A Very Brief Introduction To Cloud Computing

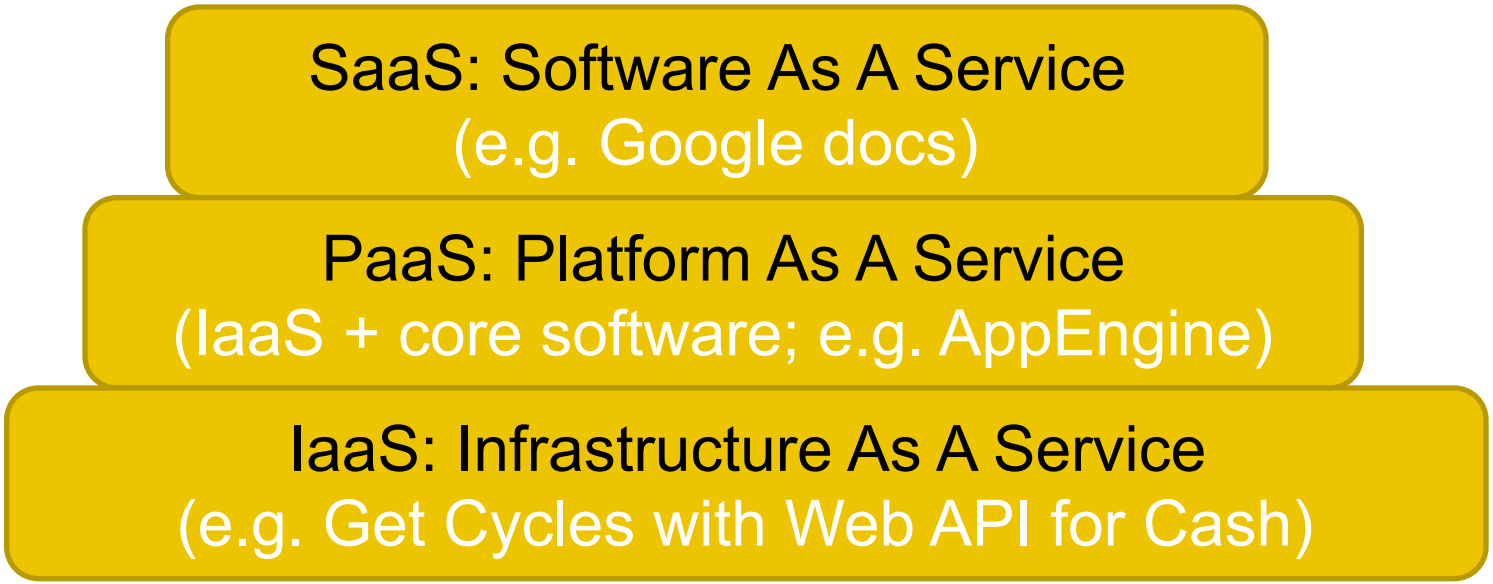
Jens Vöckler, Gideon Juve, Ewa Deelman,
G. Bruce Berriman

USC
INFORMATION
SCIENCES
INSTITUTE

ipac

What is “The Cloud”

Cloud computing refers to logical computational resources accessible via a computer network.



Clusters are still a crucial concept.

Examples for XaaS

- Software as a Service (Saas):
  - Google docs
  - MapReduce (Hadoop)
- Platform as a Service (PaaS):
  - Google App Engine
  - Microsoft Azure
- Infrastructure as a Service (IaaS):
  - Amazon Web Services (EC2)
  - FutureGrid

Of Grid, Clouds and HPC

Grids

- Pioneered the *service* model.
- Latency intrinsic to distributed systems.
- Less suitable for differential equations.
- Dependent on remotely installed software stack.

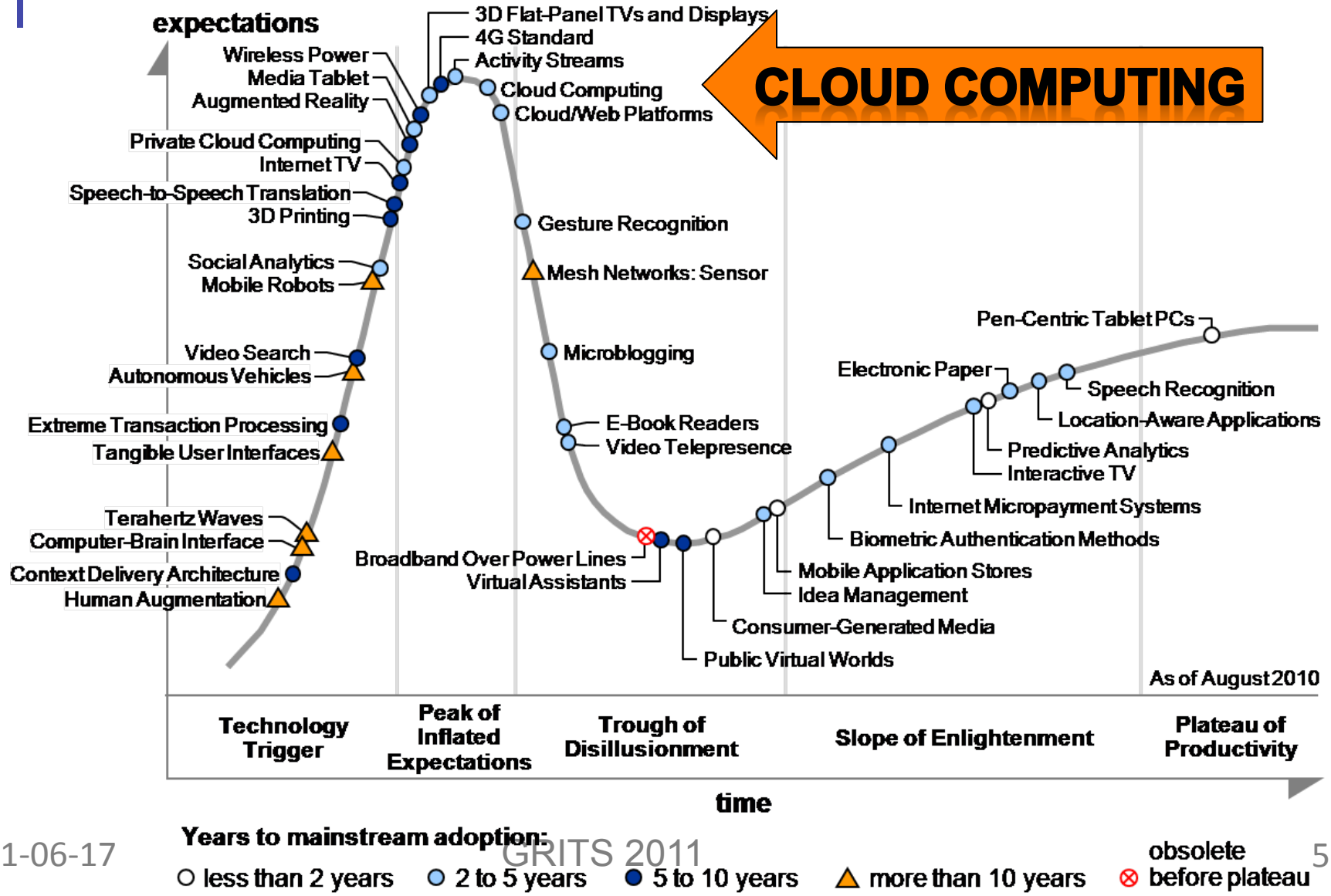
Clouds

- Elasticity on demand.
- Poor locality (affinity) affects sci. comp.
- Even less suitable for differential equations.
- Ship all your software with your image.
- MapReduce and GFS.

HPC

- Best suited for differential equations and tight MPI.

Is It A Fad?


expectations
3D Flat-Panel TVs and Displays
4G Standard
Activity Streams
Wireless Power
Media Tablet
Augmented Reality
Cloud Computing
Cloud/Web Platforms
CLOUD COMPUTING
Private Cloud Computing
Internet TV
Speech-to-Speech Translation
3D Printing
Gesture Recognition
Social Analytics
Mobile Robots
Mesh Networks: Sensor
Pen-Centric Tablet PCs
Video Search
Autonomous Vehicles
Microblogging
Electronic Paper
Speech Recognition
Location-Aware Applications
Extreme Transaction Processing
Tangible User Interfaces
E-Book Readers
Video Telepresence
Predictive Analytics
Interactive TV
Internet Micropayment Systems
Terahertz Waves
Computer-Brain Interface
Biometric Authentication Methods
Broadband Over Power Lines
Virtual Assistants
Mobile Application Stores
Idea Management
Context Delivery Architecture
Human Augmentation
Consumer-Generated Media
Public Virtual Worlds
As of August 2010
Technology Trigger
Peak of Inflated Expectations
Trough of Disillusionment
Slope of Enlightenment
Plateau of Productivity
time
Years to mainstream adoption:
less than 2 years
2 to 5 years
5 to 10 years
more than 10 years
obsolete before plateau

When To Use Clouds

- Individual projects with bursty needs.
- High-Throughput Applications with modest data needs.
- Infrastructure-challenged sites.
- Undetermined or volatile needs.

(see also: “Other Considerations” later)

2010 Top Cloud Providers

1. Amazon Web Services (IaaS)
2. Rackspace (IaaS)
3. Salesforce.com (SaaS or PaaS)
4. Google App Engine (PaaS)
5. Microsoft Azure (PaaS)
6. Joyent (IaaS)
7. GoGrid (IaaS)

According to searchcloudcomputing.techtarget.com

Software Offers

Amazon

- Linux + Windows VMs
- Variety of storage, network, compute svcs
- Two dozen additional services, i.e.
  - Some business oriented like payment, CDN, DNS
  - Some useful for science like RDS, messaging, queues, MR, EBS
  - Mechanical Turk

Rackspace

- Linux + Windows VMs
- Variety of storage, network, compute svcs
- Smaller set of additional services, i.e.
  - Object storage
  - Load balancing
  - Elastic web hosting (like Google App Engine)

Support

Amazon

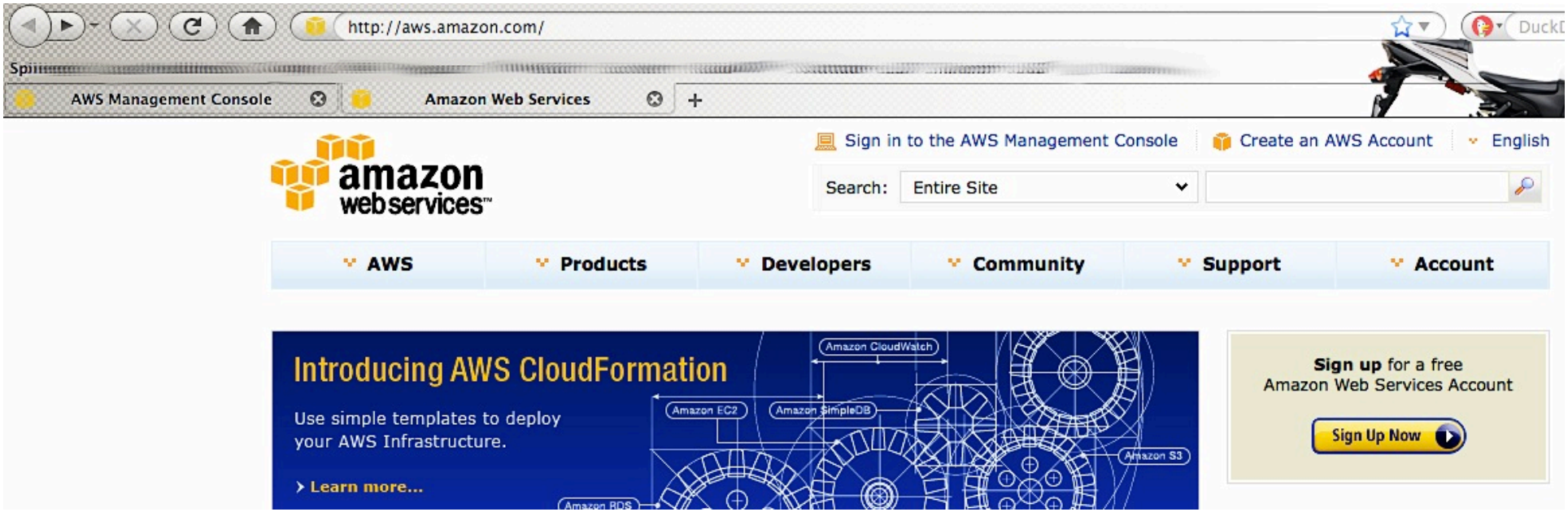
- Basic technical support in forums.
- Slow response time to forum questions.
- Or paid support
  - Lowest level starts at $50 / month for 12 hour response time

Rackspace

- 24/7/365 support
- Phone, email, ticket
- Included in service charges
- Industry leader in cloud support

How Do I Get Started

1) You need a web browser
2) And a credit card



How Much Does It Cost 1/3

1. Computing – per **hour**
   - Region: us east, us west, eu, …
   - Service: on-demand, spot, reserved
   - OS: Linux, Windows
   - Node capacity: m1.large, …

US-east on-demand Linux m1.large

- $0.34 / h

How Much Does It Cost 2/3

2 Networking – per GiB **and** month

→ Data going **in**: $0.10 / GiB

← Data coming **out**:

- 0 .. 1 GiB: $0
- < 10 **T**B: $0.15 / GiB ← max: use for estimates
- 11.. 49 TB: $0.11 / GiB
- 50 .. 149 TB: $0.09 / GiB
- > 150 TB: $0.08 / GiB

How Much Does It Cost 3/3

3 Storage – per GiB **and** month

- EBS: $0.10 / GiB * month
- S3: $0.15 / GiB * month

More Grids and Clouds

	Grids	Clouds
Problems	Too few cycles, too long queues	Too many users, too much data
Infrastructure	Clusters, supercomputers	Clusters, data centers
Architecture	Federated virtual organization	Hosted organization
Programming Model	MPI: Powerful but difficult	MapReduce: Less powerful yet simple

Other Considerations 1/4

- Security
  - Is your data yours? Safe in transit? OK that is “shares” space with strangers?
- Vendor lock-in
  - No standards (yet)
  - Using vendor services ➡ dependency
- Efficacy
  - Benchmark machine types to find **cost-performance** optimum for *your* application.

Other Considerations 2/4

- Use caching (reduce transfer-$$)
  - Transfer data once and store it for a month.
  - Reuse during the month many times.
- Consider your time-line
  - Clouds are good for short-term needs
  - Or highly bursty cycle requirements
  - Long-term better invest in your own HW
- Deploying distributed applications
  - RightScale, Chef, Puppet (,Wrangler)

Other Considerations 3/4

- System administration
  - Clouds: Onus is on **you** to get it right
    - How well do *you* know Linux sys admin tasks?
    - Or will you have to pay someone?
  - HPC/Grids: Remote admin responsibility
- Overhead
  - *Virtualization* slower than *bare metal*
  - Commodity Gig-E versus Myrinet et. al.
  - Amazon CC solves some of it, but $$$

Other Considerations 4/4

- Application size
  - Good fit: 1,000…10,000 CPU hours
  - >10k CPU hours: Put costs into budget
    - Maybe HPC elsewhere a better fit?
- No queue
  - Cloud is a finite resource
  - No queuing, just error “no capacity”
    - Happy retrying…
  - HPC can achieve 90% resource utilization

Cloud Comparison

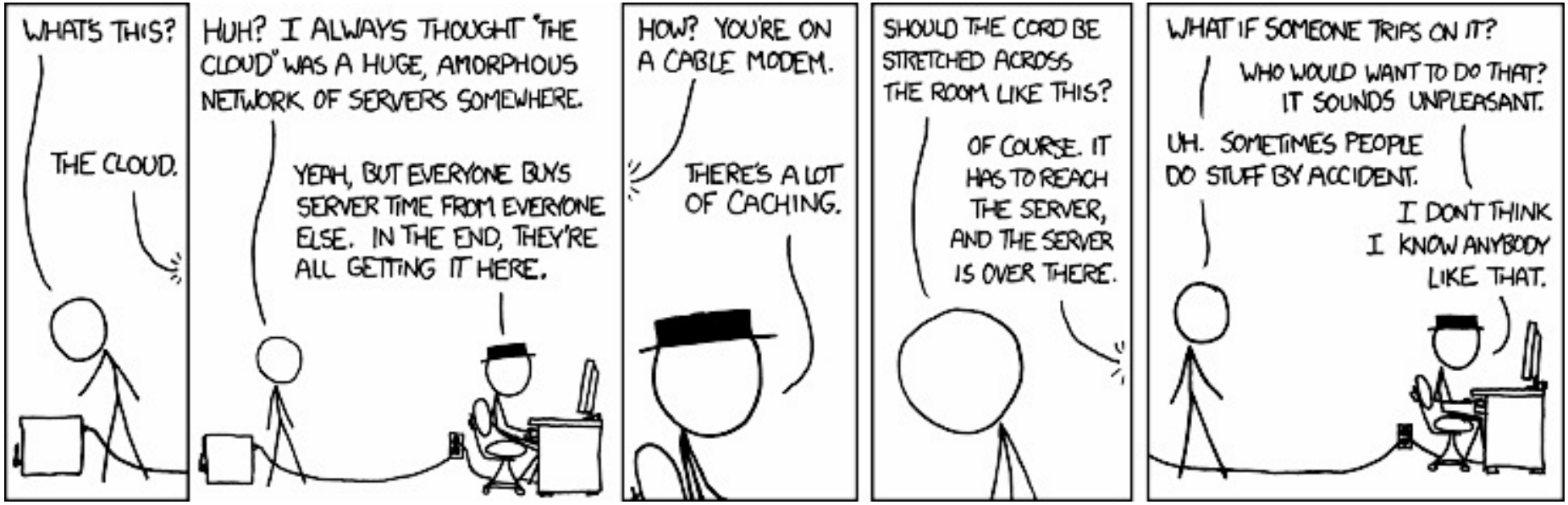
- Compare academic and commercial clouds
  - NERSC's *Magellan* cloud (Eucalyptus)
  - Amazon's cloud (EC2), and
  - FutureGrid's *sierra* cloud (Eucalyptus)
- 16k Kepler Periodograms with *plavchan*
- Constrained node- and core selection
  - Because AWS costs $$
  - 6 nodes, 8 cores each node

Cloud Comparison II

Site	CPU	RAM (SW)	Walltime	Cum. Dur.	Speed-Up
Magellan	8 x 2.6 GHz	19 (0) GB	5.2 h	226.6 h	43.6
Amazon	8 x 2.3 GHz	7 (0) GB	7.2 h	295.8 h	41.1
FutureGrid	8 x 2.5 GHz	29 (½) GB	5.7 h	248.0 h	43.5

- Given 48 physical cores
  - Speed-up ≈ 43 considered *pretty good*
- AWS cost ≈ $31
  - 7.2 h x 6 x c1.large ≈ $29
  - 1.8 GB in + 9.9 GB out ≈ $2

http://xkcd.com/908/



(There's planned downtime every night when we turn on the Roomba and it runs over the cord.)

Acknowledgements

- Gideon Juve,
- Bruce Berriman,
- Ewa Deelman,
- Shane Canon (NERSC)

INFORMATION SCIENCES INSTITUTE

agent of innovation

USC Viterbi School of Engineering

http://pegasus.isi.edu/

ipac

Common Misconceptions

Shane Canon, “Debunking Some Common Misconceptions…” Talk during ScienceCloud HPDC 2011

- “Clouds are simple to use and don’t require system administrators.”
  - IaaS VMs must be customized for app.
  - No batch system. No global file system.
  - Users must secure images, resources.
  - Require significant amount of System Administrative experience.

Common Misconceptions

- "My job will run immediately."
  - The cloud is, in the end, a finite resource.
  - *Scientific* applications will always need more resources.
- "Clouds are more efficient."
  - Virtualization less efficient than bare metal.
    - Amazon EC2 is 2…50 times slower on variety of benchmarks run by NERSC.
    - Though Amazon CC looks reasonable.

Common Misconceptions

- "Clouds allow you to ride Moore's Law without additional investment."
  - EC2 price has dropped 18% in 5 years.
  - But cores per socket (at same $$) have increased two- to five-fold in 5 years.

Cloud Computing is a Business Model.

A Word About MapReduce

- (Patented) framework
  - Based on functional programming
  - Introduced by Google in 2004
  - Key-value pairs and lists as key data structures
  - Very useful for language applications
  - Scalability and Fault-tolerance “built-in”

“If you only have a hammer, the world consists of nails”

A Word About MapReduce

- Fault Tolerance and Scalability, because *map* and *reduce* are
  - Typically small,
  - Stateless (functional paradigm),
  - Perform I/O with disk,
    - Replicated blocks file system
  - Run in parallel, and
  - Create independent results.
- *Reduce* can only start after all *Map* have finished.

MapReduce

- You can unroll MapReduce for HPC