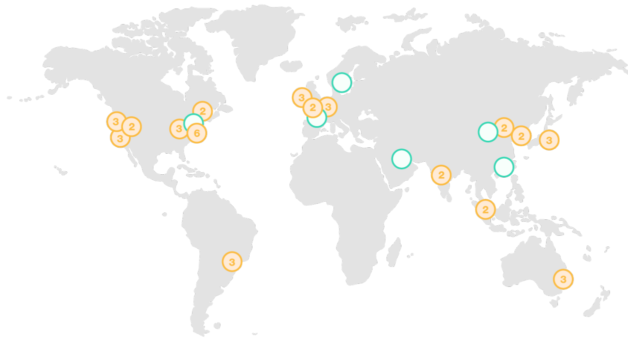
AMAZON WEB SERVICES (AWS)

What is Amazon Web Services?

- A collection of remote computing services that together make up a cloud computing platform
  - offered over the Internet by Amazon.com
- Grew out of Amazon's need to rapidly provision and configure machines of standard configurations for its own business.

http://aws.amazon.com

Amazon Web Services Architecture



- AWS is located in geographical ***Regions***
  - Region: Geographic location, price, laws, network locality.
  - wholly contained within a single country and all of its data and services stay within the designated Region.
- Each region has multiple ***Availability Zones***
  - distinct data centers providing AWS services
  - isolated from each other to prevent outages from spreading between Zones

Terminology

- Instance: One running virtual machine.
- Instance Type: hardware configuration - cores, memory, disk.
- Instance Store Volume: Temporary disk associated with instance.
- Image (AMI): Stored bits which can be turned into instances.
- Key Pair: Credentials used to access VM from command line.

Project 3

- Use MapReduce and Amazon clusters to create an inverted index
  - What is an inverted index?
- Write mapper and reducer
- Check out resources, run through the tutorials
  - Don't get overwhelmed!
  - Important part of CS is learning tools, systems on your own