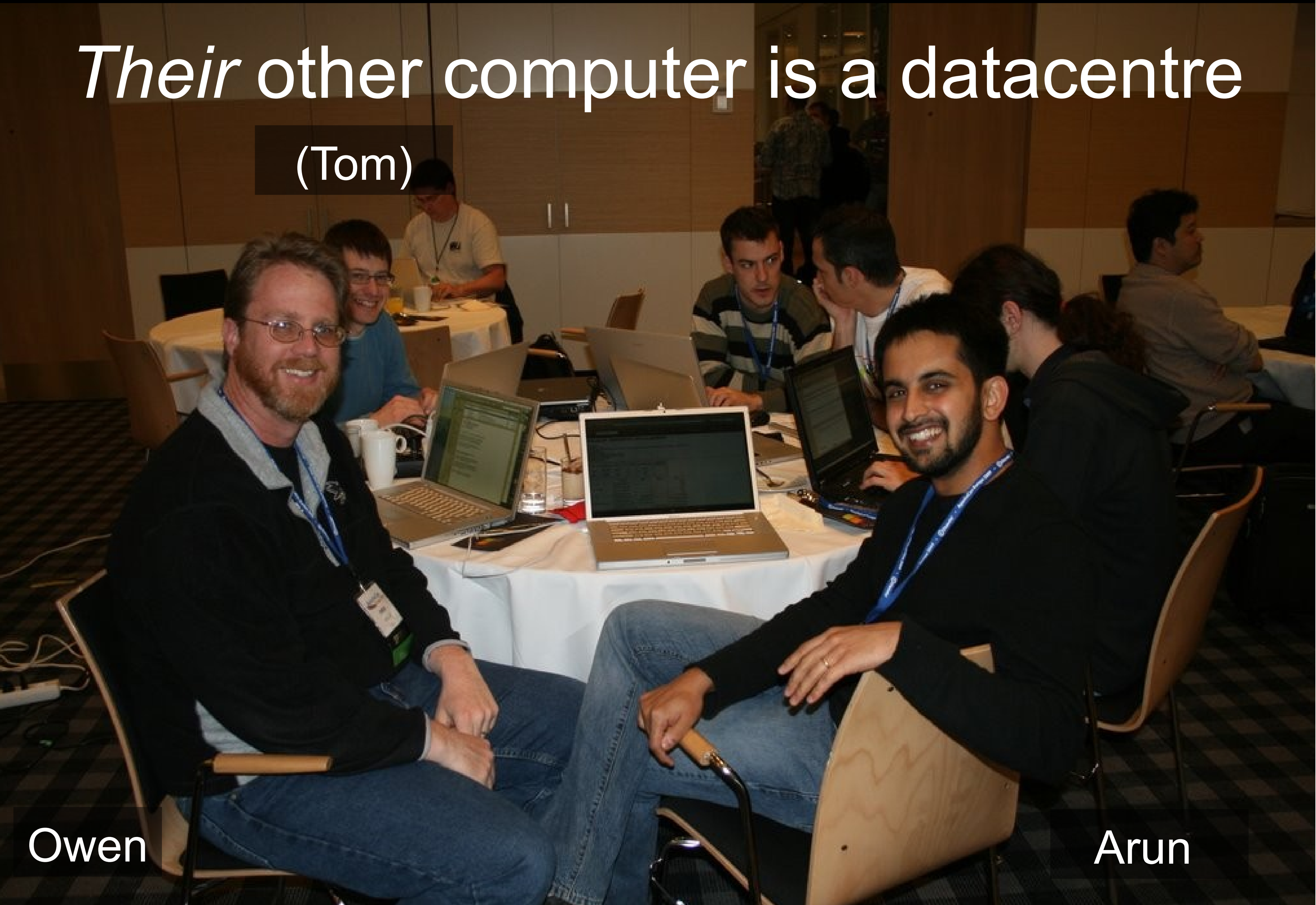
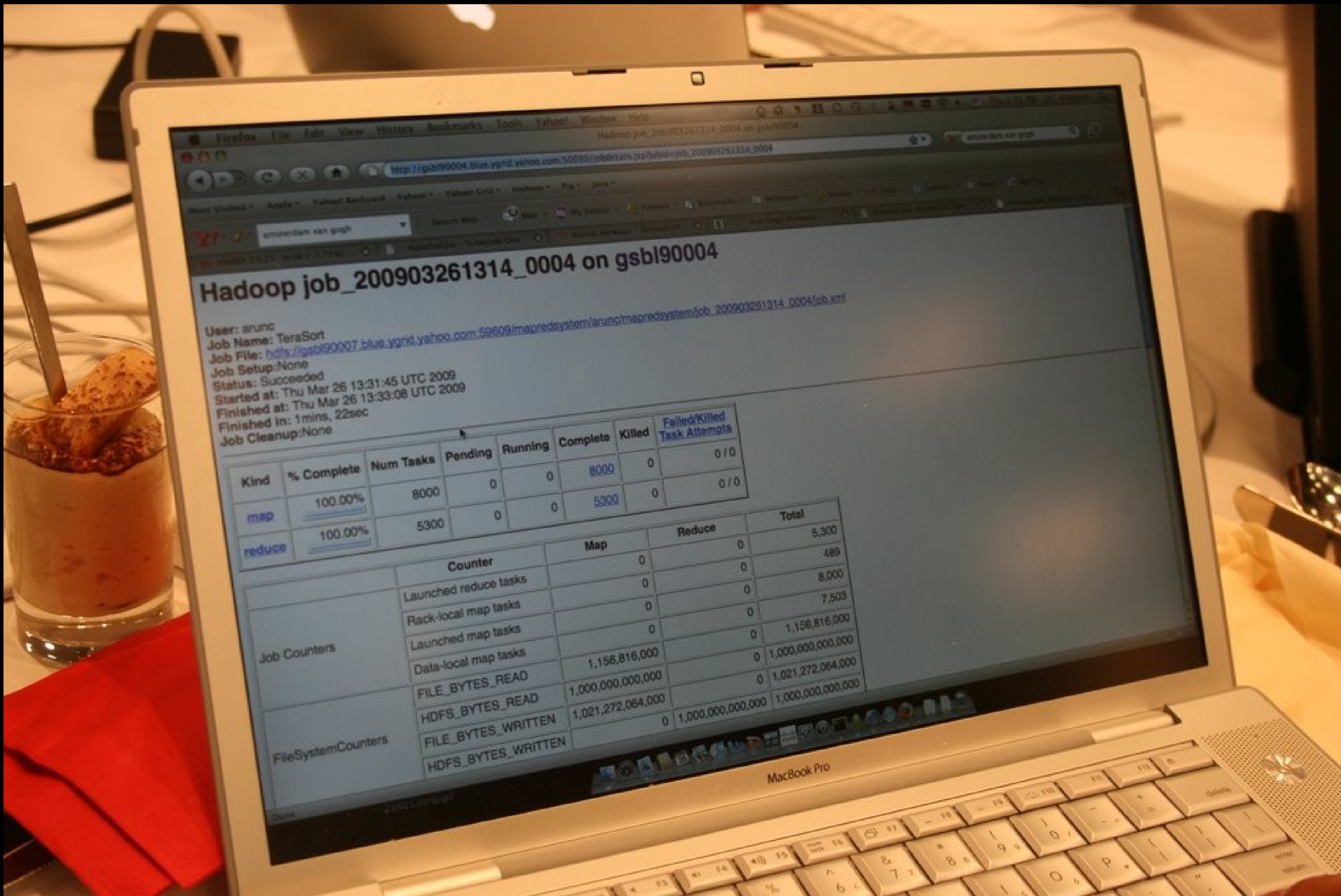

My other computer
is a data center


Julio
Steve
My other computer is a data center
Our other computer is a datacentre

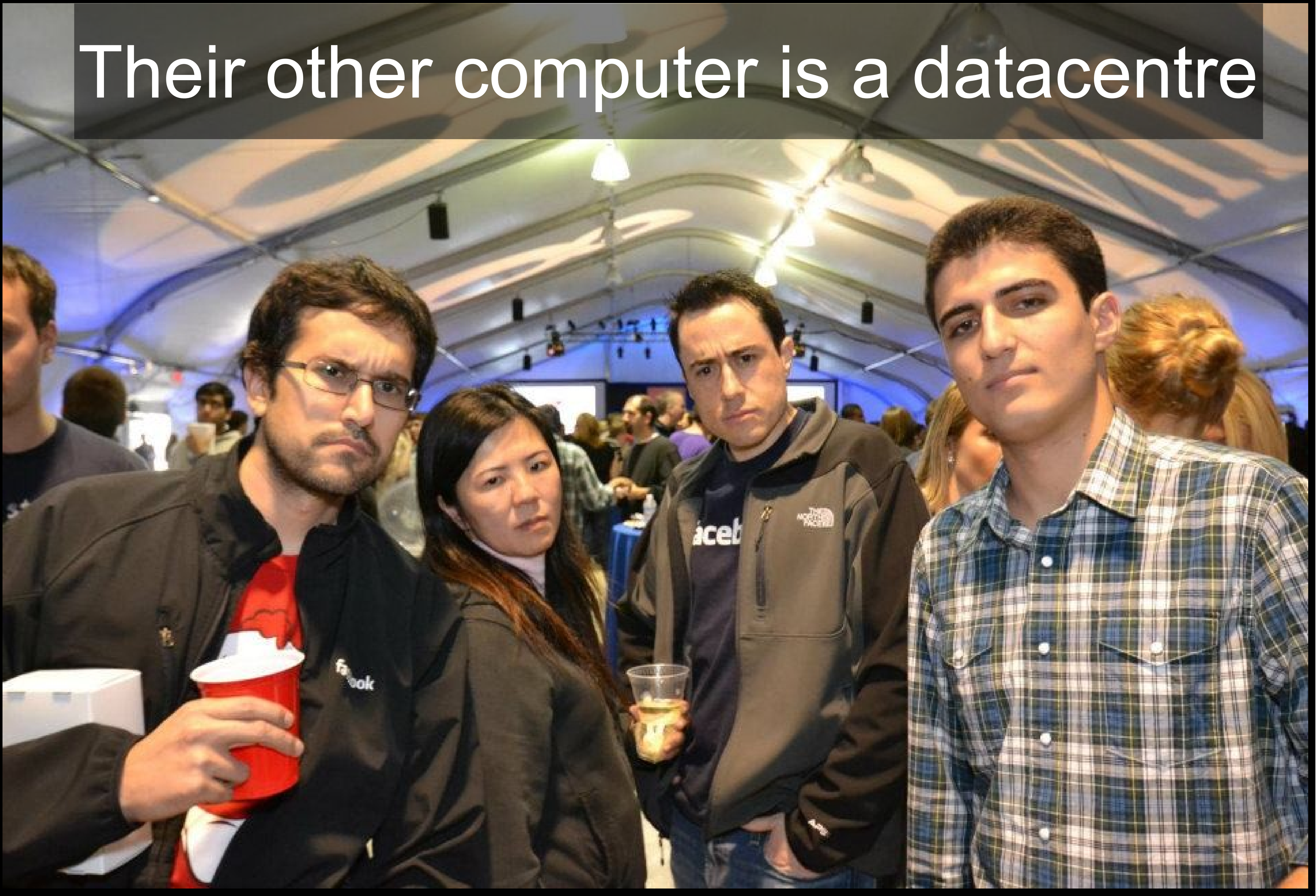
Their other computer is a datacentre



That sorts a Terabyte in 82s



Their other computer is a datacentre

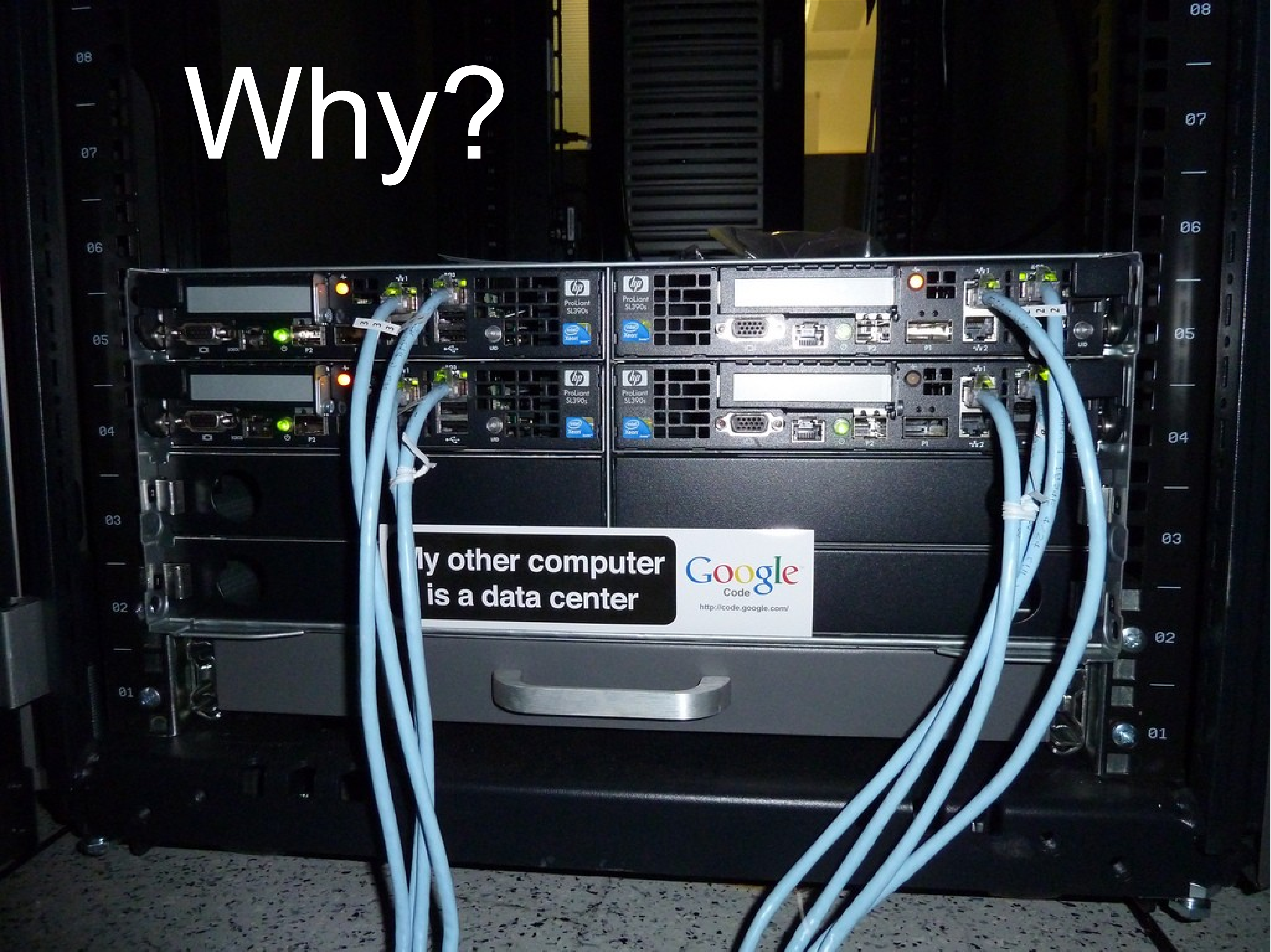


facebook: 45 Petabytes

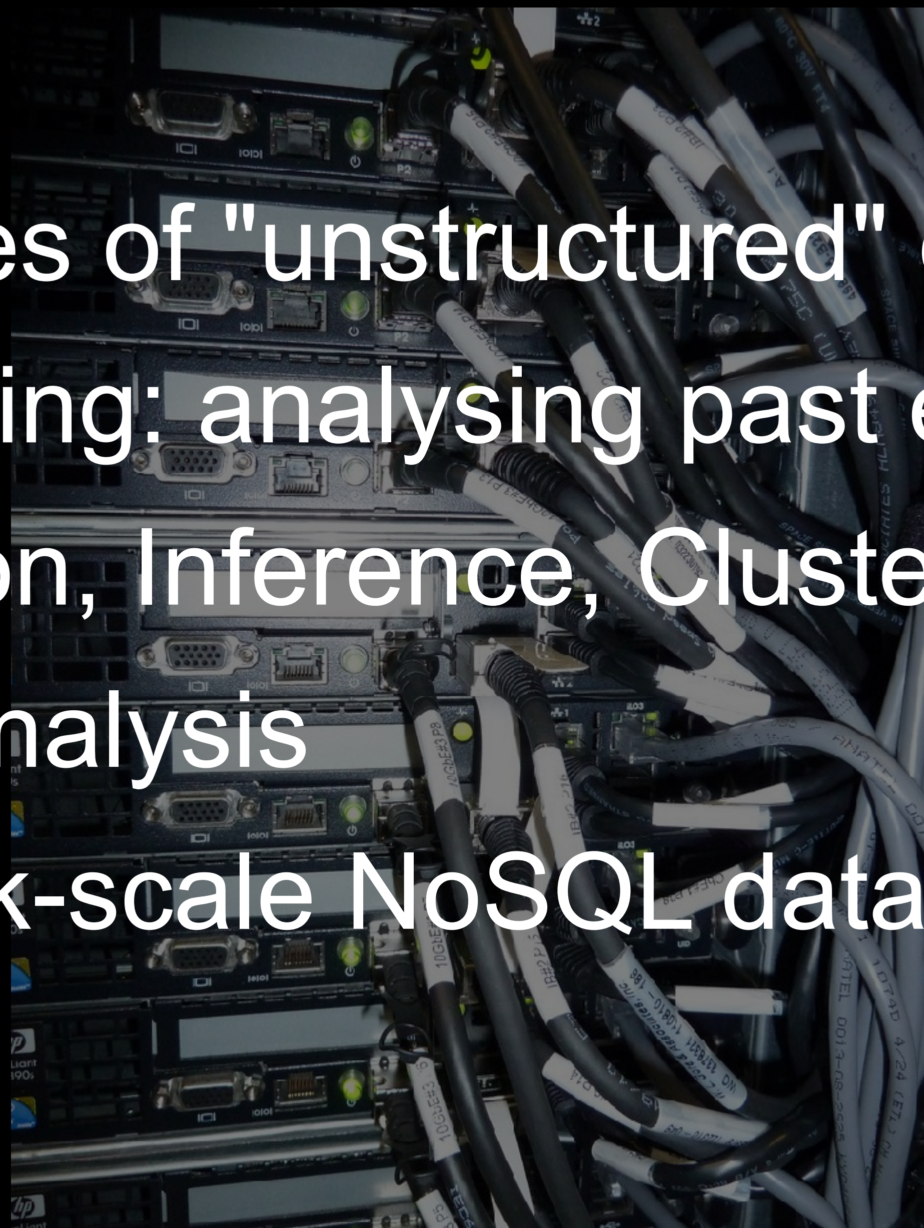
This is a datacentre

Yahoo! 8000 nodes, 32K cores, 16 Petabytes

Why?


other computer
is a data center
Google
Code
http://code.google.com/

Big Data



Petabytes of "unstructured" data

Datamining: analysing past events

Prediction, Inference, Clustering

Graph analysis

facebook-scale NoSQL databases

Big Data vs HPC

Big Data : Petabytes

- Storage of low-value data
- H/W failure common
- Code: frequency, graphs, machine-learning, rendering
- Ingress/egress problems
- Dense storage of data
- Mix CPU and data
- Spindle:core ratio

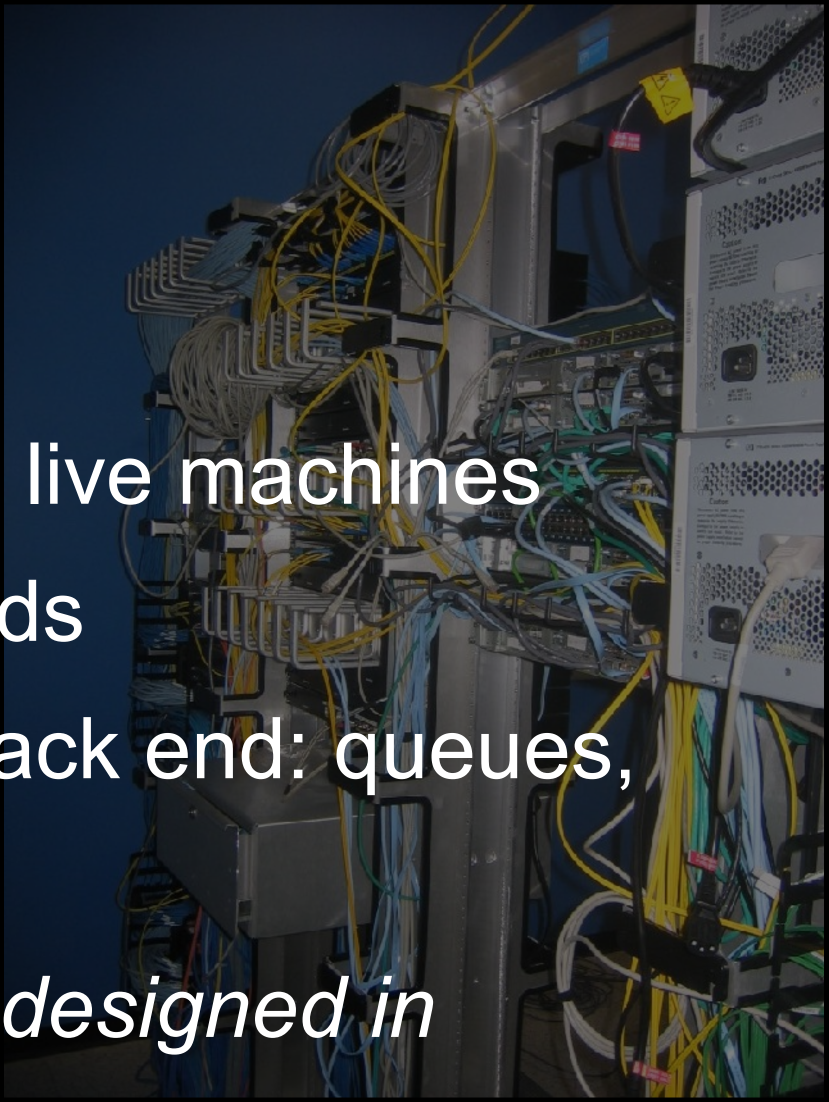
HPC: petaflops

- Storage for checkpointing
- Surprised by H/W failure
- Code: simulation, rendering
- Less persistent data, ingress & egress
- Dense compute
- CPU + GPU
- Bandwidth to other servers

Architectural Issues

- Failure is inevitable – design for it.
- Bandwidth is finite – be *topology aware.*
- SSD expensive, low seek times, best written to sequentially.
- HDD less $, more W; awful random access
  – stripe data across disks, read/write in bulk

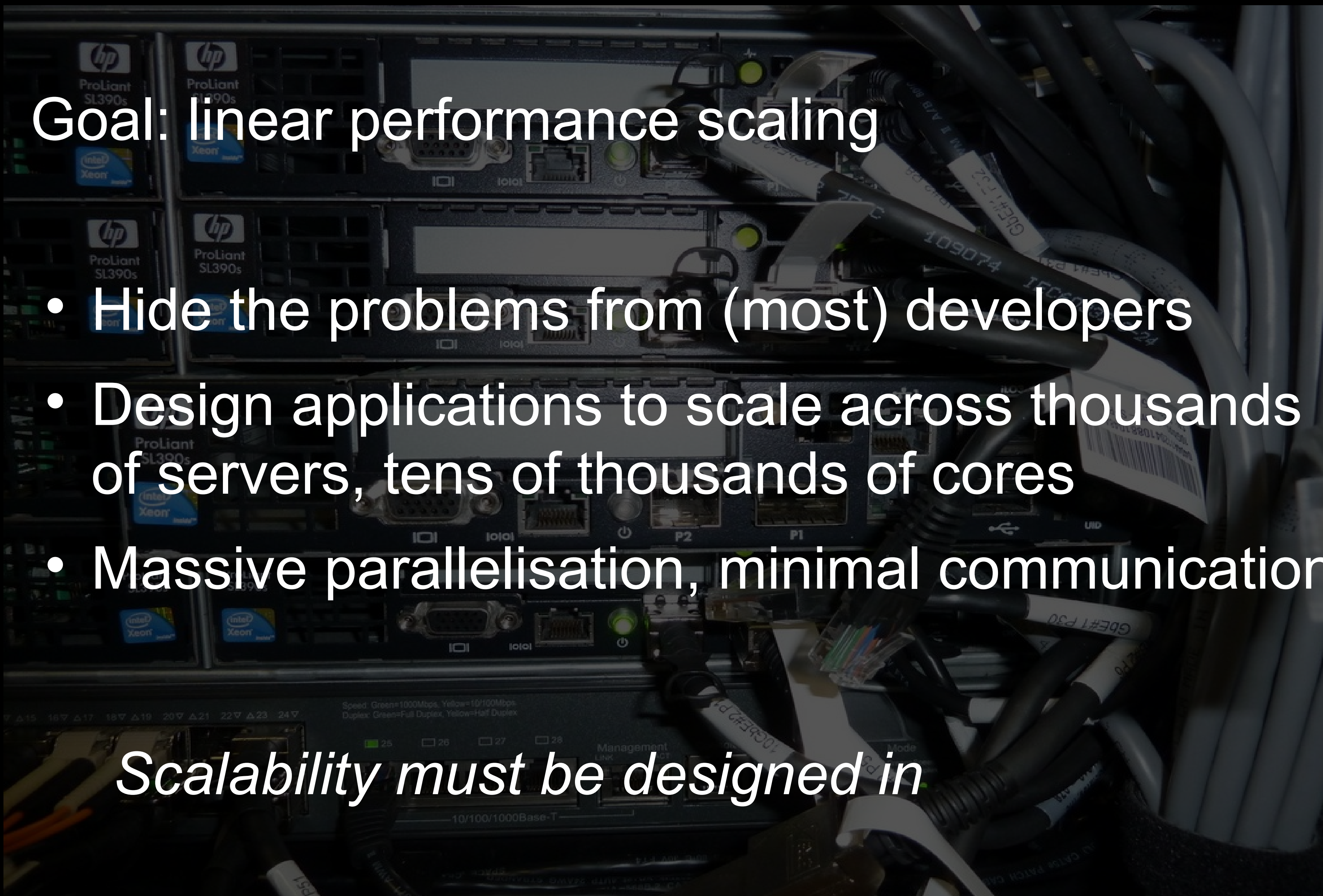
Coping with Failure



- Avoid SPOFs
- Replicate data
- Restart work
- Redeploy applications onto live machines
- Route traffic to live front ends
- Decoupled connection to back end: queues, scatter/gather

Failure-tolerance must be designed in

Scale



Goal: linear performance scaling

- Hide the problems from (most) developers
- Design applications to scale across thousands of servers, tens of thousands of cores
- Massive parallelisation, minimal communication

Scalability must be designed in

Algorithms and Frameworks

- MapReduce – Hadoop, CouchDB, (Dryad)
- BSP – Pregel, Giraph, Hama
- Column Tables – Cassandra, HBase, BigTable
- Location Aware filesystem: GFS, HDFS
- State Service: Chubby, Zookeeper, Anubis
- Scatter/gather – search engines
- (MPI)

MapReduce: Hadoop

Bath Bluetooth Dataset

```
gate1,b46cca4d3f5f313176e50a0e38e7fde3,,2006-10-30,16:06:17
gate1,f1191b79236083ce59981e049d863604,,2006-10-30,16:06:20
gate1,b45c7795f5be038dda8615ab44676872,,2006-10-30,16:06:21
gate1,02e73779c77fcd4e9f90a193c4f3e7ff,,2006-10-30,16:06:23
gate1,eef1836efddf8dbfe5e2a3cd5c13745f,,2006-10-30,16:06:24
```

- 2006-2009
- Multiple sites
- 10GB data

Map to device ID

```
class DeviceMapper extends Mapper {
  def parser = new EventParser()
  def one = new IntWritable(1)

  def map(LongWritable k, Text v,
        Mapper.Context ctx) {
    def event = parser.parse(v)
    ctx.write(event.device, one)
  }
}
```

(gate, device,day,hour) $\rightarrow$ (deviceID, 1)

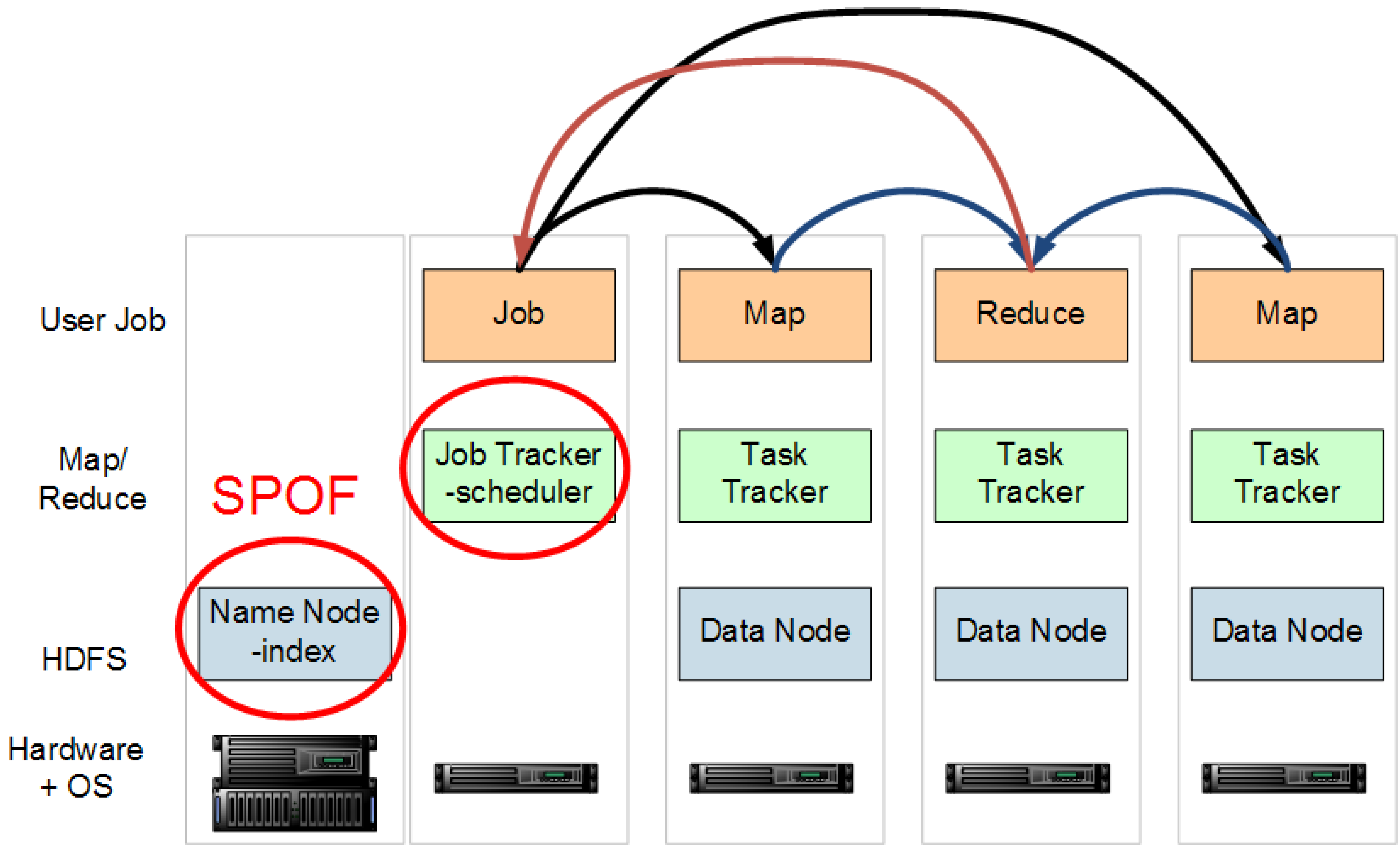
Reduce to device count

```
class CountReducer2 extends Reducer {
  def iw = new IntWritable()

  def reduce(Text k,
             Iterable values,
             Reducer.Context ctx) {
    def sum = values.collect() {it.get()}.sum()
    iw.set(sum)
    ctx.write(k, iw);
  }
}
```

(device,[count1, count2, ...]) → (deviceID, count')

Hadoop running MapReduce



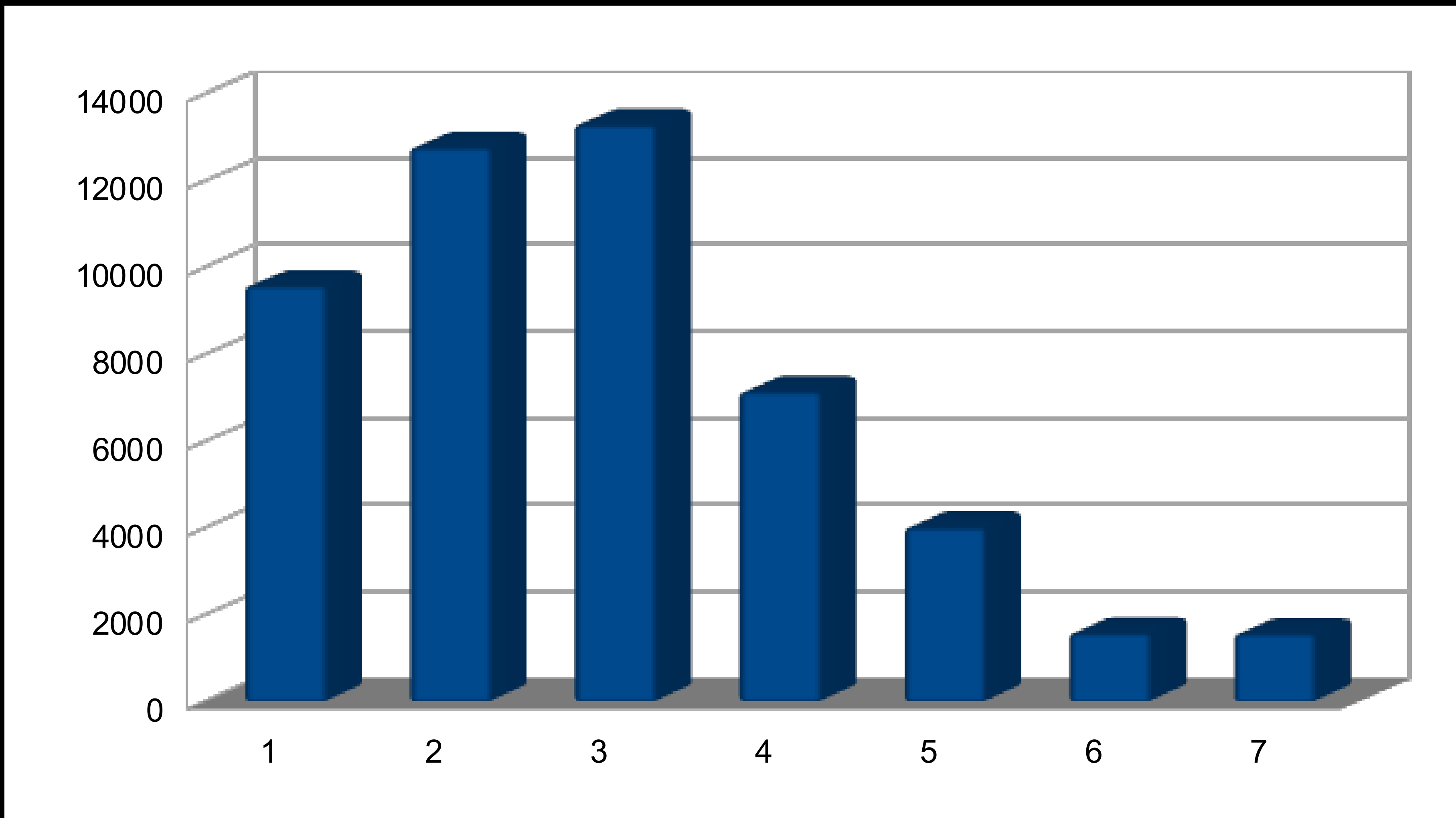
HDFS Filesystem

- Commodity disks
- Scatter data across machines
- Replicate data across racks
- Trickle-feed backup to other sites
- Archive unused files
- In idle time: check health of data, rebalance
- *Report data location for placement of work*

Results

```
0072adec0c1699c0af152c3cdb6c018e  2128
0120df42306097c70384501ebbdd888c  243
01541fef30e606ce88f8b0e931f010d2  5
0161257b1b0b8d1884975dd7b62f4387  15
01ad97908c53712e58894bc7009f5aa0  22
0225a2b080a4ac8f18344edd6108c46c  3
0276aba603a2aead55fe67bc48839cec  9
027973a027d85ad4dd4a15efa5142204  1
02e73779c77fcd4e9f90a193c4f3e7ff  3953
02e9a7bef5ba4c1caf5f35e8ada226ed  2
...
```

Map: day of week; reduce: count



Other Questions

- Peak hours for devices
- Predictability of device
- Time to cross gates/transit city
- Routes they take
- Which devices are often co-sighted?
- When do they stop being co-sighted?
- Clustering: resident, commuter, student, tourist

Hardware Challenges

- Energy Proportional Computing [Barroso07]
- Energy Proportional Datacenter Networks [Abts10]
- Dark Silicon and the End of Multicore Scaling [Esmaeilzadeh10]

Scaling of CPU, storage, network

CS Problems

- Scheduling
- Placement of data and work
- Graph Theory
- Machine Learning
- Heterogenous parallelism
- Algorithms that run on large, unreliable, clusters
- *Dealing with availability*

New problem: availability

- Availability in Globally Distributed Storage Systems [Ford10]
- Failure Trends in a Large Disk Drive Population [Pinheiro07]
- DRAM Errors in the Wild [Schroeder09]
- Characterizing Cloud Computing Hardware Reliability [Vishwanath10]
- Understanding Network Failures in Data Centers [Gill11]

What will your datacentre do?

Power Concerns

- PUE : 1.5-2X system power
  =server W saved has follow-on benefits
- Idle servers still consume 80% peak power
  =keep busy or shut down
- Gigabit copper networking costs power
- SSD front end machines save on disk costs, and can be started/stopped fast.

Where?

- Low cost (hydro) electricity
- Cool and dry outside air (for low PUE)
- Space for buildings
- Networking: fast, affordable, >1 supplier
- Low risk of earthquakes and other disasters
- Hardware: easy to get machines in fast
- Politics: tax, govt. stability/trust; data protection

Oregon and Washington States

Trend: containerized clusters

Logging & Health

- Everything will fail
- There is always an SPOF
- Feed the system logs into the data mining infrastructure: post-mortem and prediction

Monitor and mine the infrastructure
-with the infrastructure