The Clouds of Revolution

April 1, 2010

Peter Van Roy

ICTEAM Institute
Université catholique de Louvain
Louvain-la-Neuve, Belgium

© 2010 Peter Van Roy

The Clouds of Revolution

April 1, 2010

Peter Van Roy

ICTEAM Institute
Université catholique de Louvain
Louvain-la-Neuve, Belgium

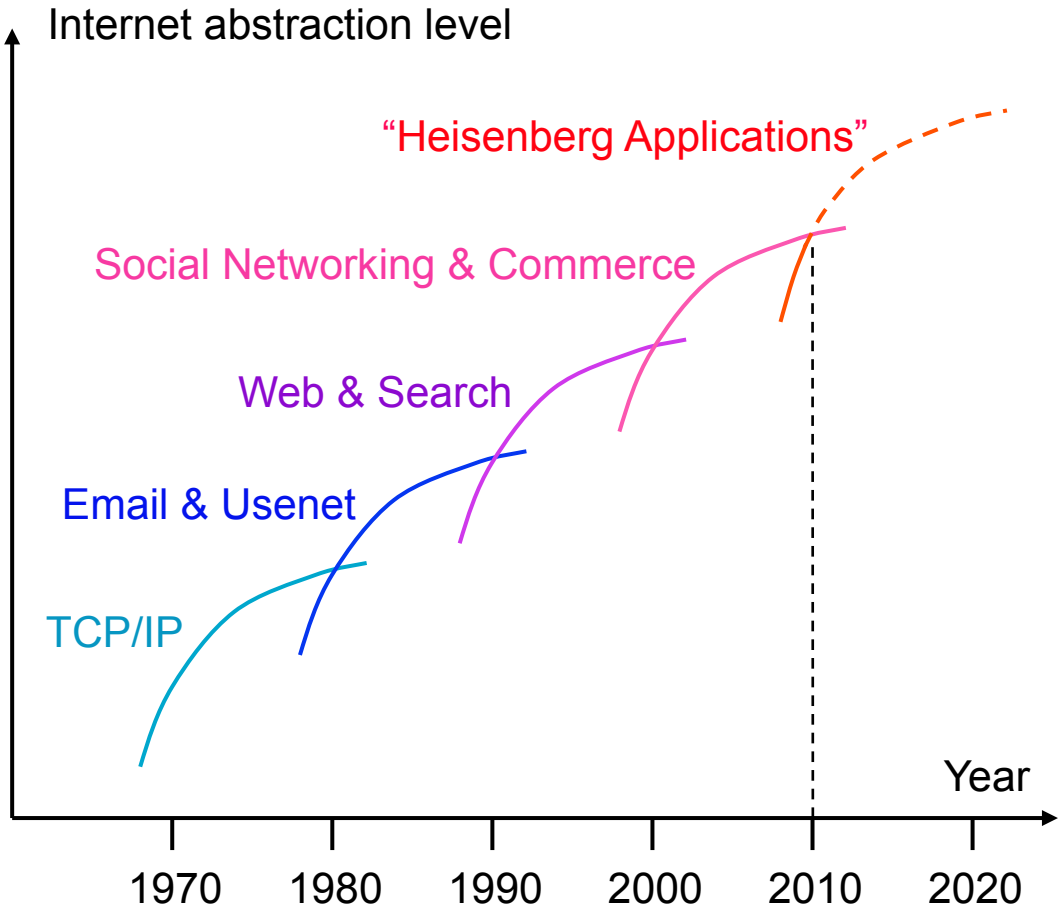
© 2010 Peter Van Roy

Overview

- The combination of cloud computing and data-intensive algorithms is revolutionizing Internet applications
  - Google and Amazon started the revolution almost by accident because of their internal needs, but now it has its own momentum
- Clouds make possible the “Heisenberg Application”, a new kind of data-intensive application that works in bursts
  - We conjecture that Heisenberg applications will cause a qualitative jump in the functionality of Internet operations
- Rest of the talk
  - History of Internet revolutions
  - Cloud computing
  - Data-intensive algorithms
  - Heisenberg applications

© 2010 Peter Van Roy

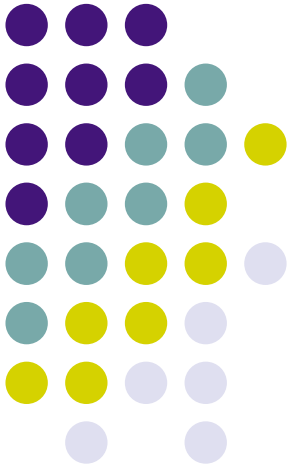
History of Internet Revolutions



- The Internet has gone through four revolutions since its inception
  - About one every ten years; each revolution takes about ten years to be internalized
- We are now on the brink of a fifth revolution
- It is based on a combination of cloud computing and data-intensive algorithms that we call Heisenberg applications

© 2010 Peter Van Roy

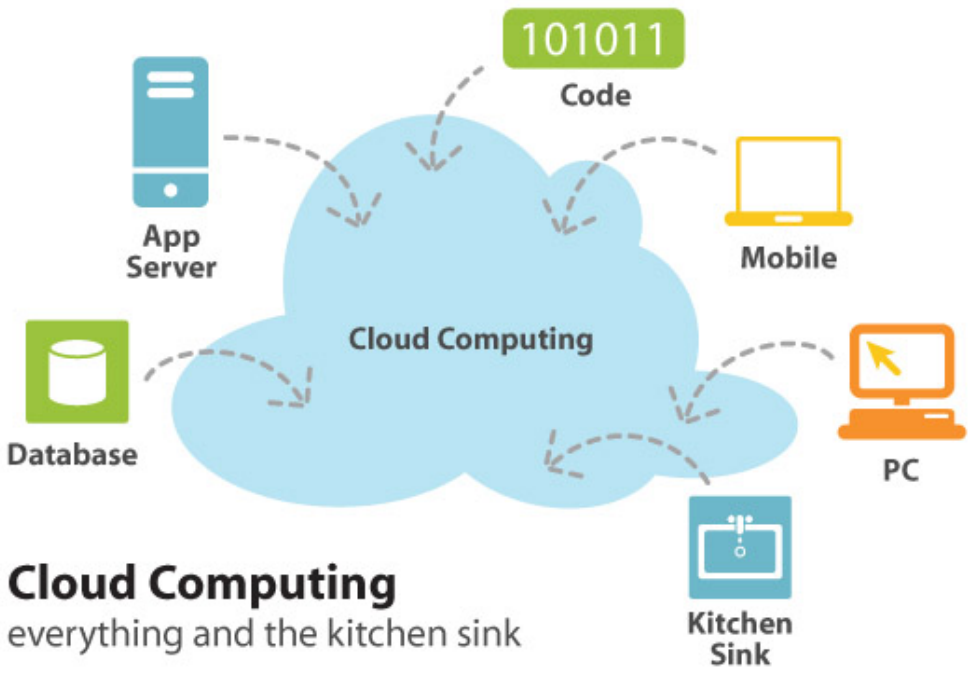
Cloud Computing



© 2010 Peter Van Roy

Cloud Computing

- Cloud computing is a hot topic in computing today
- It allows enterprises to offload their computing infrastructure
  - More economical and scalable
- It gives mobile devices an easy way to manage data
  - Data always available to mobile interactive applications



- Is that all that cloud computing offers?
  - No, this is just the tip of the iceberg!
  - Cloud computing is the vanguard of a much more profound revolution

© 2010 Peter Van Roy

What is Cloud Computing? (1)

- "Cloud computing" has become a buzzword of praise and scorn
  - "*It's stupidity. It's worse than stupidity: it's a marketing hype campaign. Somebody is saying this is inevitable–and whenever you hear somebody saying that, it's very likely to be a set of businesses campaigning to make it true*."
    – Richard Stallman, free software guru, in *The Guardian*, Sep. 29, 2008
- But what does it mean exactly?
  - Let us introduce it by giving some expert definitions

- The European Union definition: sense and simplicity (we hope!)
- A 'cloud' is an *elastic* execution environment of resources involving *multiple stakeholders* and providing a *metered service* at multiple granularities for a specified level of quality (of service).
  — From *The Future of Cloud Computing: Opportunities for European Cloud Computing Beyond 2010*, EU Expert Group Report Version 1.0, Jan. 2010.

© 2010 Peter Van Roy

What is Cloud Computing? (2)

- The Berkeley definition: the systems view
- Cloud Computing refers to both the applications delivered as services over the Internet and the hardware and systems software in the datacenters that provide those services. The services themselves have long been referred to as Software as a Service (SaaS), so we use that term. *The datacenter hardware and software is what we will call a Cloud*.
  - From *Above the Clouds: A Berkeley View of Cloud Computing*, UC Berkeley, Technical Report UCB/EECS-2009-08, Feb. 10, 2009

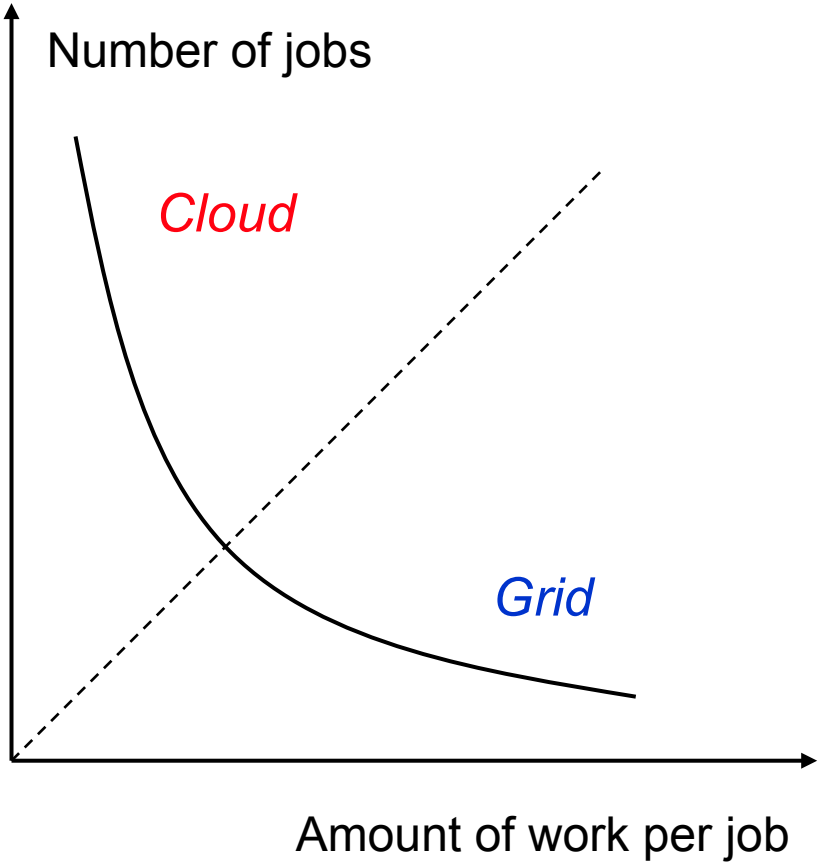
© 2010 Peter Van Roy

What is Cloud Computing? (3)

- The ars technica definition: the expert user view
- Cloud computing is an approach to client-server in which the "server" is a *dynamically scalable network* of loosely coupled heterogeneous nodes that are *owned by a single institution* and that tends to be biased toward *storage-intensive workloads*, and the "clients" are a wide variety of individuals and institutions that use fractions of shared nodes to run jobs that are transient with respect to time, lightweight with respect to compute-intensity, and anywhere from lightweight to heavy with respect to storage-intensity.
  - From *The Cloud: A Short Introduction*, Jon Stokes, Ars Technica, arstechnica.com, Nov. 8, 2009

© 2010 Peter Van Roy

Cloud versus Grid



- Clouds and grids are often confused, but there are (or were?) some clear differences
- Both provide utility computing with a dynamic collection of heterogeneous loosely coupled nodes
- Clouds tend to have many small jobs that are storage intensive ("business computation driven")
- Grids tend to have a few big jobs that are computation intensive ("scientific computation driven")

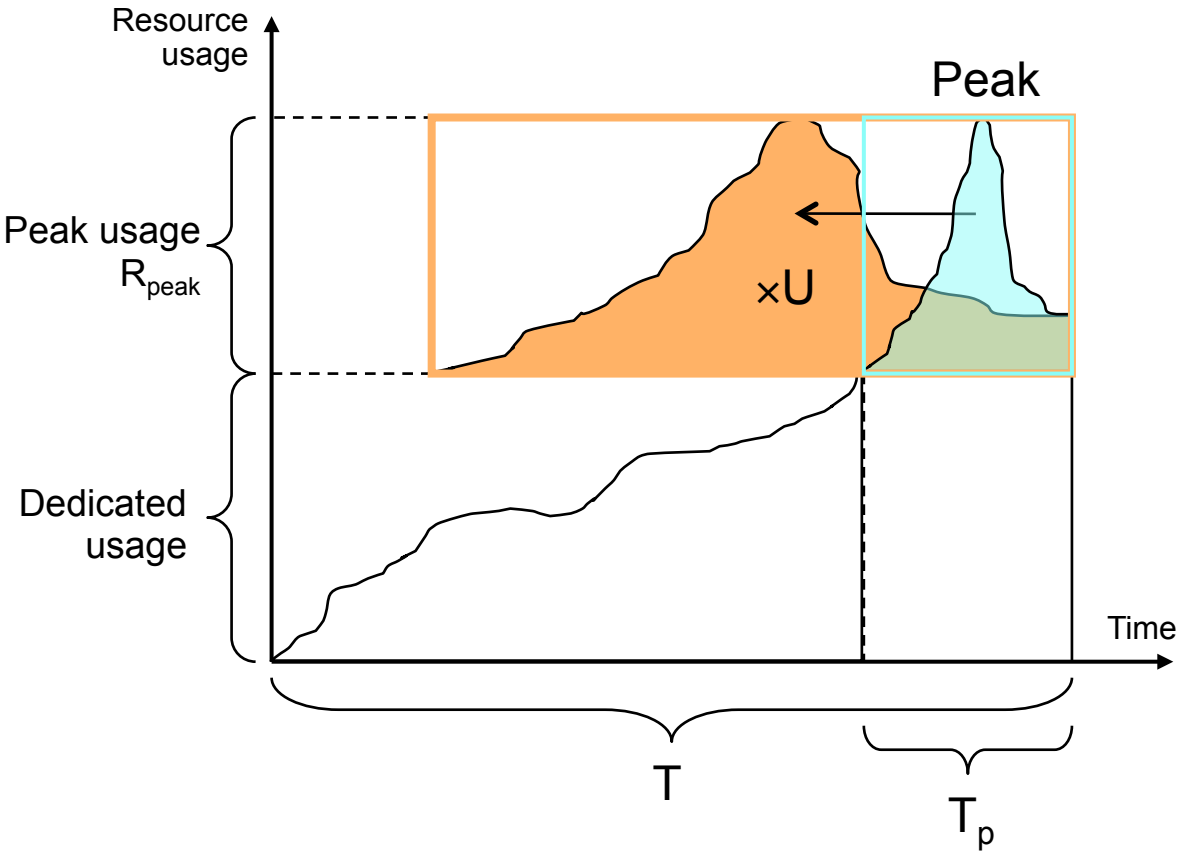
© 2010 Peter Van Roy

Economics of Cloud Computing

- Who provides your computing infrastructure?
  - How to decide for in-house solution, cloud solution, or a hybrid?
- Pure cloud with "pay per use" is best if unit cost of cloud services is lower than dedicated capacity
- Pure cloud can also be useful even if unit cost is higher
  - If the peak-to-average ratio is higher than the cost difference
  - Because the dedicated capacity must be built to peak and the cloud is paid according to average
- Hybrid solution (partly cloud, partly dedicated) makes sense if peaks are of "short enough" duration
  - If utility premium $U>1$ and peak duration $T_p<T/U$ then a hybrid solution costs less than a dedicated solution

© 2010 Peter Van Roy

Economics of Hybrid Solution



Question: do we implement the peak locally or in the cloud?

- Local cost = $R_{peak} \cdot T$
- Cloud cost = $U \cdot R_{peak} \cdot T_p$

Therefore the cloud is cheaper if $U \cdot T_p < T$

Note: actual cloud cost will be less than $U \cdot R_{peak} \cdot T_p$, since it is proportional to the actual use (solid blue shape, not blue rectangle)

© 2010 Peter Van Roy

Some Commercial Clouds

- Amazon Web Services (IaaS, since 2006)
  - Includes Simple Storage Service (S3), Elastic Computing Cloud (EC2), Elastic Block Store (EBS)
  - Comes with many secondary tools: e-commerce, CDN (Content-Distribution Network), Mechanical Turk, etc.
- Windows Azure (IaaS)
  - Hosting service for .NET applications and SQL databases
- Rackspace, Joyent (IaaS)
- Google AppEngine (PaaS)
  - Automatic scaling and reliability at the price of a highly constrained application structure (3-tier Web application)

© 2010 Peter Van Roy

Some Cloud Stores

- Amazon Simple Storage Service (S3) (2006)
  - Two-level hierarchy: buckets (like folders) storing data objects (1B to 5GB with metadata)
- EMC Atmos (Fall 2008)
  - Highly scalable data objects with metadata
- Key-Value stores (see the NoSQL controversy)
  - Highly scalable with some database functionality
  - Dynamo, Memcached
  - Scalaris (distributed transactions using Paxos, strong consistency)
- Cloud databases
  - Google BigTable: hybrid between row and column database, distributed transactions using Paxos, eventually consistent
  - Apache HBase, provides BigTable-like abilities for Hadoop
  - Apache Cassandra, combines Dynamo with BigTable, both eventual and strong consistency

© 2010 Peter Van Roy

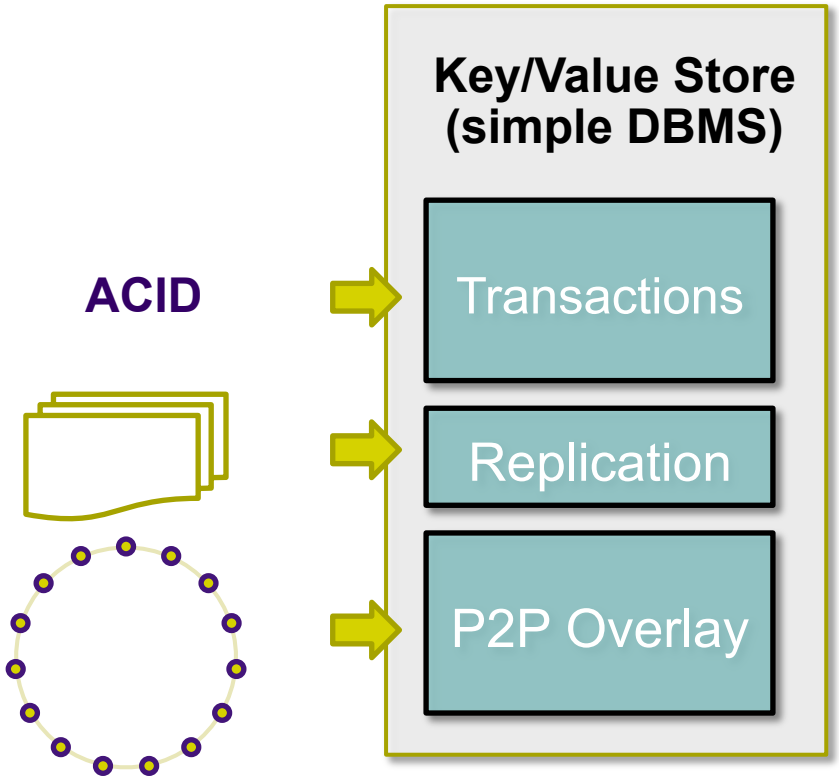
The NoSQL Controversy

- NoSQL is a current trend in non-relational databases
  - May lack table schemas, may lack ACID properties, no join operations
  - Main advantages are excellent performance, with good horizontal scalability and elasticity (ideal fit to clouds)
    - SQL databases have good vertical scalability but are not elastic
- Often only weak consistency guarantees, such as eventual consistency (e.g., Google BigTable)
  - Some exceptions: Cassandra also provides strong consistency, Scalaris and Beernet provide a key-value store with transactions and strong consistency

© 2010 Peter Van Roy

Scalaris and Beernet

- Scalaris and Beernet are key-value stores developed in the SELFMAN project (www.ist-selfman.org) (*)
  - They provide transactions and strong consistency on top of loosely coupled peers using the Paxos uniform consensus algorithm for atomic commit
  - They are scalable to hundreds of nodes; with ten nodes they have similar performance as MySQL servers
  - Scalaris won first prize in the IEEE Scalable Computing Challenge 2008
- They are an example of a scalable application infrastructure



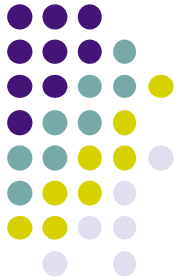
(*) SELFMAN was coordinated by P. Van Roy

© 2010 Peter Van Roy

Data-intensive Algorithms

© 2010 Peter Van Roy

Data-intensive Algorithms



- Computing science is changing fundamentally
- It is becoming focused on programming with large data sets
  - Data-intensive algorithms running on large-scale distributed systems are realizing one by one the old dreams of artificial intelligence
  - The canonical example is Google Search using PageRank
    - It extracts useful information from the Web link graph
  - Many other applications are now following this path: data mining (e.g., recommendation systems), machine learning, statistical language translation, image recognition, visualization, complex problem solving, etc.
- This is where most of the innovation will happen in Internet applications in the next decade

© 2010 Peter Van Roy

Google Search and the PageRank Algorithm

- Each Web page holds a quantity of stuff called its "importance"
- At each step, the "importance" flows out along the outgoing links
  - And new stuff comes in through the incoming links
  - Not all flows out (damping factor $d \approx 0.85$) since paths are not infinite
- We iterate until the amount is the same for all pages
  - The final value gives an indication of how important a page is: a page is more important when there are more links from pages that are themselves important
- This is a global fixpoint calculation: the PageRank values are the entries of the dominant eigenvector of the Web adjacency matrix with damping factor

$$\mathbf{R} = \begin{bmatrix} PR(p_1) \\ PR(p_2) \\ \vdots \\ PR(p_N) \end{bmatrix}$$

PageRank vector

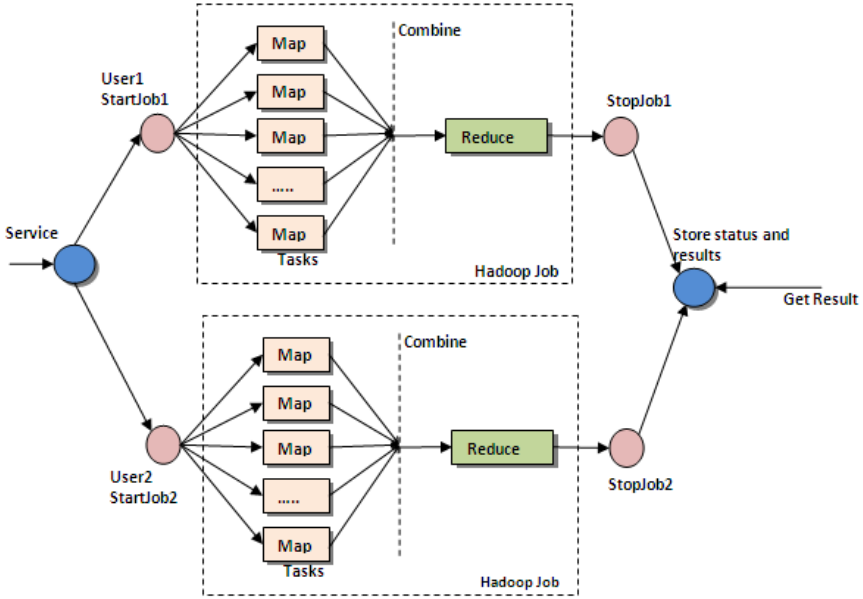
$$\mathbf{R} = \begin{bmatrix} (1-d)/N \\ (1-d)/N \\ \vdots \\ (1-d)/N \end{bmatrix} + d \overbrace{\begin{bmatrix} \ell(p_1,p_1) & \ell(p_1,p_2) & \cdots & \ell(p_1,p_N) \\ \ell(p_2,p_1) & \ddots & & \vdots \\ \vdots & & \ell(p_i,p_j) & \\ \ell(p_N,p_1) & \cdots & & \ell(p_N,p_N) \end{bmatrix}}^{\textit{Normalized Web adjacency matrix}} \mathbf{R}$$

PageRank equation: multiply R by adjacency matrix and adjust with damping factor

© 2010 Peter Van Roy

Map-Reduce for Clouds

- Map-reduce is an important functional computation primitive for data-intensive algorithms [Dean & Ghemawat, OSDI04]
  - It is horizontally scalable and works well on large data sets residing in a cloud
  - The user specifies a map function that processes a list of key/value pairs to generate a new list
  - The user specifies a reduce function that combines the new key/value pairs to get the result

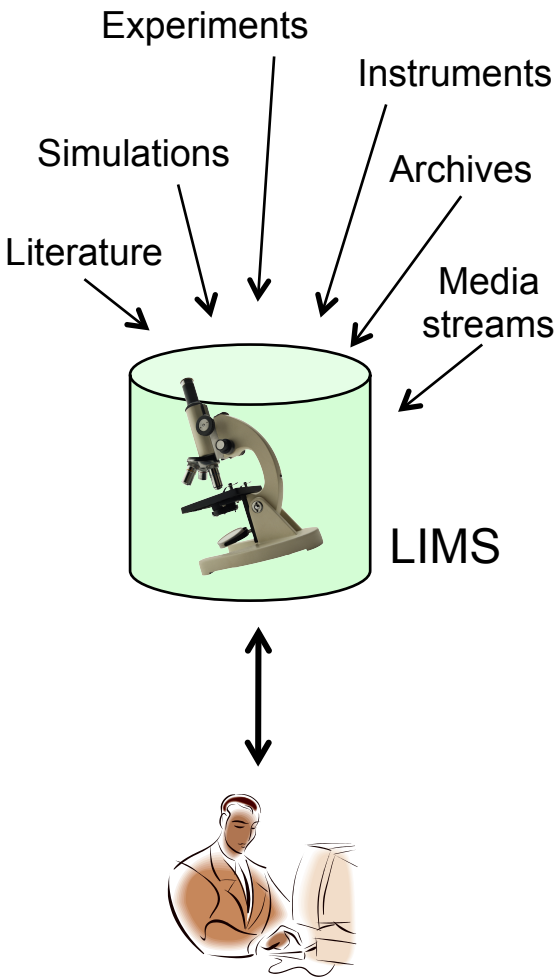


- The map-reduce run-time handles partitioning the input data, scheduling execution across a set of machines, fault tolerance (machine failures), and managing inter-machine communication
- Many calculations can be done easily: PageRank (!), word counting in large documents, search in large documents, reverse Web-link graph, inverted index, distributed sort, etc.

© 2010 Peter Van Roy

Data Exploration for Science ("Fourth Paradigm")

- Three paradigms of scientific exploration
  - Empirical: describing natural phenomena
  - Theoretical: defining and reasoning on models
  - Computational: simulating complex phenomena
- New fourth paradigm: data exploration
  - Capturing natural or simulated data
  - Using data-intensive algorithms to analyze data
- For example, astronomers do not look through telescopes directly; they "look" through the lens of computation
  - For large telescopes, software costs dominate the capital expenditure (e.g., Sloan Digital Sky Survey and others)



© 2010 Peter Van Roy

Laboratory Information Management Systems (LIMS)

- Pipeline from instrument or simulation into a data archive
  - Recalibrate and clean the data, normalize (like relational databases), publish on Internet
- Commercial LIMS exist but are too expensive
- General idea: whenever lots of data are created, use cloud-based tools to analyze and archive them
  - Need files that are self-describing (meta-data)
  - Clustering algorithms, data mining algorithms, search, visualization tools,
  - Will be used by other groups than just scientists (e.g., recording life stories for ordinary people): this is an opportunity

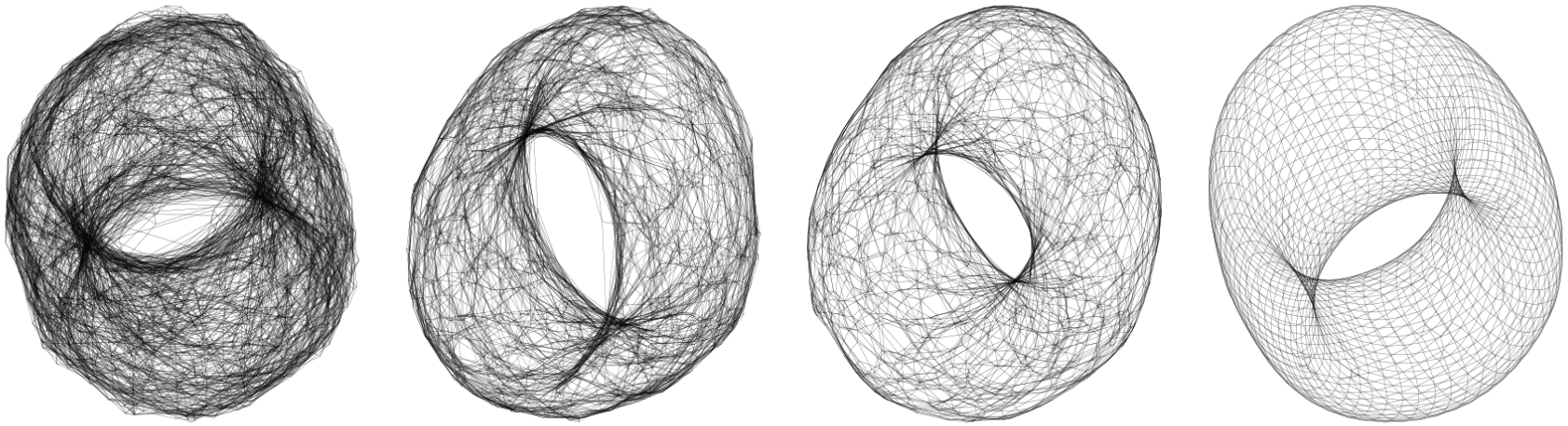
© 2010 Peter Van Roy

Using the Network as Data

- An important form of data-intensive algorithm uses the structure of a large network as its "data"
  - Gossip algorithms use random peer-to-peer communication between nodes to solve global problems (such as diffusion, search, aggregation, monitoring, topology management)
  - Structured overlay networks self-organize to provide efficient and robust communication and storage abilities
  - Swarm intelligence describes the collective behavior of decentralized, self-organizing systems (natural or artificial). In addition to collaboration and coordination (e.g., flocking behavior), it can also be used for solving global problems (e.g., ant-colony optimization).
- Each of these three approaches can be used as a base for building applications (see, e.g., Scalaris)

© 2010 Peter Van Roy

Gossip Algorithms: Topology Management



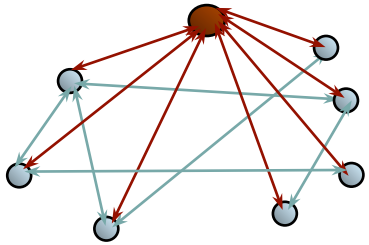
after 3 cycles | after 5 cycles | after 8 cycles | after 15 cycles

- The T-Man algorithm does topology management using a gossip algorithm
  - Each node periodically picks a random node and exchanges information with it
  - Each node has a ranking function that knows what distances nodes are supposed to have in the desired topology (i.e., a torus emerging from a random graph)
- The topology emerges in a few cycles (one cycle = one update per node)
- The algorithm is efficient, extremely robust, and can track changes

© 2010 Peter Van Roy

Structured Overlay Networks: Taming Peer-to-Peer

- Hybrid (client/server)
  - Napster

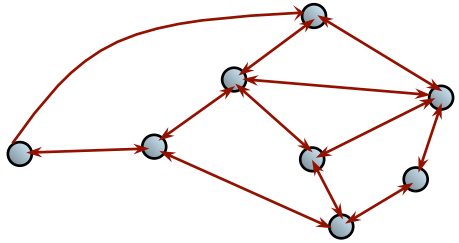


R = *N*-1 (hub)

R = 1 (others)

H = 1

- Unstructured overlay
  - Gnutella, Kazaa, Morpheus, Freenet, …
  - Uses flooding

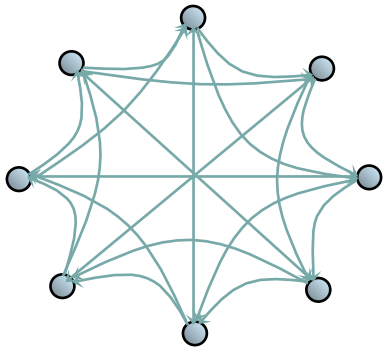


R = ? (variable)

H = 1…7

(but no guarantee)

- Structured overlay
  - Exponential network with augmented ring structure
  - DHT (Distributed Hash Table), e.g., Chord, DKS, Scalaris, Beernet
  - Self-organizes upon node join/leave/failure



R = log *N*

H = log *N*

(with guarantee)

© 2010 Peter Van Roy

Heisenberg Applications

© 2010 Peter Van Roy

Heisenberg Applications

- Let us now combine cloud computing with data-intensive applications
  - What can a cloud offer to a data-intensive application?
  - Easy: it offers a lot of resources (storage and computing power) at low cost (pay per use)
  - So what's the big deal?
- The big deal is elasticity: the ability to scale resource usage up and down rapidly according to demand
  - Elasticity is new: it did not exist before clouds
- Let us see what elasticity can offer to applications…

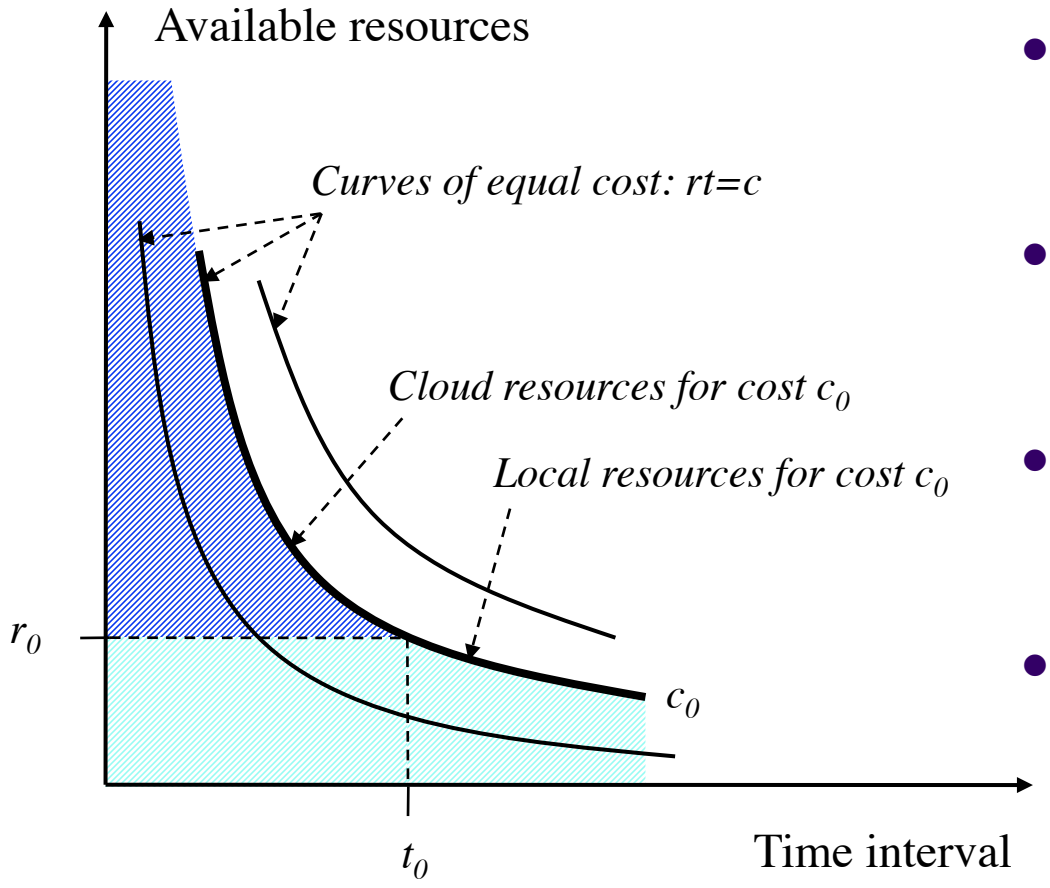
© 2010 Peter Van Roy

Computational Heisenberg Principle (1)

- A cloud has two key properties:
  - Pay per use: pay only for the resources actually used
  - Elasticity: ability to scale resource usage up (and down) rapidly
- Therefore for a given cost, as the time interval decreases more resources can be made available:
  - **For a given maximum cost, the product of resource amount and usage time is less than a constant**
  - There is an analogy with Heisenberg's Uncertainty Principle in physics: the product of uncertainty in time and uncertainty in energy is less than a constant. This increases the probability of histories with arbitrarily high energies if the time period is short enough.
    - According to most physicists, this is a statement about the nature of the system itself, and not about our ability to measure the system!
- This principle opens the door to many new applications
  - Just as Heisenberg's Uncertainty Principle is a positive force (it drives many physical effects, such as black-hole emission), the Computational Heisenberg Principle is also a positive force

© 2010 Peter Van Roy

Computational Heisenberg Principle (2)

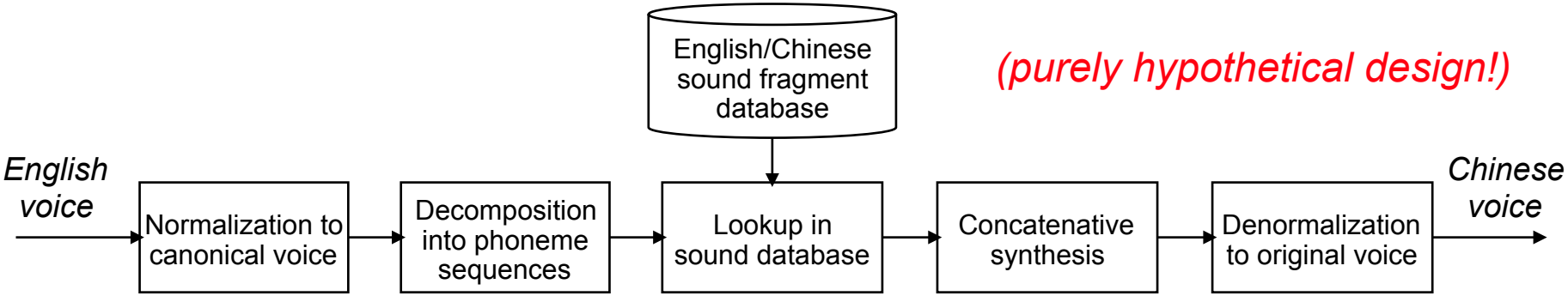


- The **light blue area** shows local resources available for maximum cost c_0
- The **dark blue area** shows additional resources available in the cloud
- For a given cost, as the time interval becomes shorter, more resources are available
- The dark blue area is the home of many new "Heisenberg applications"

© 2010 Peter Van Roy

Example Heisenberg Application: Real-Time Voice Translation (1)

- The pieces of this application already exist; for example the IRCAM research institute has implemented many of them
- It requires combining domain knowledge (in sound and language) with an enormous sound fragment database, hosted on a cloud



- Performance will be gradually improved through feedback from bilingual speakers and improved speech recognition technology
- Franz Och, head of translation services at Google, announced recently that they are working on this (Feb. 10, 2010)

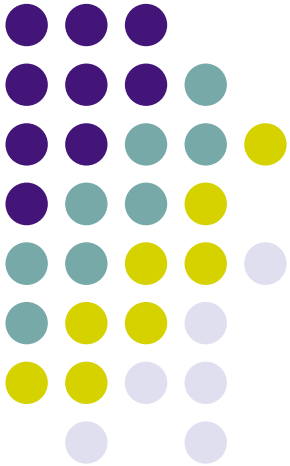
© 2010 Peter Van Roy

Example Heisenberg Application: Real-Time Voice Translation (2)

- The application has enormous storage and computational requirements
- The application uses these resources in bursts ("Heisenberg application")
- Assume an average person spends 15 minutes per day on the phone to China
  - This is 15/1440 ≈ 1/100 of their time
  - For the same cost, this makes available 100× the resources that one person could use continuously (i.e., at low cost)
  - Of course, more is available if you pay more

© 2010 Peter Van Roy

Outlook



© 2010 Peter Van Roy

Outlook

- Cloud computing is much more than enterprise computing and mobile applications
  - Data-intensive applications of all kinds can be hosted on clouds
- Heisenberg applications are coming
  - You don't have to be Google or Amazon to play this game
  - Anybody can play this game, including humble master's students and obscure Belgian companies!
  - All you need are a domain expert + domain data + a cloud expert
  - Domain knowledge is the key!
- This is a perfect topic for an ICTEAM project
  - We have expertise in large-scale distributed computing and domain knowledge (e.g., signal processing, machine learning, data mining)
  - Let us submit a project in this area (ARC?)

© 2010 Peter Van Roy