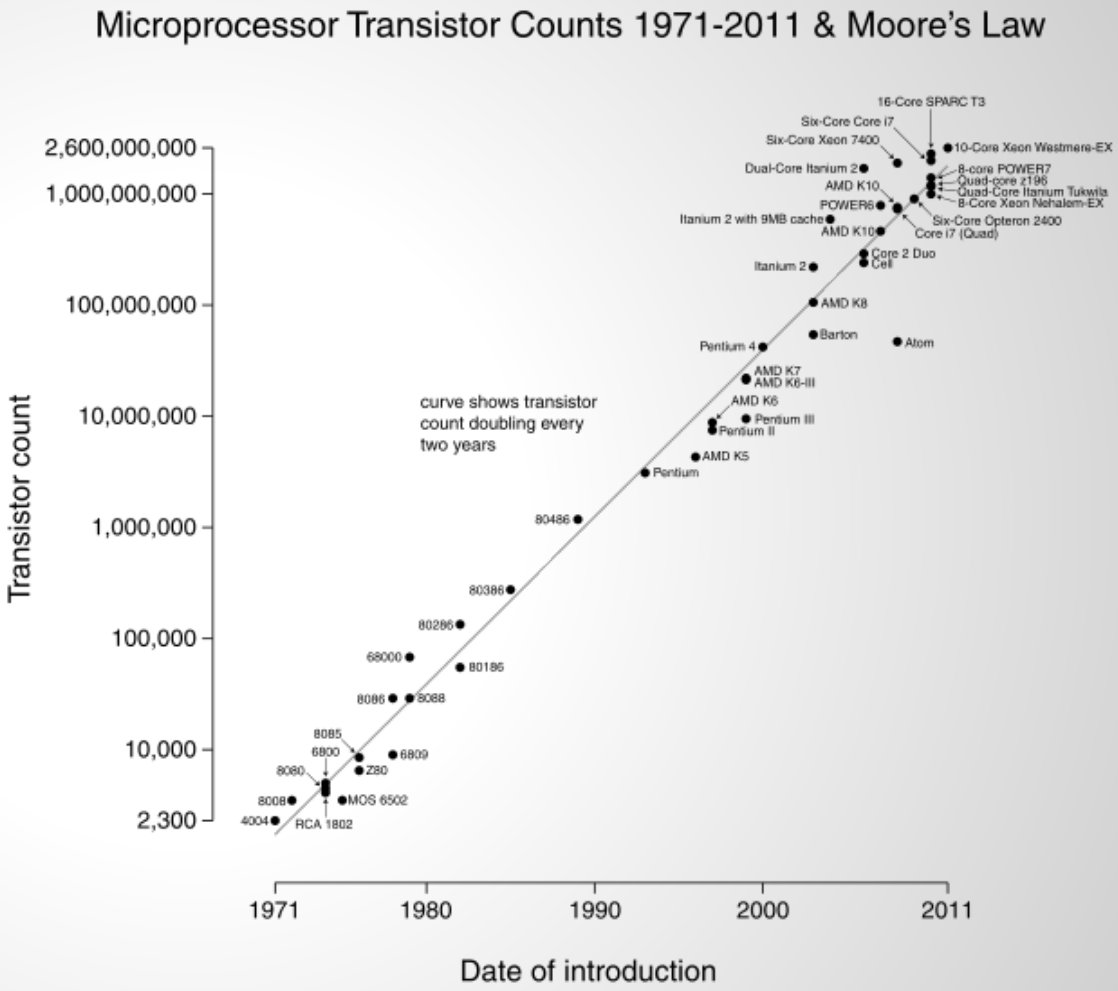
Cloud Computing

John McSpedon

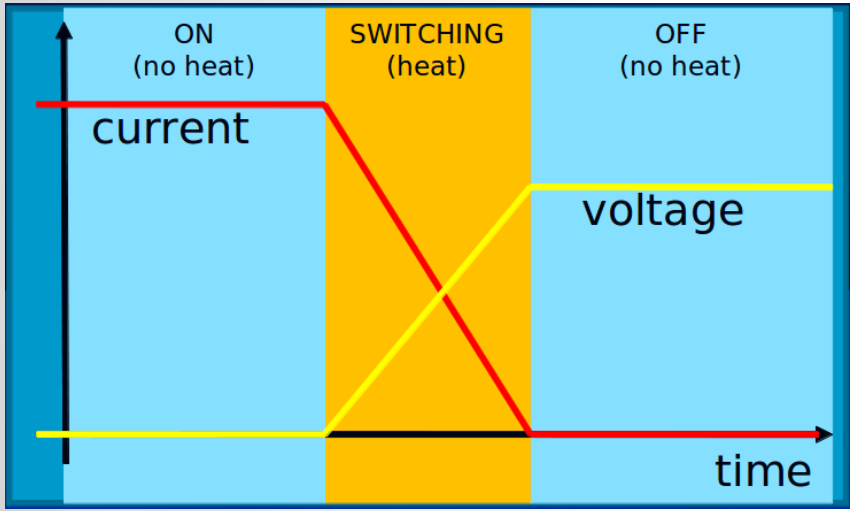
Why Parallel Computation?

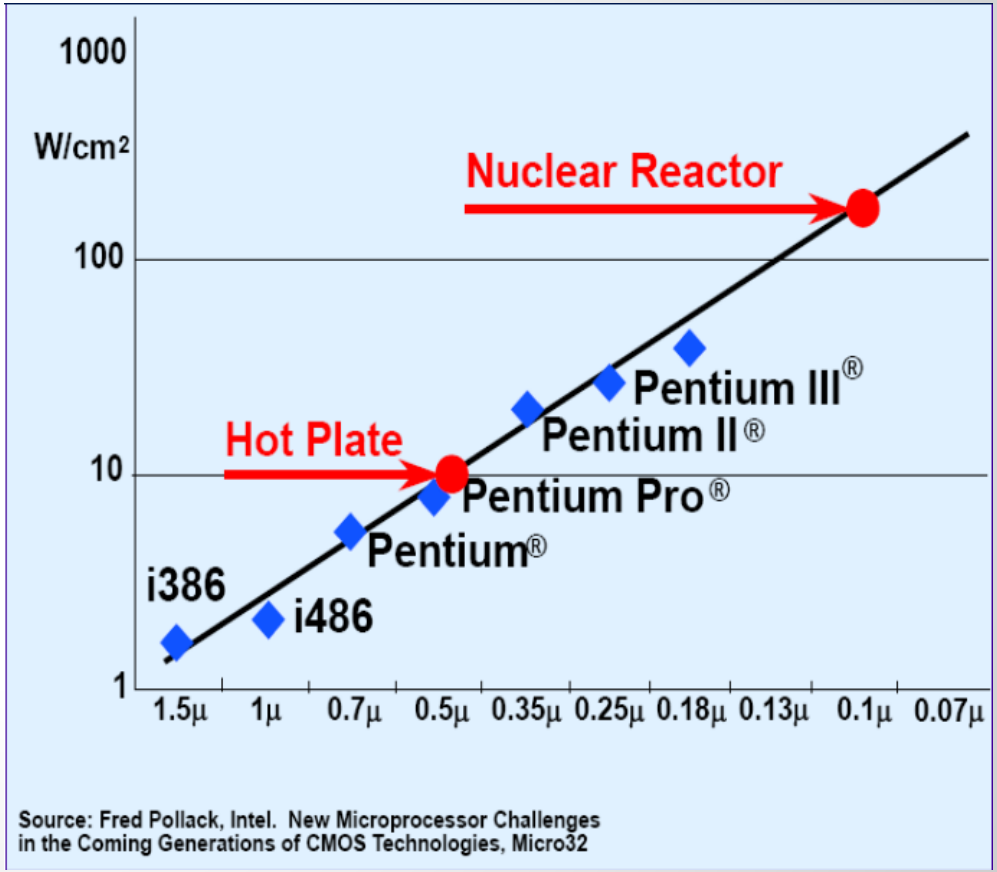
- Traditional Moore’s Law
- Signal Propagation
- Memory Access Latency
- Huge Datasets

Moore’s Law


Microprocessor Transistor Counts 1971-2011 & Moore’s Law
Transistor count
2,600,000,000
1,000,000,000
100,000,000
10,000,000
1,000,000
100,000
10,000
2,300
Date of introduction
1971
1980
1990
2000
2011
curve shows transistor count doubling every two years
16-Core SPARC T3
Six-Core Core i7
Six-Core Xeon 7400
10-Core Xeon Westmere-EX
Dual-Core Itanium 2
8-core POWER7
Quad-core z196
Quad-Core Itanium Tukwila
8-Core Xeon Nehalem-EX
AMD K10
POWER6
Itanium 2 with 9MB cache
Six-Core Opteron 2400
AMD K10
Core i7 (Quad)
Core 2 Duo
Cell
Itanium 2
AMD K8
Barton
Atom
Pentium 4
AMD K7
AMD K6-III
AMD K6
Pentium III
Pentium II
AMD K5
Pentium
80486
80386
80286
68000
80186
8086
8088
8085
6800
6809
Z80
8080
8008
MOS 6502
4004
RCA 1802

Power Density

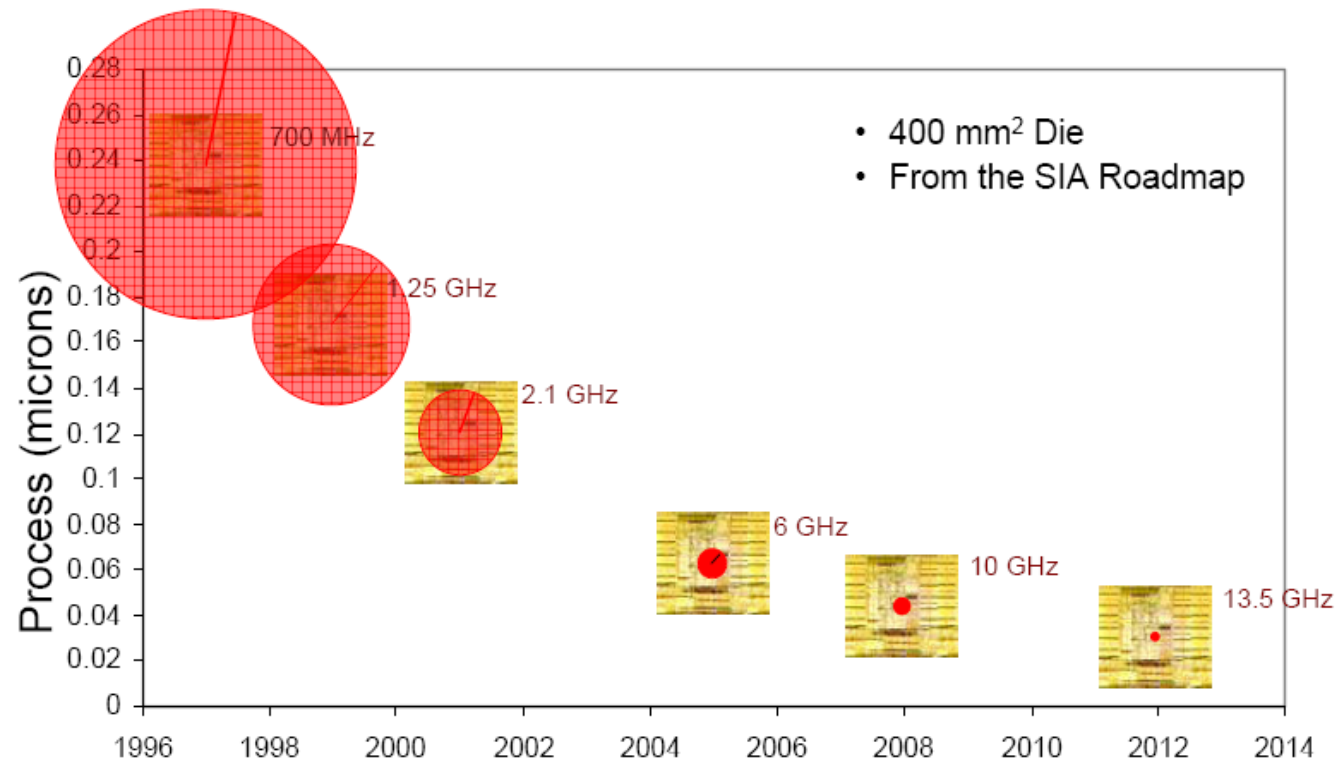




Signal Propagation

Internal signals propagate at ≈⅔ *c*

Signal radius of one clock cycle?

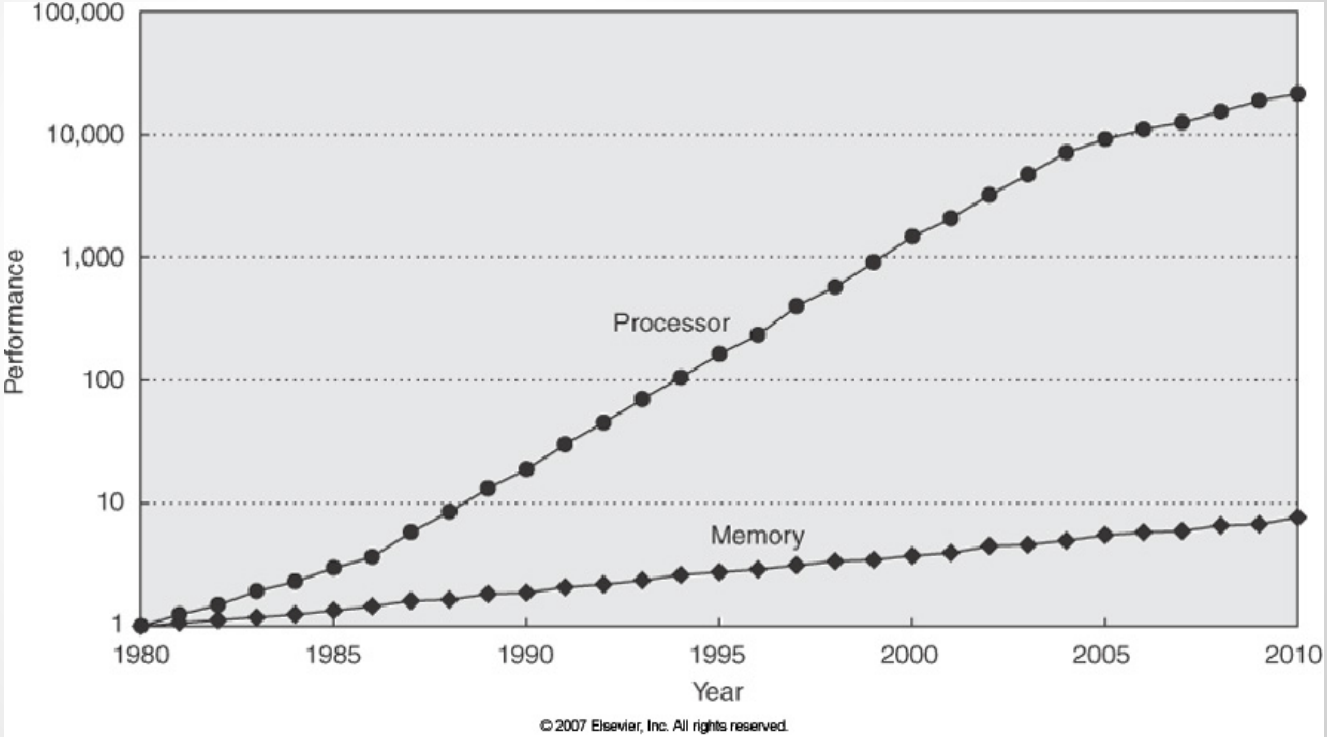


Memory Access Latency

1 machine x 1TB

or

1000 machines x 1GB



Huge Datasets

VOC 2009: 900MB

TME Motorway: 32GB

SUN database: 37GB

>900 million Websites to index

200-300 PB of images on Facebook

Parallel Computation at Princeton

- MATLAB parfor
- CS ionic cluster (PBS)
- MapReduce/Hadoop
- Amazon EC2

MATLAB parfor

ridiculously simple

```
parfor i = 1:length(A)
    B(i) = f(A(i));
end
```

```
s = 0;
parfor i = 1:n
    if p(i)    % p is fxn
        s = s + 1;
    end
end
```

requires consecutive range of integers

parfor Demo

CS ionic cluster

- ≈100 node cluster for use by CS department
- controlled by a PBS/Torque queue
- users communicate via beowulf listserv
- jobs submitted via scripts/command line from head node of ionic.cs.princeton.edu

ionic cluster nodes

27x (2 cores @ 2.2GHZ, 8+ GB RAM, 2x73GB disk)

9x (4 cores @ 2.3GHZ, 16 GB RAM, 4x146 GB disk)

48x (2 cores @ ~2 GHZ, 8 GB RAM, 1x750 GB disk)

3x (6 cores @ 3.1GHZ, 48 GB RAM, 2x146 GB disk)

ionic resources

- CS Guide intro: https://csguide.cs.princeton.edu/resources/clusters
- Job Submission Guide (see chapter 2): http://docs.adaptivecomputing.com/torque/4-2-6/torqueAdminGuide-4.2.6.pdf
- Current Node Status: http://ionic.cs.princeton.edu/ganglia/
- Queue Policy Guide: http://docs.adaptivecomputing.com/maui/pdf/mauiadmin.pdf

ionic: .sh for single processor job

Hello World files

```
mcspedon-hp-dv7:~$ ssh mcspedon@ionic.cs.princeton.edu
Last login: Wed Mar 26 17:16:43 2014 from nat-oitwireless-outside-vapornet3-b-227.princeton.edu
[mcspedon@head ~]$ cd COS598C/hello_world/
[mcspedon@head hello_world]$ gcc -o hello hello_world.c
[mcspedon@head hello_world]$ ls
hello  hello.sh  hello_world.c
[mcspedon@head hello_world]$ qsub ./hello.sh
3648004.head.ionic.cs.princeton.edu
[mcspedon@head hello_world]$ ls
hello  hello.err  hello.out  hello.sh  hello.txt  hello_world.c
[mcspedon@head hello_world]$ cat hello.out
Starting 3648004.head.ionic.cs.princeton.edu at Wed Mar 26 17:19:55 EDT 2014 on node096.ionic.cs.princeton.
edu

Hello World

Done at  Wed Mar 26 17:19:55 EDT 2014
[mcspedon@head hello_world]$ cat hello.txt
Hello Filesystem
```

ionic: single node MATLAB job

bash script to call find_k_closest_imgs.m

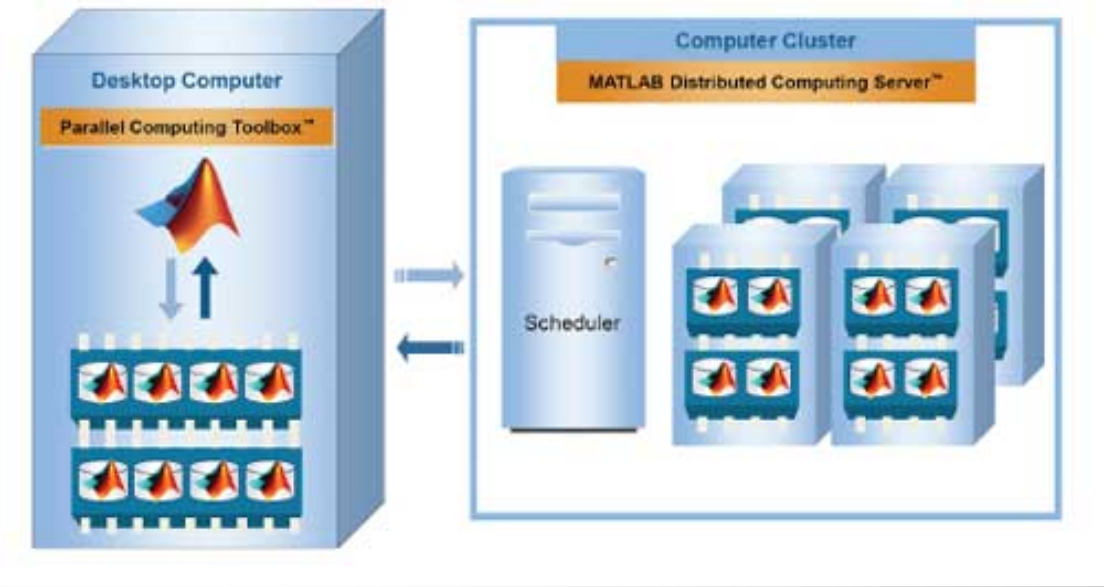
```
mcspedon-hp-dv7:~$ ssh mcspedon@ionic.cs.princeton.edu
Last login: Wed Mar 26 17:18:56 2014 from nat-oitwireless-outside-vapornet3-b-227.princeton.edu
[mcspedon@head ~]$ cd COS598C/ImageSearch/Codebase/
[mcspedon@head Codebase]$ ls
boxes_query04_20140324T161840.mat  k_closest.jpg        test_whiten.m
find_k_closest_imgs.m              learn_image.m        voc-release5
generative_RELEASE                 matlab_singlenode.sh  weighted_filter.jpg
getAllJPGs.m                       query_dir_by_img.m
initmodel_var.m                    templateMatching
[mcspedon@head Codebase]$ qsub matlab_singlenode.sh
3648005.head.ionic.cs.princeton.edu
[mcspedon@head Codebase]$ ls
boxes_query04_20140324T161840.mat  initmodel_var.m              query_dir_by_img.m
boxes_query04_20140326T172958.mat  k_closest.jpg                templateMatching
find_k_closest_imgs.m              learn_image.m                test_whiten.m
generative_RELEASE                 matlab_singlenode.sh         voc-release5
getAllJPGs.m                       matlab_singlenode.sh.o3648005  weighted_filter.jpg
```

MATLAB Distributed Computing Server

Scales Parallel Computing Toolbox

Duplicates user's MATLAB licenses

(up to 32 instances on ionic cluster)



ionic: multiple node MATLAB job

Usually called as MATLAB fxn, but MATLAB has been removed from ionic head node.

In communication with CS IT department.

Supposedly users can request a single node with 16 processors in the meantime.

MapReduce/Hadoop

- Google FS (2003)
- Google MapReduce (2004)
- Google Bigtable (2006)

Google FS

Assumptions

- commodity hardware with nonzero failure rate
- multi-GB files designed for single-write-many-reads
- append more important than random write
- high bandwidth more important than low latency

Simplest unit is 64MB *chunk*

1 master, several *chunkservers*

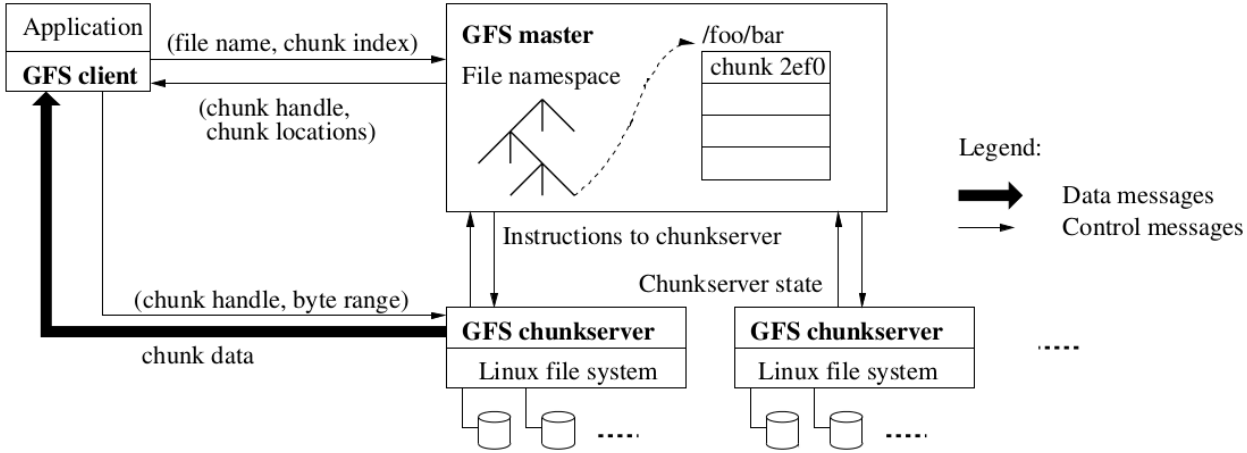
Google FS

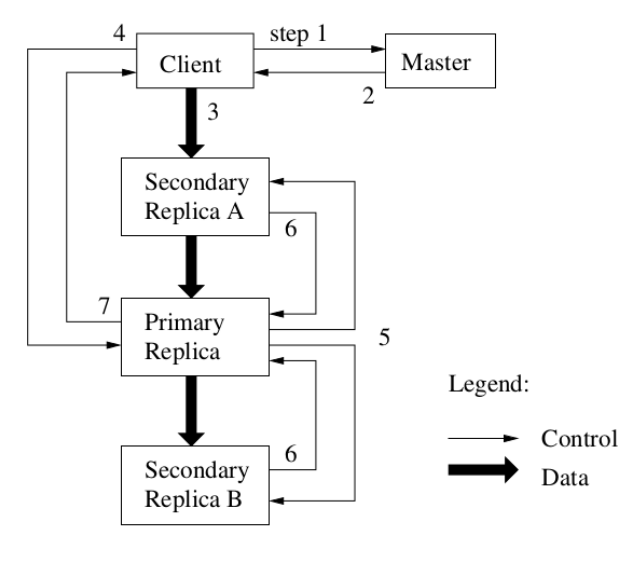
Master stores:

file/chunk namespaces,

file -> chunk(s) mapping,

chunk replica locations



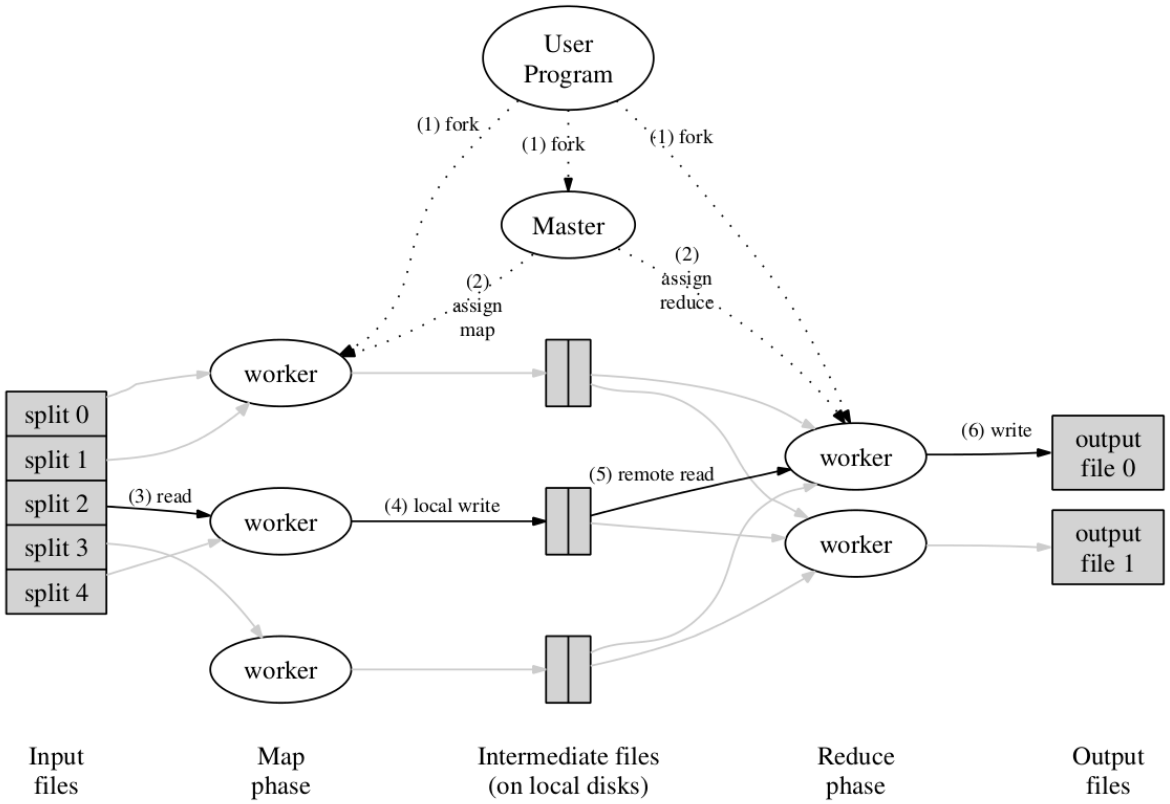


Google MapReduce

map: (k1, v1) -> list(k2, v2)
reduce: (k2, list(v2)) -> list(v2)

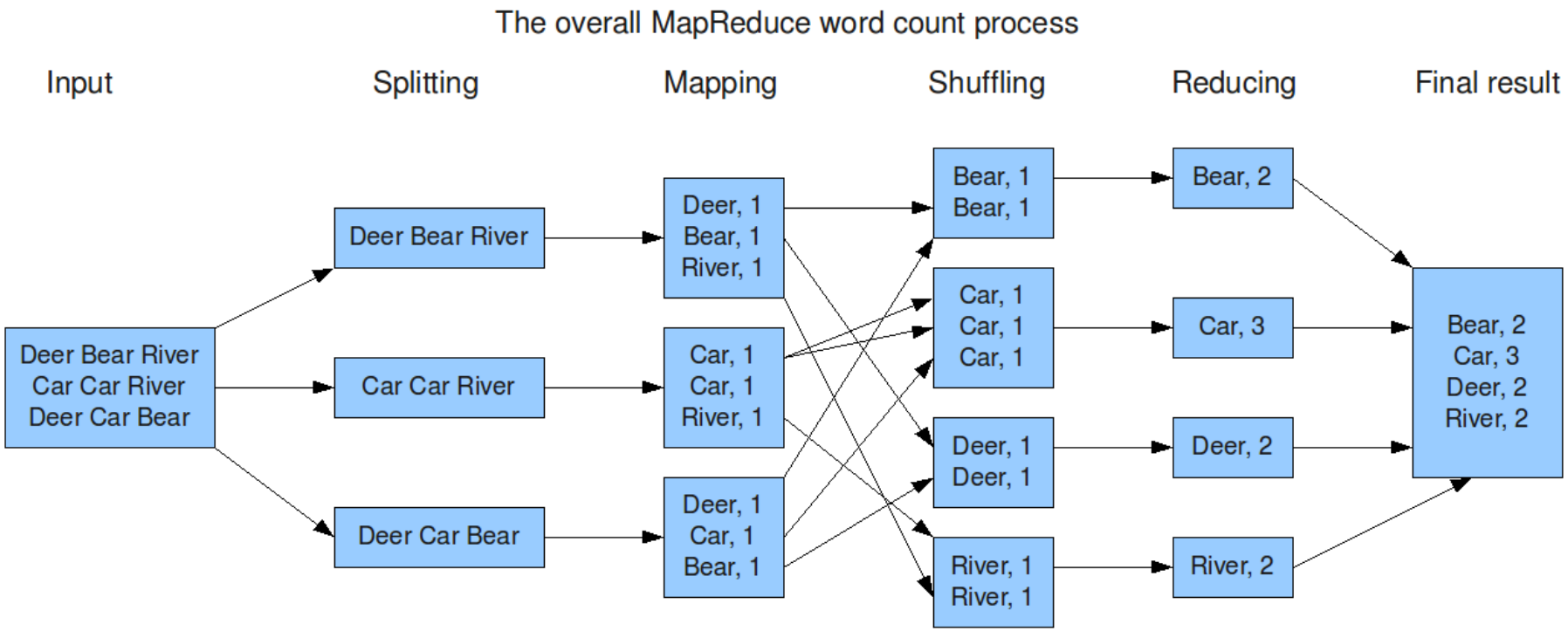
choose, e.g.
M = 200,000
R = 5,000
(2,000 workers)

WordCount
Distributed Grep
URL Access Frequency
Reverse Web-Link Graph
Distributed Sort



MapReduce: Word Count

```
map:
    for each word in input
        output (word, 1)
reduce:
    for each key
        sum(values)
```


The overall MapReduce word count process
Input
Splitting
Mapping
Shuffling
Reducing
Final result
Deer Bear River
Car Car River
Deer Car Bear
Deer Bear River
Car Car River
Deer Car Bear
Deer, 1
Bear, 1
River, 1
Car, 1
Car, 1
River, 1
Deer, 1
Car, 1
Bear, 1
Bear, 1
Bear, 1
Car, 1
Car, 1
Car, 1
Deer, 1
Deer, 1
River, 1
River, 1
Bear, 2
Car, 3
Deer, 2
River, 2
Bear, 2
Car, 3
Deer, 2
River, 2

MapReduce: Distributed Grep (1 of 2)

```
map1:
    for each line in input
        output (matching line, 1) if match
reduce1:
    for each key
        sum(values)
```

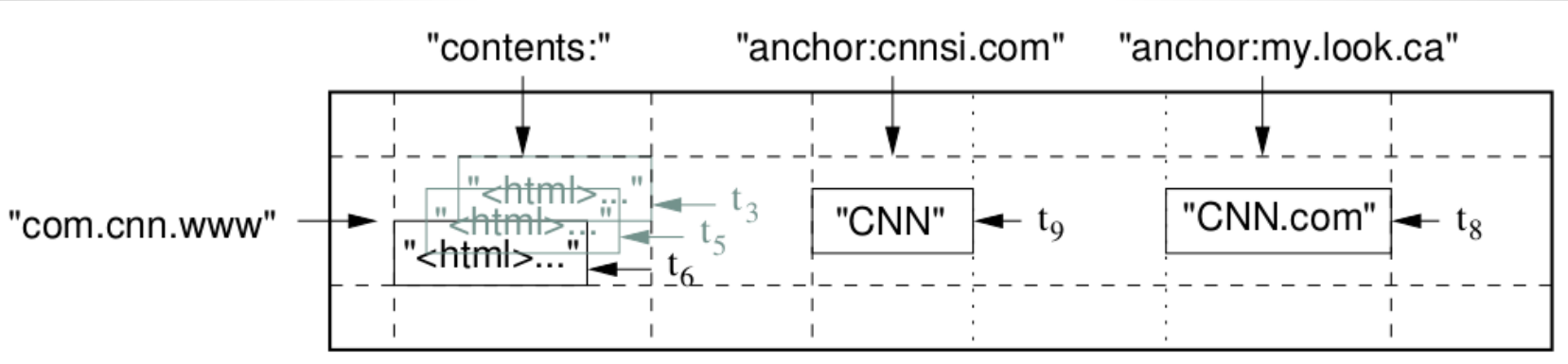
MapReduce: Distributed Grep (2 of 2)

```
map2:
    for each (matching line, freq)
        output (freq, matching line)
reduce2:
    identity fxn
    (This sorts matching lines by their frequency)
```

Google Bigtable

Built on top of Google FS, SSTable, Chubby Lock Service

Choice of row name is important for compression



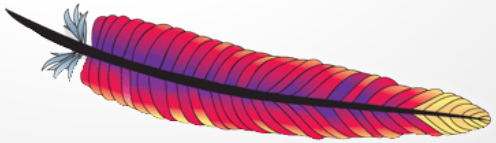
Apache Hadoop

Open source implementations of Google whitepapers

- Hadoop Distributed File System
- Hadoop MapReduce
- Apache Hbase

Yahoo! web search: 42,000 node cluster

Facebook backend: 200+PB data on HDFS/Hbase



Hadoop 2.2 Pseudo-Cluster

- Each CPU core is a worker in MapReduce job
- Communicate via network interface (ip 127.0.0.1)
- Allows user to test code without charge
- Similar steps for installing Hadoop on small clusters

Installation References

official instructions: https://hadoop.apache.org/docs/r2.2.0/hadoop-project-dist/hadoop-common/SingleNodeSetup.html#Single_Node_Setup

64-bit build with fixes for common bugs: http://www.csrdu.org/nauman/2014/01/23/geting-started-with-hadoop-2-2-0-building/

64-bit install:

http://www.csrdu.org/nauman/2014/01/25/hadoop-2-2-0-single-node-cluster/

disabling ipv6: http://askubuntu.com/questions/346126/how-to-disable-ipv6-on-ubuntu

suggested changes to .bashrc:

http://codesfusion.blogspot.com/2013/10/setup-hadoop-2x-220-on-ubuntu.html?m=1

Installation References (continued)

```
master:~$ cat /etc/hostname
master
master:~$ cat /etc/hosts
127.0.0.1       localhost master
127.0.1.1       master

# The following lines are desirable for IPv6 capable hosts
::1     ip6-localhost ip6-loopback
fe00::0 ip6-localnet
ff00::0 ip6-mcastprefix
ff02::1 ip6-allnodes
ff02::2 ip6-allrouters
net.ipv6.conf.all.disable_ipv6 = 1
net.ipv6.conf.default.disable_ipv6 = 1
net.ipv6.conf.lo.disable_ipv6 = 1
master:~$ cat /proc/sys/net/ipv6/conf/all/disable_ipv6
1
master:~$ ^^ '1' denotes that IPv6 is disabled
```

```
master:~$ tail -n13 /home/hduser/.bashrc
#Hadoop Stuff!
export JAVA_HOME=/usr/lib/jvm/jdk/
export HADOOP_INSTALL=/usr/local/hadoop
export HADOOP_HOME=/usr/local/hadoop
export HADOOP_CONF_DIR=$HADOOP_INSTALL/conf
export HADOOP_OPTS=-Djava.net.preferIPv4Stack=true
export PATH=$PATH:$HADOOP_INSTALL/bin
export PATH=$PATH:$HADOOP_INSTALL/sbin
export HADOOP_MAPRED_HOME=$HADOOP_INSTALL
export HADOOP_COMMON_HOME=$HADOOP_INSTALL
export HADOOP_HDFS_HOME=$HADOOP_INSTALL
export YARN_HOME=$HADOOP_INSTALL
export YARN_CONF_DIR=$HADOOP_INSTALL/conf
```

Hadoop Word Count: Map

```
public static class Map extends MapReduceBase implements Mapper<LongWritable, Text, Text, IntWritable> {
        private final static IntWritable one = new IntWritable(1);
        private Text word = new Text();

        public void map(LongWritable key, Text value, OutputCollector<Text, IntWritable> output, Reporter
reporter) throws IOException {
          String line = value.toString();
          StringTokenizer tokenizer = new StringTokenizer(line);
          while (tokenizer.hasMoreTokens()) {
            word.set(tokenizer.nextToken());
            output.collect(word, one);
          }
        }
      }
```

Hadoop Word Count: Reduce

```
public static class Reduce extends MapReduceBase implements Reducer<Text, IntWritable, Text, IntWritable>
{
          public void reduce(Text key, Iterator<IntWritable> values, OutputCollector<Text, IntWritable>
output, Reporter reporter) throws IOException {
            int sum = 0;
            while (values.hasNext()) {
              sum += values.next().get();
            }
            output.collect(key, new IntWritable(sum));
          }
        }
```

Hadoop Word Count demo

bash scripts

1. Check that current ip address of computer matches second line of /etc/hosts

```
mcspedon-hp-dv7:~$ cat /etc/hosts
127.0.0.1       localhost
140.180.241.239 master
```

2. Call startup.sh
3. If ‘jps’ returns the following processes…

```
3198 DataNode
3855 NameNode
3357 NodeManager
4013 Jps
3954 ResourceManager
```

4. Call wordcount.sh

Amazon Elastic Compute Cloud (EC2)

- Low overhead costs
- Outsource cluster management
- Access large-storage/ GPU devices
- (Don’t manually configure Hadoop)

EC2 Introductory Material

Overview: http://docs.aws.amazon.com/AWSEC2/latest/UserGuide/EC2_GetStarted.html

Pricing:

http://aws.amazon.com/ec2/pricing/

Map Reduce:

http://docs.aws.amazon.com/ElasticMapReduce/latest/DeveloperGuide/emr-get-started-count-words.html

Simple Queue Service:

http://docs.aws.amazon.com/AWSSimpleQueueService/latest/SQSGettingStartedGuide/Welcome.html

Free EC2 Resources (first year)

- 750 hrs of Linux Micro instance
- 750 hrs of Microsoft Server Micro instance
- 750 hrs+15GB Elastic Load Balancing
- 30 GB storage, 15GB outbound traffic
- 2 million IOs

Data Transfer in to EC2

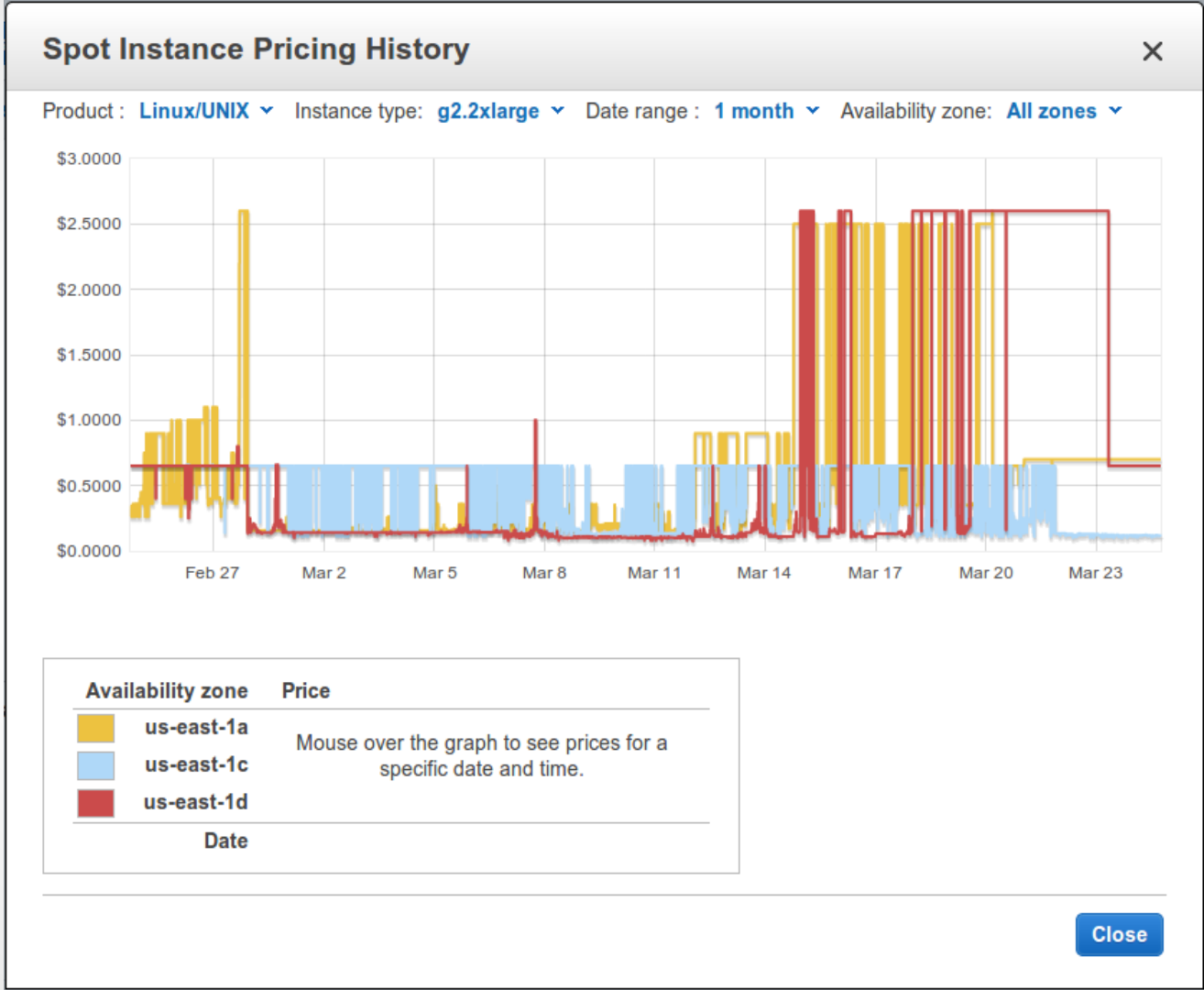
Billable EC2 Resources

CPU hours (rounded up to nearest hour)

Data Transfer out of EC2 (0-2 cents/GB)

0.4 cents per 10K IO requests

Reserved/ Spot Instances



Demo: Reserving EC2 Instance

Install Amazon Command Line Tools

Make ‘Administrators’ Security Group

(specify valid incoming addresses for SSH sessions)

IP masks for Princeton

Make Key Pair

https://console.aws.amazon.com/ec2/v2/home?region=us-east-1

Elastic Map Reduce Word Count

```
import sys
   import re

   def main(argv):
     line = sys.stdin.readline()
     pattern = re.compile("[a-zA-Z][a-zA-Z0-9]*")
     try:
       while line:
         for word in  pattern.findall(line):
           print  "LongValueSum:" + word.lower() + "\t" + "1"
         line =  sys.stdin.readline()
     except "end of file":
       return None
```

Demo: Elastic MapReduce

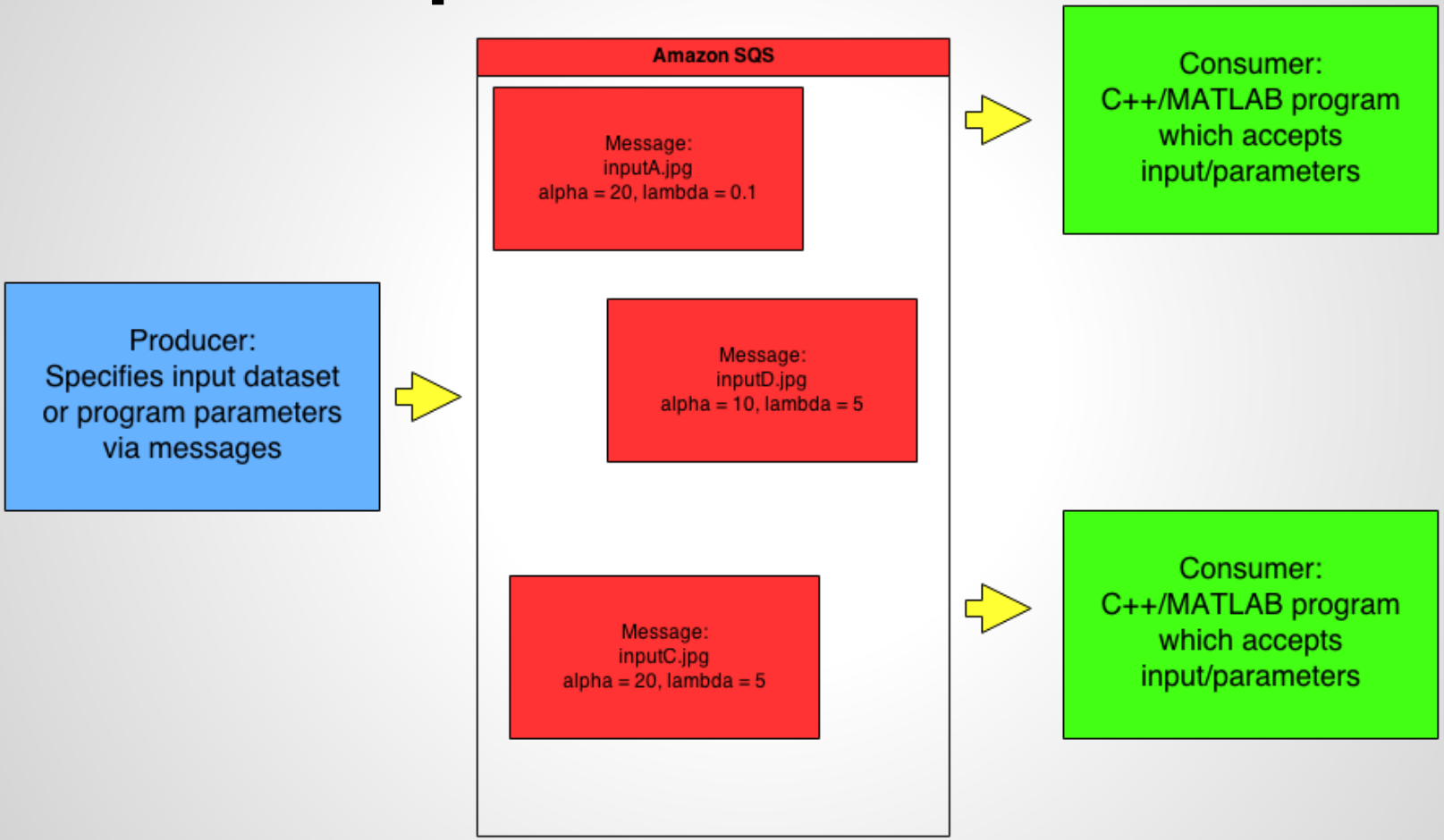
create storage location:

https://console.aws.amazon.com/s3/

run EMR:

https://console.aws.amazon.com/elasticmapreduce/vnext/home?region=us-east-1#

Amazon Simple Queue Service



Amazon SQS

main SQS console:

https://console.aws.amazon.com/sqs/home?region=us-east-1#

e.g. Python SDK for accessing queue:

http://boto.readthedocs.org/en/latest/ref/sqs.html

Additional Resources

Non-CS clusters at Princeton:
http://www.princeton.edu/researchcomputing/computational-hardware/

Hadoop Image Processing Interface:

http://hipi.cs.virginia.edu/

Matlab licensing on EC2:

http://www.mathworks.com/discovery/matlab-ec2.html