中国科学院大学
University of Chinese Academy of Sciences

Big Data System and Large – Scale Datasets Analysis

Yi Sun, Shimin Chen

Welcome to large-Scale Datasets analysis

Theme of this Part

Large-Scale Data Management

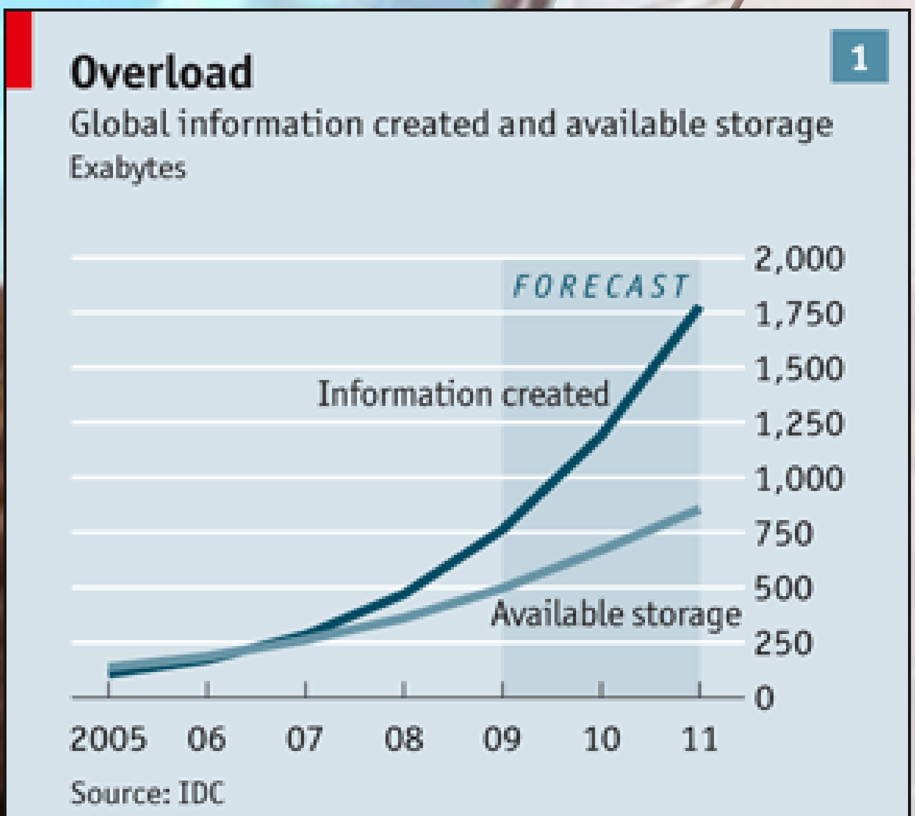
Data Science and Analytics

Big Data Analytics

- How to manage very large amounts of data and extract value and knowledge from them

Data is the new Oil

We are producing more data than we are able to store although the unit cost of disk storage decreases dramatically!

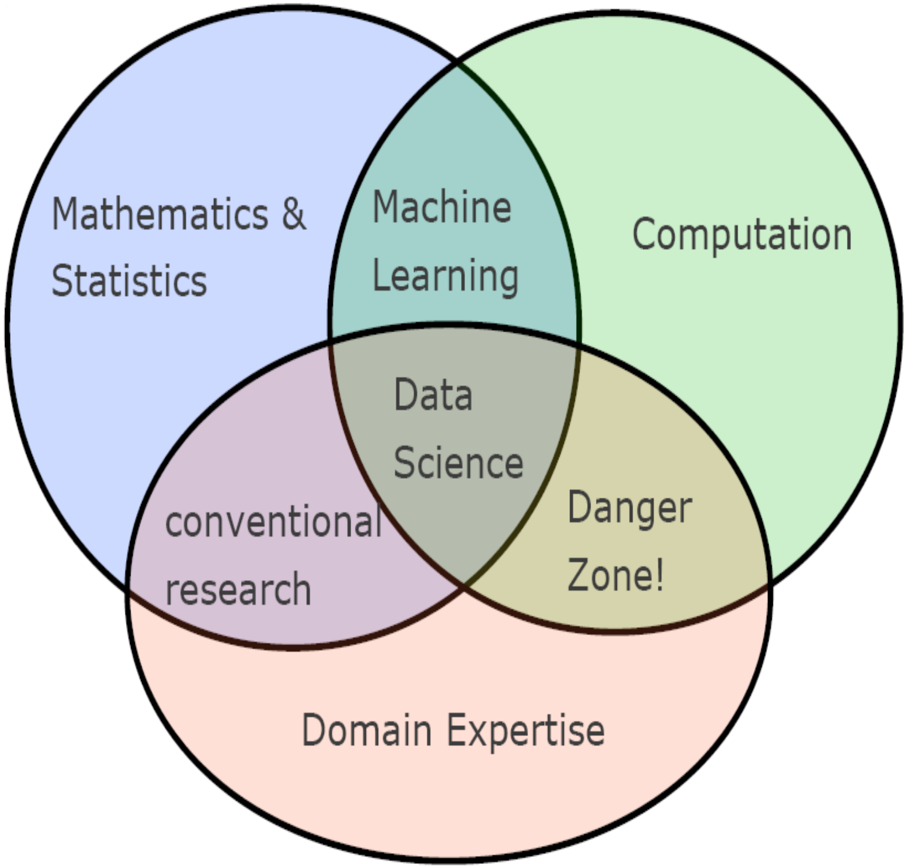


Year	Unit cost
1956	$10,000/MB
1980	$193/MB
1990	$9/MB
2000	$6.9/GB
2010	$0.08/GB

http://ns1758.ca/winch/winchest.html

Welcome to Large-Scale Datasets Analysis

- **Data Science: automatically extracting knowledge from data**
  - Mathematics & Statistics
  - Machine Learning
  - Domain Expertise
- **Applications in Business**
  - Lots and lots
- **Applications in the Sciences**
  - Astronomy, Cosmology
  - High-energy Physics
  - Biology, Genomics
  - Neuroscience
  - The Social Sciences
- **Education, Medicine**
- **Government**



About myself

Name	Yi Sun
Education	Ph.D in theoretical physics
Dream of a physicist	To be able to explain all phenomena in the Universe with the minimum number of elements
Biggest problem	How was the universe born? What is the fundamental interaction?
Dream of a computer scientist	Given any computational problem, can we decide the computability and complexity based on an existing computation model?
Biggest problem P = NP?	P: deterministic polynomial time decidable NP: nondeterministic polynomial time verifiable

About myself

- **Recently many physics of physics are used in computer science.**
  - Quantum physics, thermodynamics, statistical physics, stochastic processes…(Quantum computing and communication, Ising model is NP-hard, Quantum gravity is NP-Hard. etc…)
  - Three Big E's Revolution

About myself

- **Physics is based on motion law, computer science is the study of algorithms. Constructive proof. (Is Nash equilibrium NP? Is WSP NP?, etc…)**
  - Physics perturbation
    - Quantum Gravity can't be perturbed
    - New models → superstring and M theory
  - Computer approximation
    - Many problems can't be approximated in current computing models.
    - New models → quantum machine

About myself

- **General interest: Networking applications & security**
  - Big data analysis
  - Broad interests in engineering (and theoretical) issues in networking
- **Specific interests**
  - Big data in Education
  - Network security and Mobile internet network

What is Big Data?

What makes data, “Big” Data?

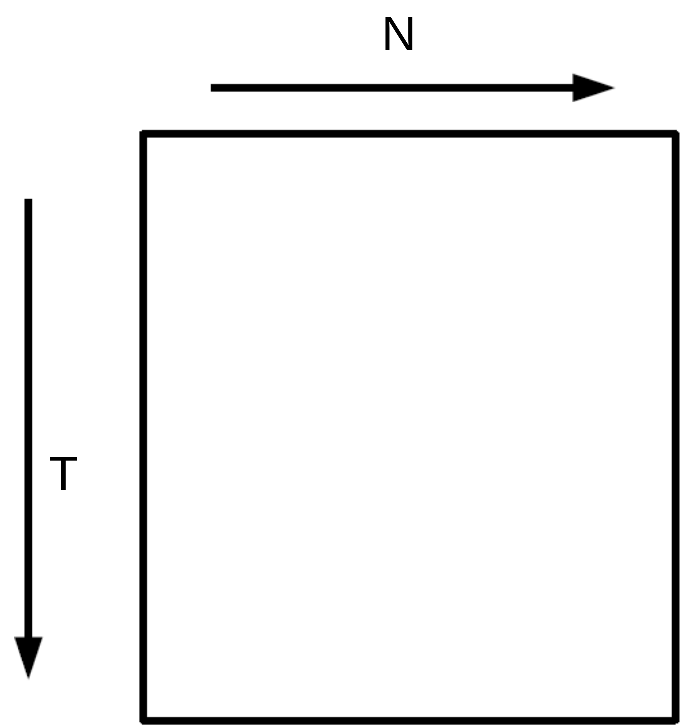
Big Data Definition

- No single standard definition…

"*Big Data*" is data whose scale, diversity, and complexity require new architecture, techniques, algorithms, and analytics to manage it and extract value and hidden knowledge from it…

Big Data Definition

- **Data often comes to in the form of a table**
  - N: dimension of each vector (possibly very sparse)
  - T: number of training samples (possibly infinite)
- **Big Data is large T, or large N, or both**
  - Large T, small N: great!
  - Infinite T, small N: on-line / streaming
  - Small T, large N: hell!
- **Problems:**
  - (distributed) data storage and access
  - can't use algo super-linear in T
  - Large N: overfitting
  - Parallelizing
  - Dealing with unbalanced set
  - Representing high-dim data

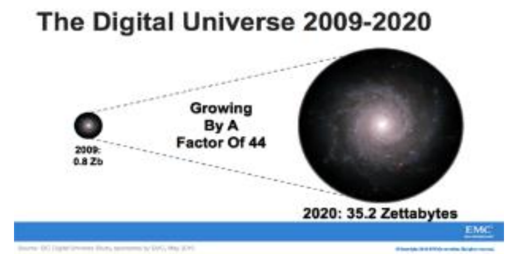


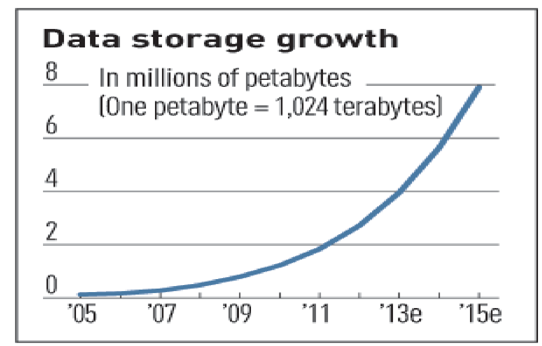
11

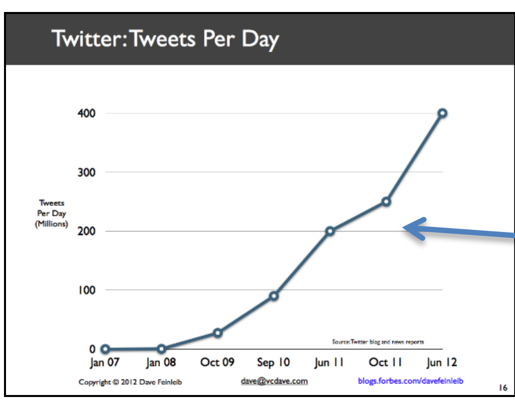
Characteristics of Big Data

1-Scale (Volume)

- Data Volume
  - 44x increase from 2009 2020
  - From 0.8 zettabytes to 35zb
- Data volume is increasing exponentially





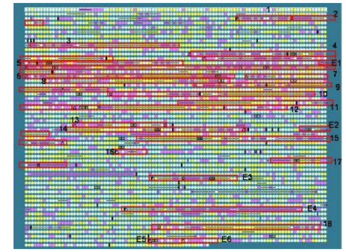
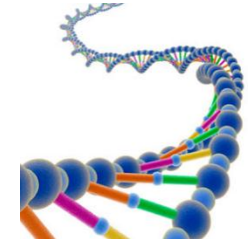
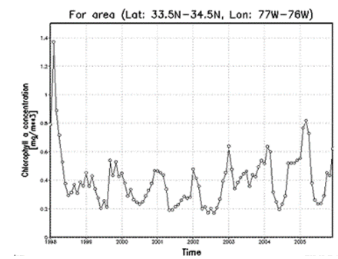


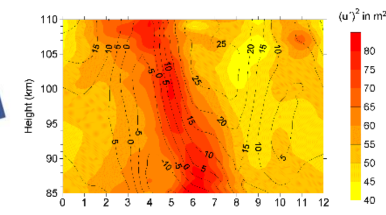
Exponential increase in collected/generated data

Characteristics of Big Data

2-Complexity (Variety)

- Various formats, types, and structures
- Text, numerical, images, audio, video, sequences, time series, social media data, multi-dim arrays, etc…
- Static data vs. streaming data
- A single application can be generating/collecting many types of data

To extract knowledge➔ all these types of data need be to linked together

Characteristics of Big Data

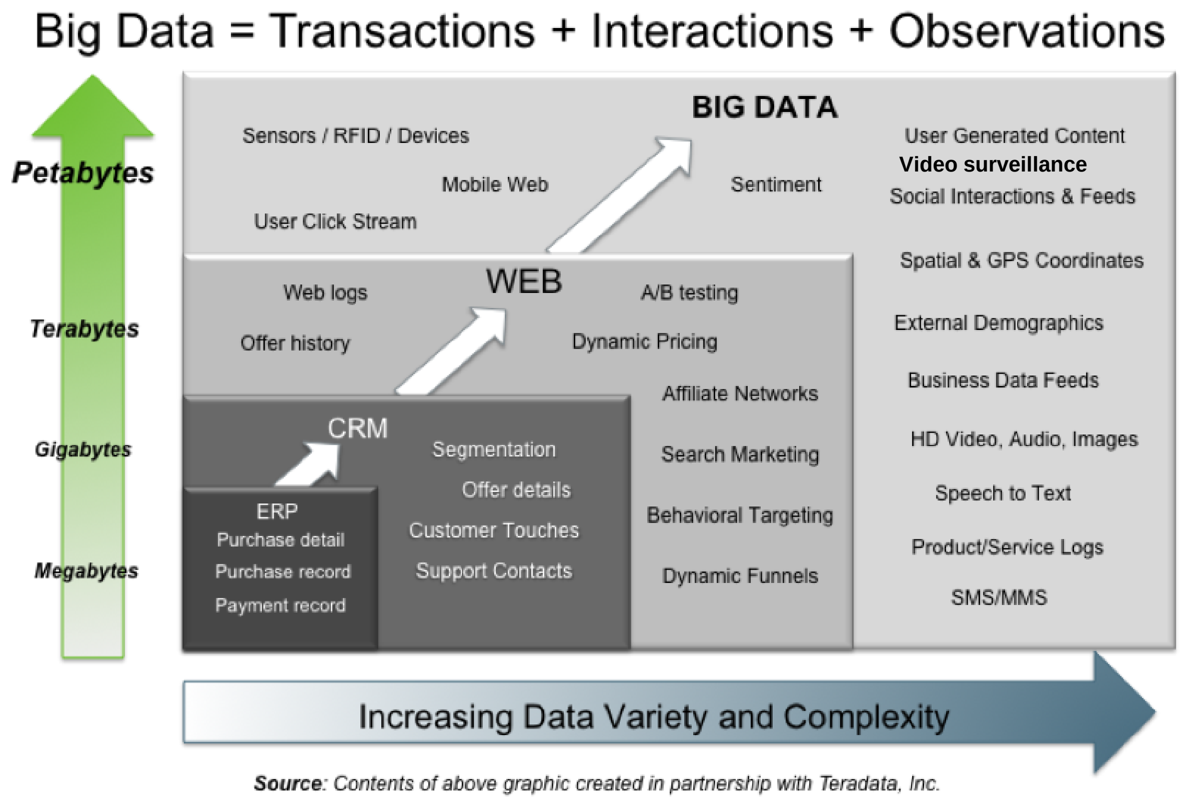
3-Speed (Velocity)

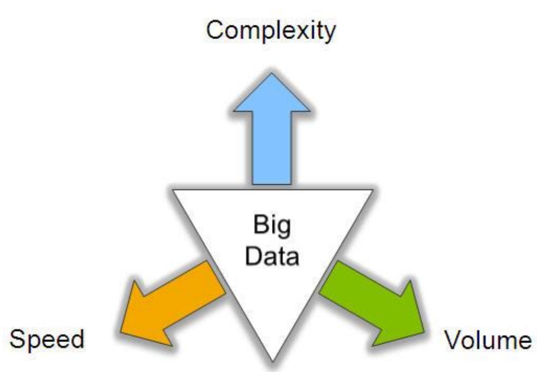
- Data is being generated fast and need to be processed fast
- Online Data Analytics
- Late decisions ➔ missing opportunities

- Examples
  - E-Promotions: Based on your current location, your purchase history, what you like ➔ send promotions right now for store next to you
  - Healthcare monitoring: sensors monitoring your activities and body ➔ any abnormal measurements require immediate reaction

Big Data: 3V's

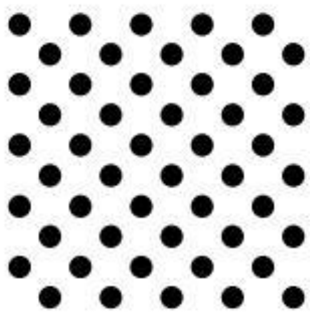

BIG DATA?
VOLUME
Large amounts of data.
VELOCITY
Needs to be analyzed quickly.
VARIETY
Different types of structured and unstructured data.


Big Data = Transactions + Interactions + Observations
Petabytes
Terabytes
Gigabytes
Megabytes
BIG DATA
Sensors / RFID / Devices
Mobile Web
User Click Stream
Sentiment
User Generated Content
Video surveillance
Social Interactions & Feeds
Spatial & GPS Coordinates
External Demographics
Business Data Feeds
HD Video, Audio, Images
Speech to Text
Product/Service Logs
SMS/MMS
WEB
Web logs
A/B testing
Offer history
Dynamic Pricing
Affiliate Networks
Search Marketing
Behavioral Targeting
Dynamic Funnels
CRM
Segmentation
Offer details
Customer Touches
Support Contacts
ERP
Purchase detail
Purchase record
Payment record
Increasing Data Variety and Complexity
Source: Contents of above graphic created in partnership with Teradata, Inc.


Complexity
Big Data
Speed
Volume

Some Make it 4V's

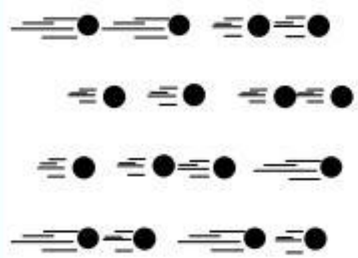
Volume



Data at Rest

Terabytes to exabytes of existing data to process

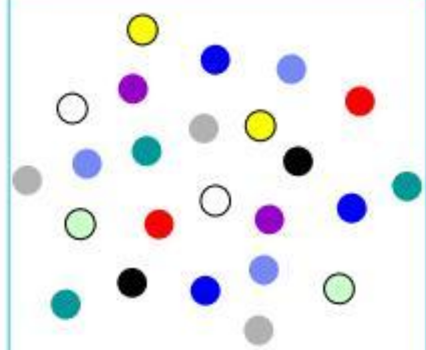
Velocity



Data in Motion

Streaming data, milliseconds to seconds to respond

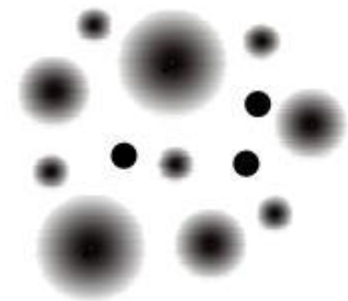
Variety



Data in Many Forms

Structured, unstructured, text, multimedia

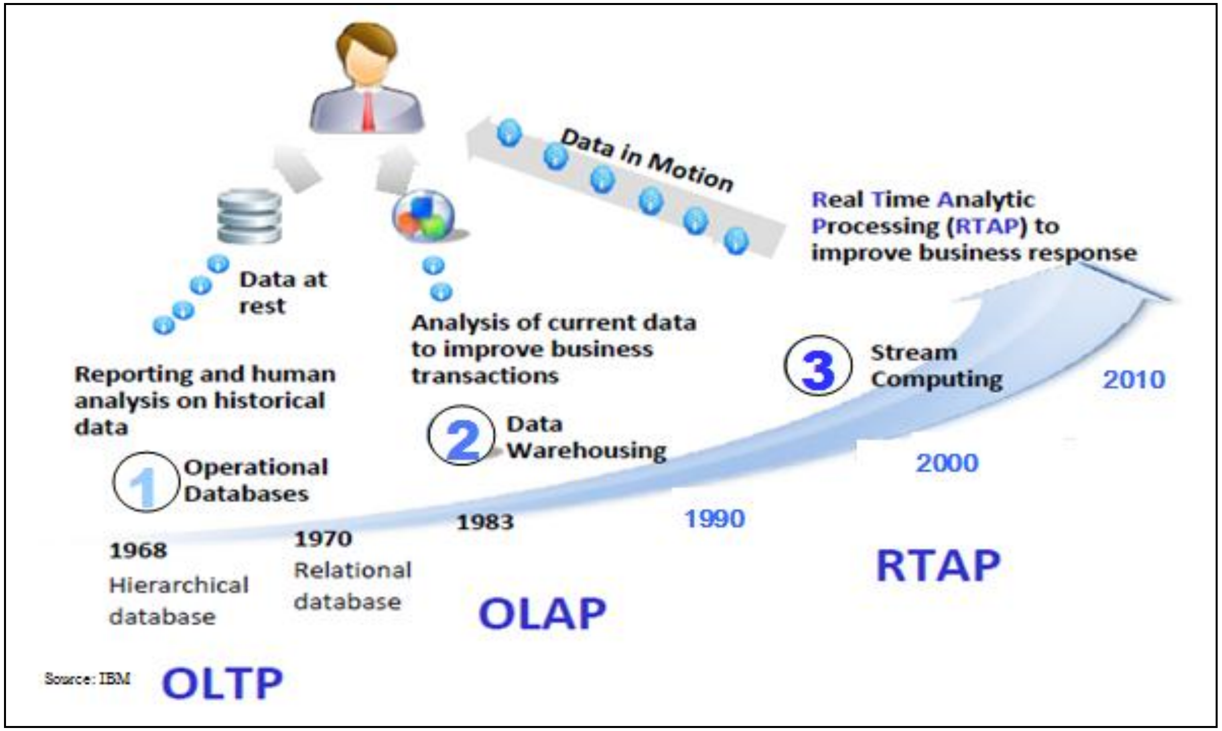
Veracity*



Data in Doubt

Uncertainty due to data inconsistency & incompleteness, ambiguities, latency, deception, model approximations

Harnessing Big Data



- **OLTP:** Online Transaction Processing (DBMSs)
- **OLAP:** Online Analytical Processing (Data Warehousing)
- **RTAP:** Real-Time Analytics Processing (Big Data Architecture & technology)

Big Data: More than Volume

Volume = Length $\times$ Width $\times$ Depth

Big Data Length: Collect & Compare

Big Data Width: Discover & Integrate

Big Data Depth: Analyze & Understand

Who’s Generating Big Data

- The progress and innovation is no longer hindered by the ability to collect data
- But, by the ability to manage, analyze, summarize, visualize, and discover knowledge from the collected data in a timely manner and in a scalable fashion

Soci
(all c

orks

video surveillance
(sensing china)

Mobile devices
(tracking all objects all the time)

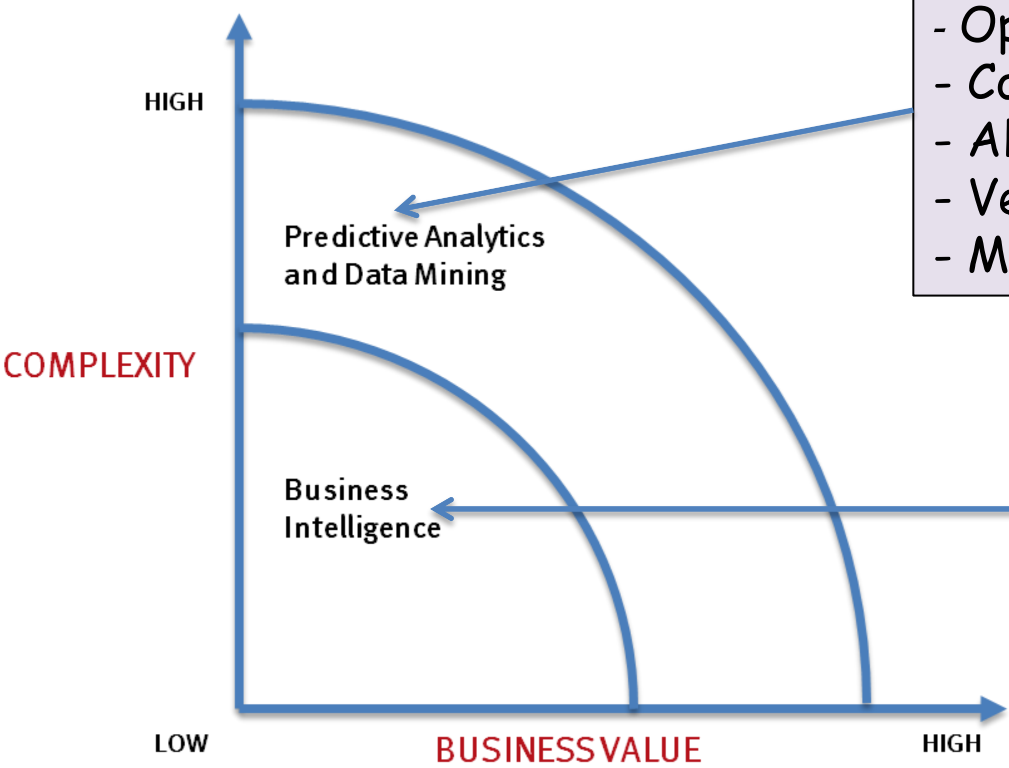
The Model Has Changed…

- **The Model of Generating/Consuming Data has Changed**

Old Model: Few companies are generating data, all others are consuming data

New Model: All of us are generating data, and all of us are consuming data

What's driving Big Data

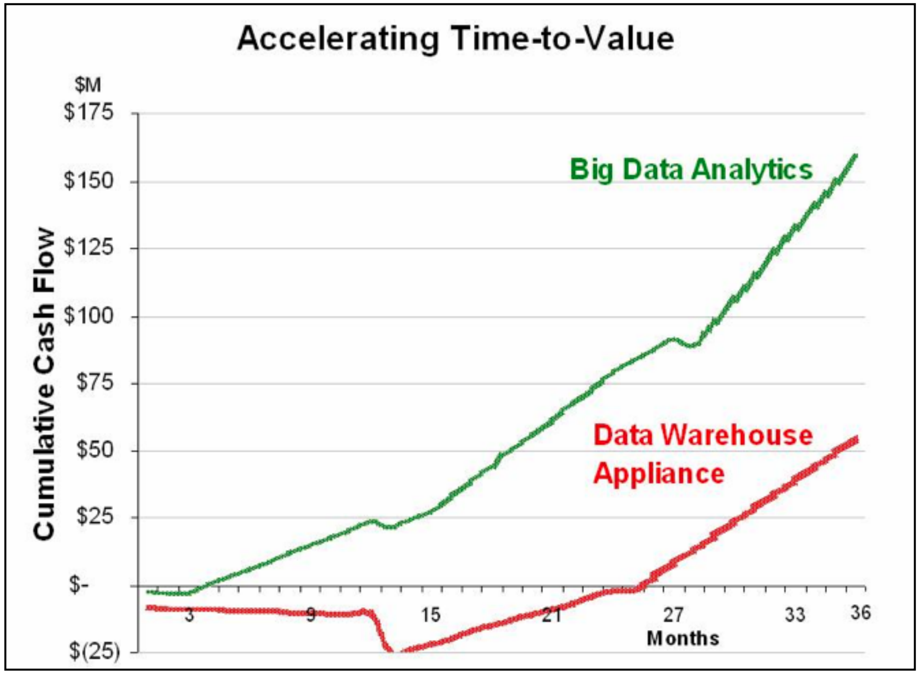


- Optimizations and predictive analytics
- Complex statistical analysis
- All types of data, and many sources
- Very large datasets
- More of a real-time

- Ad-hoc querying and reporting
- Data mining techniques
- Structured data, typical sources
- Small to mid-size datasets

Value of Big Data Analytics

- Big data is more real-time in nature than traditional DW applications
- Traditional DW architectures (e.g. Exadata, Teradata) are not well-suited for big data apps
- Shared nothing, massively parallel processing, scale out architectures are well-suited for big data apps



Big Data: What is the Big deal?

- **Many success stories**
  - Google:grew from processing 100 TB of data a day with MapReduce in 2004 [45] to processing 20 PB a day
  - Facebook boasting of 2.5 petabytes of user data, growing at about 15 terabytes per day.
  - Twitter…
  - 腾讯，百度，阿里，中国电信，360 about 10PB on line per day.

- There will be a shortage of talent necessary for organizations to take advantage of big data. By 2018, the United States alone could face a shortage of 140,000 to 190,000 people with deep analytical skills as well as 1.5 million managers and analysts with the know-how to use the analysis of big data to make effective decisions.!

http://www.mckinsey.com/Insights/MGI/Research/Technology_and_Innovation/Big_data_The_next_frontier_for_innovation!

Big Data Values

Big data can generate significant financial value across sectors

US health care
- $300 billion value per year
- ~0.7 percent annual productivity growth

Europe public sector administration
- €250 billion value per year
- ~0.5 percent annual productivity growth

Global personal location data
- $100 billion+ revenue for service providers
- Up to $700 billion value to end users

US retail
- 60+% increase in net margin possible
- 0.5–1.0 percent annual productivity growth

Manufacturing
- Up to 50 percent decrease in product development, assembly costs
- Up to 7 percent reduction in working capital

SOURCE: McKinsey Global Institute analysis

Applications for Big Data Analytics

Smarter Healthcare

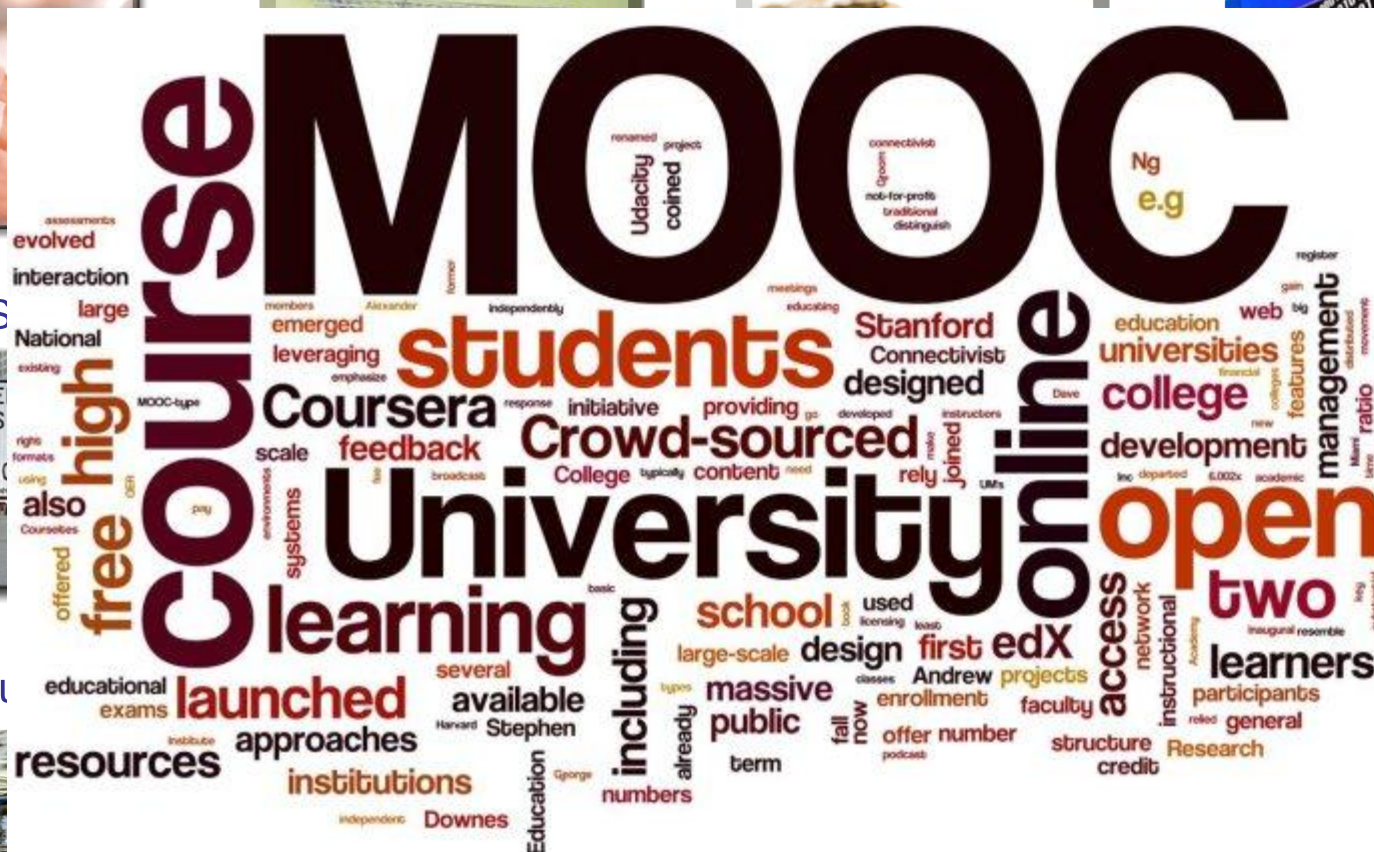
Multi-channel

Finance

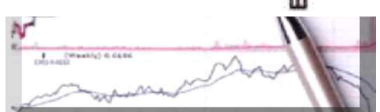
Log Analysis

Homeland S

h Quality

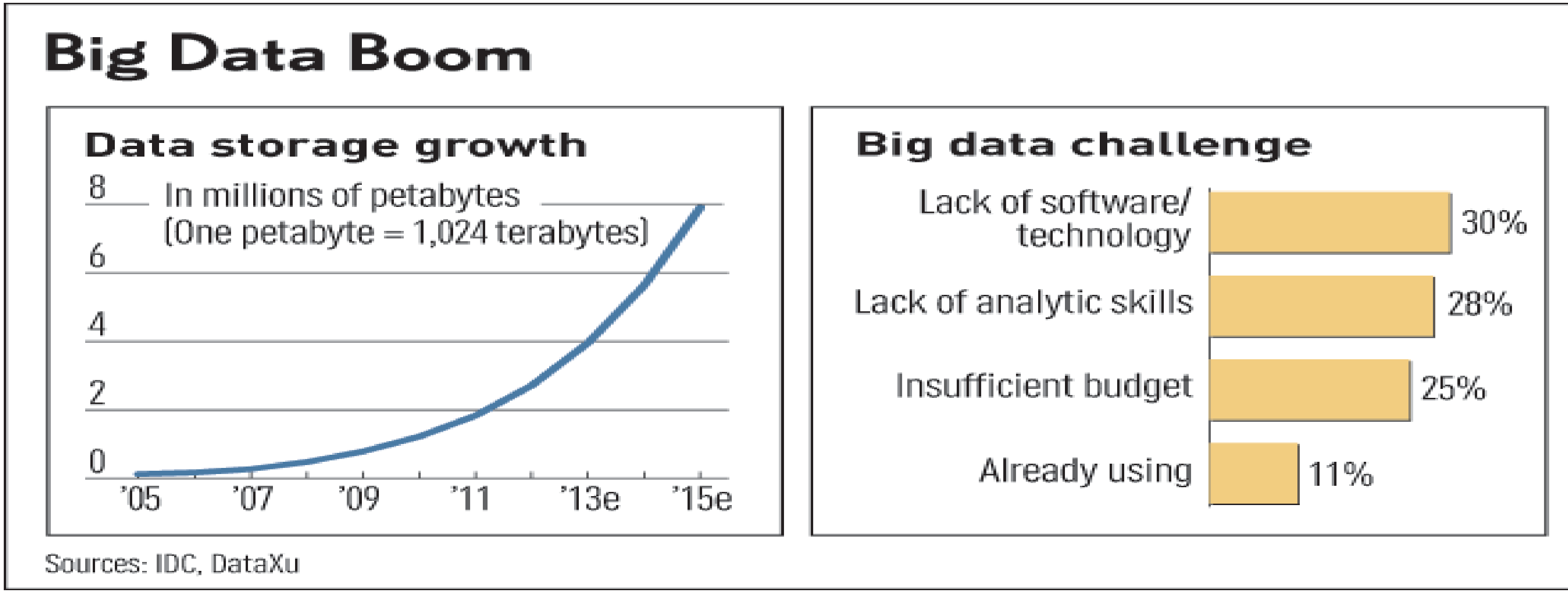


Manufactu



hurn, NBO

Challenges in Handling Big Data



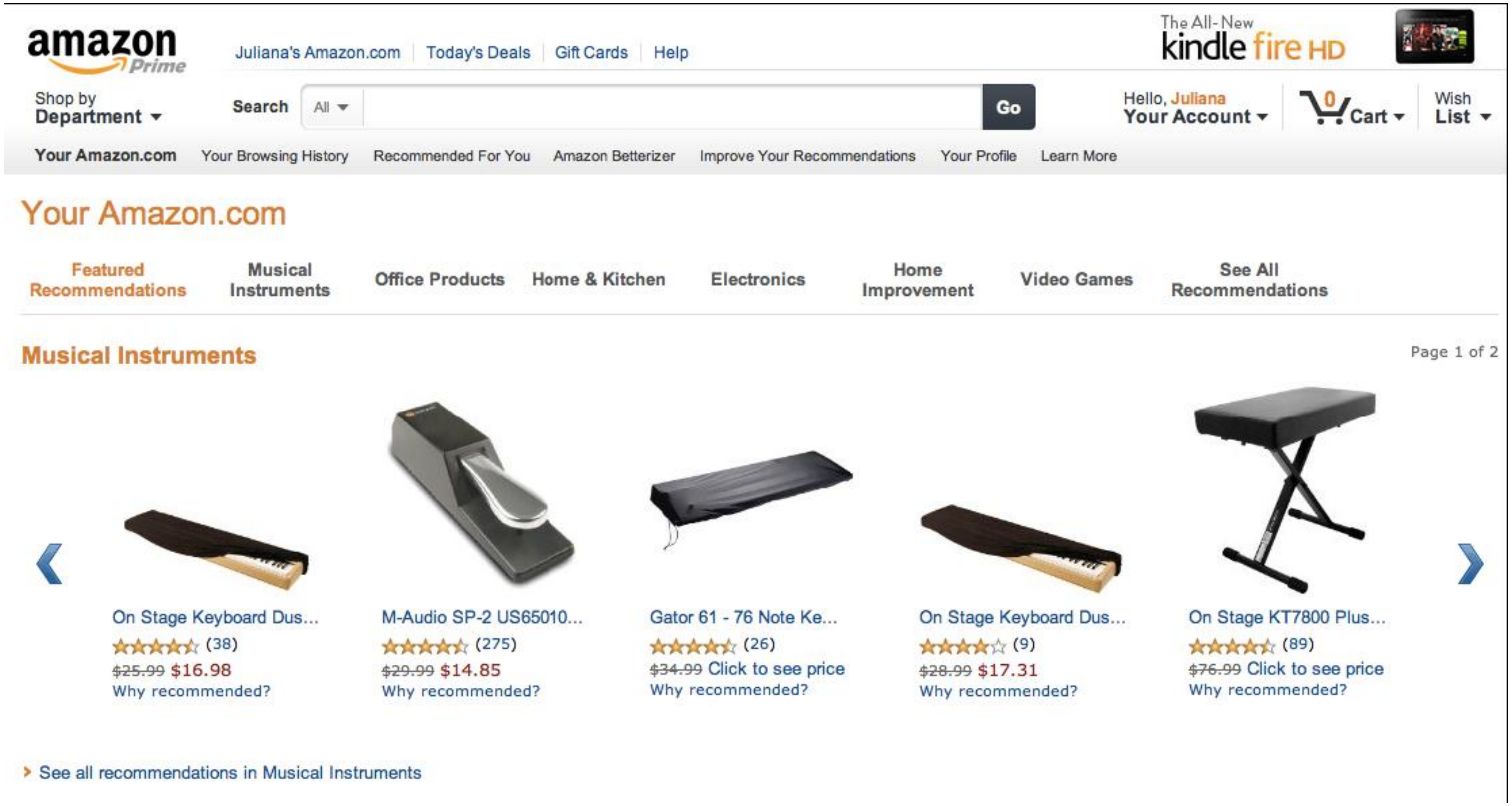
- **The Bottleneck is in technology**
  - New architecture, algorithms, techniques are needed
- **Also in technical skills**
  - Experts in using the new technology and dealing with big data

Big Data: New Opportunities

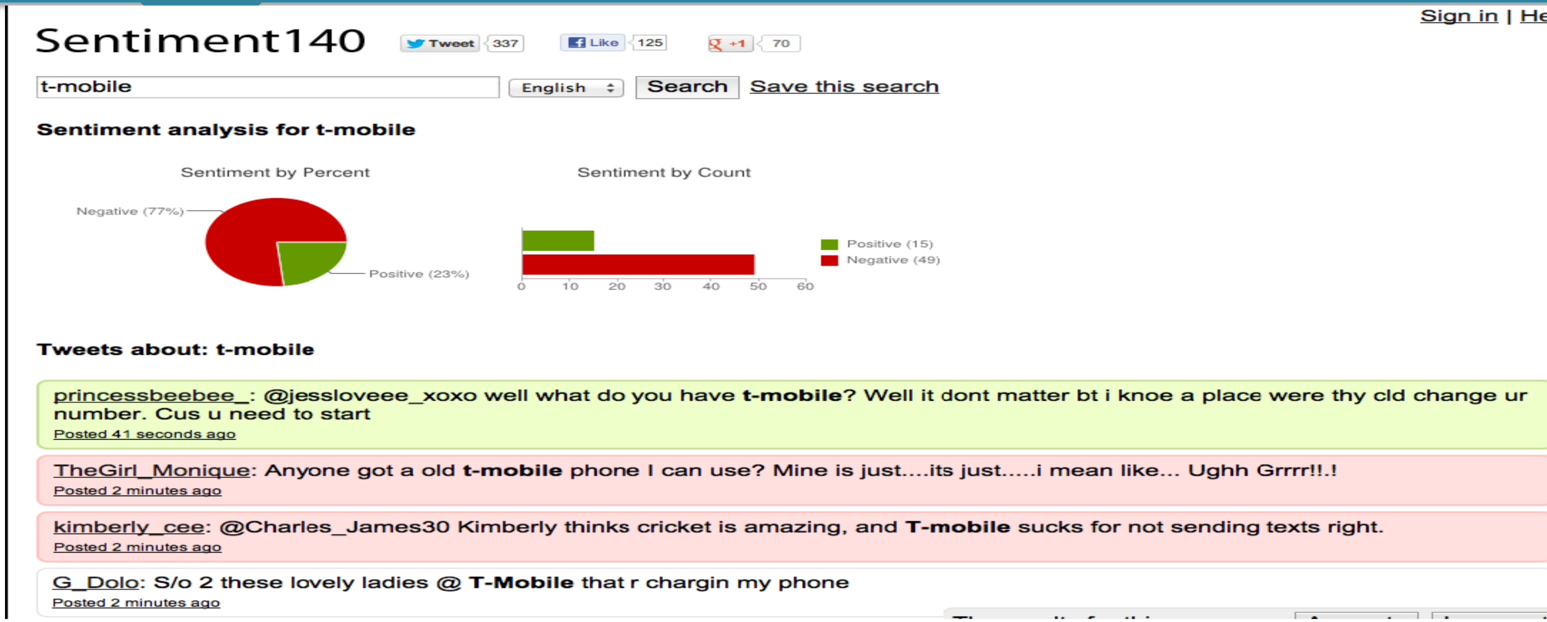
- Smart Cities: 50% of the world population lives in cities !
  - Census, crime, emergency visits, taxis, public transportation, real estate, noise, energy, …!
- Cities are making their data available!!
  - http://www.data.gov/united-states-datasites!
  - https://nycopendata.socrata.com/!
  - http://www.datatang.com/ (数据堂)
- Make cities more efficient and sustainable, and improve the lives of their citizens *!*

Big Data: New Opportunities

- Data is currency: companies are profiting from knowledge extracted from Big Data!
  - Better understand customers, targeted advertising, …!

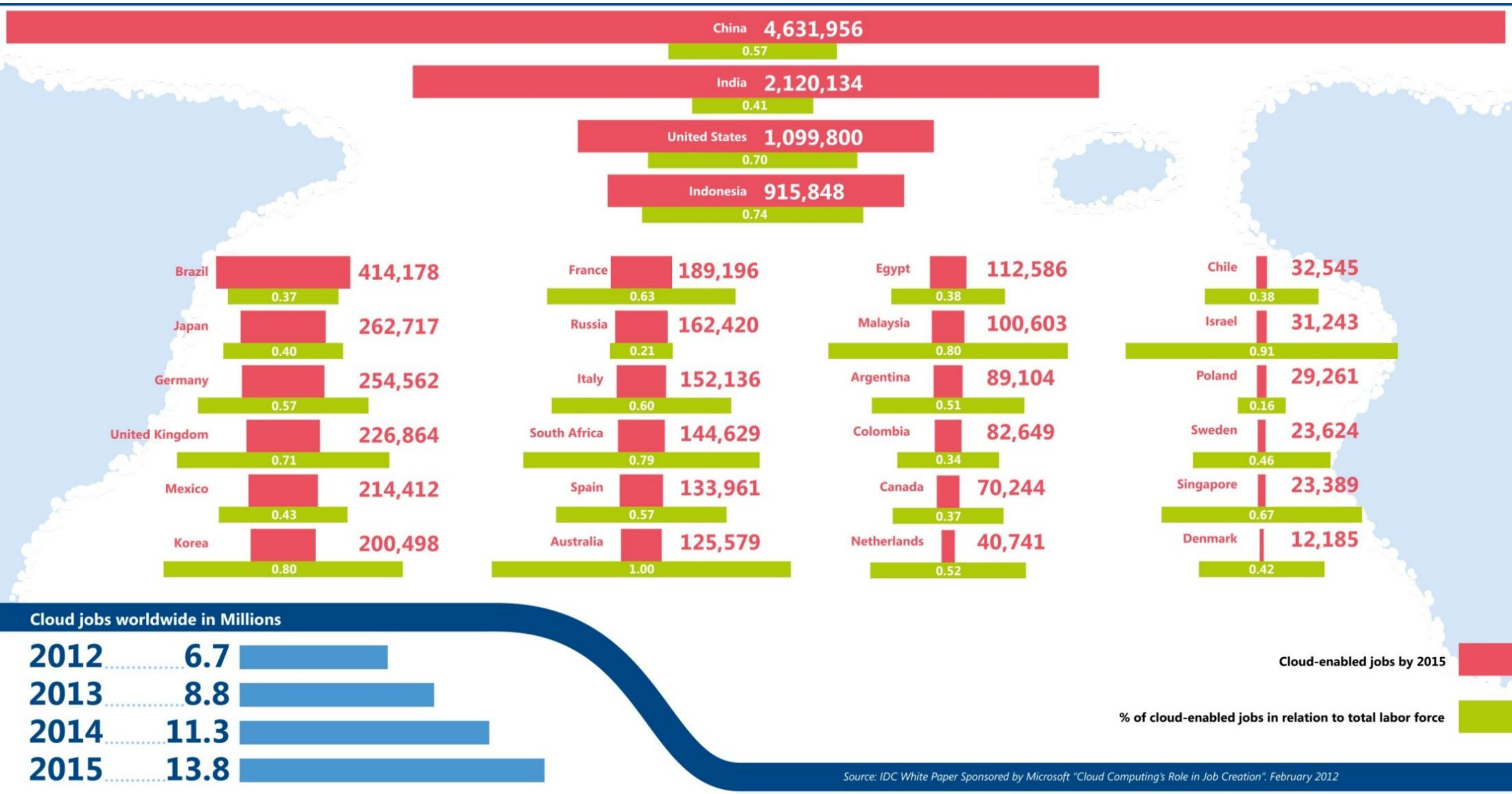


Big Data: New Opportunities



http://blogs.wsj.com/venturecapital/tag/big-- - -data/

Jobs v. Countries



What will we learn?

- **Based on different types of data:**
  - Data is **high dimensional**
  - Data is a **graph**
  - Data is **never-ending**
  - Data is **labeled**
- **Based on different models of computation:**
  - MapReduce (Dr. Chen's lectures)
  - Streams
  - Passive vs. Active (online) algorithms

What will we learn?

- We will learn to solve real-world problems:
  - Recommender systems
  - Association rules
  - Link analysis
  - Duplicate, spam detection
  - Big data in education (through Projects)
- We will learn various "tools":
  - Linear algebra (SVD, Rec. Sys., Communities)
  - Optimization (stochastic gradient descent)
  - Dynamic programming (frequent itemsets)
  - Hashing (LSH, Bloom filters)
  - Machine learning techniques……

Text and Ref. Books

- [MMDS] Anand Rajaraman and Jeffrey D. Ullman. Mining of Massive Datasets. Cambridge University Press, 2011.

What You Need to Do

- Enthusiasm!
  - To read and explore new research or app ideas
  - To actively participate in the discussion
- Your prerequisite
  - Basic linear algebra and programming skill
- Your workload
  - class participation
    - actively participate in class discussions
    - make insightful comments and/or initiate interesting discussions
  - exam
  - class project

Class Project

- Goal: obtain hands-on industry and research experience
- I'll suggest potential topics
- You may also choose your own topic
- Emphases
  - Application ideas
  - Research Algorithm design and implementation
  - Teamwork

Q&A

Q&A

- No stupid questions, but it is stupid if not ask!
- Ask a good question, and impress your professor and classmates!

Distribute Hash Table

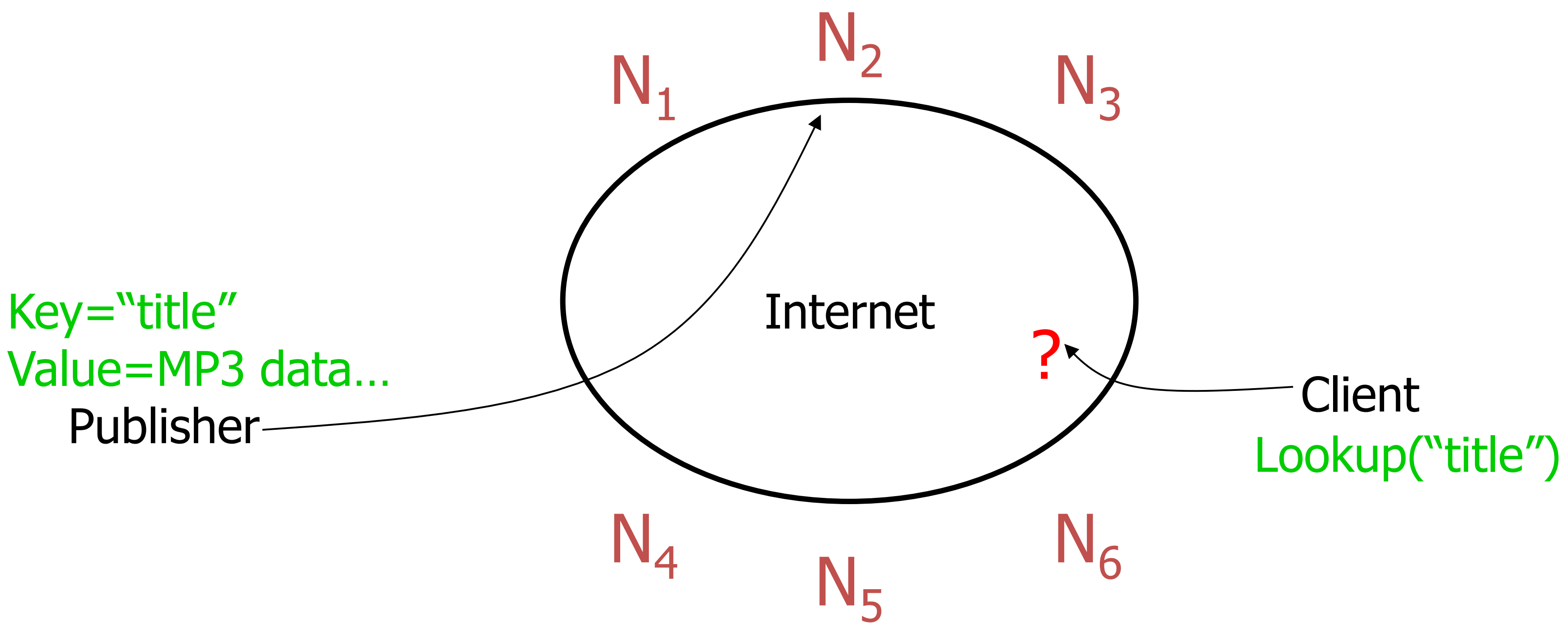
Distributed Hash Table

- Hash table spread over many nodes
  - Distributed over a wide area
- Main design goals
  - *Decentralization*
    - no central coordinator
  - *Scalability*
    - efficient even with large # of nodes
  - *Fault tolerance*
    - tolerate nodes joining/leaving

A Peer-to-peer Storage Problem

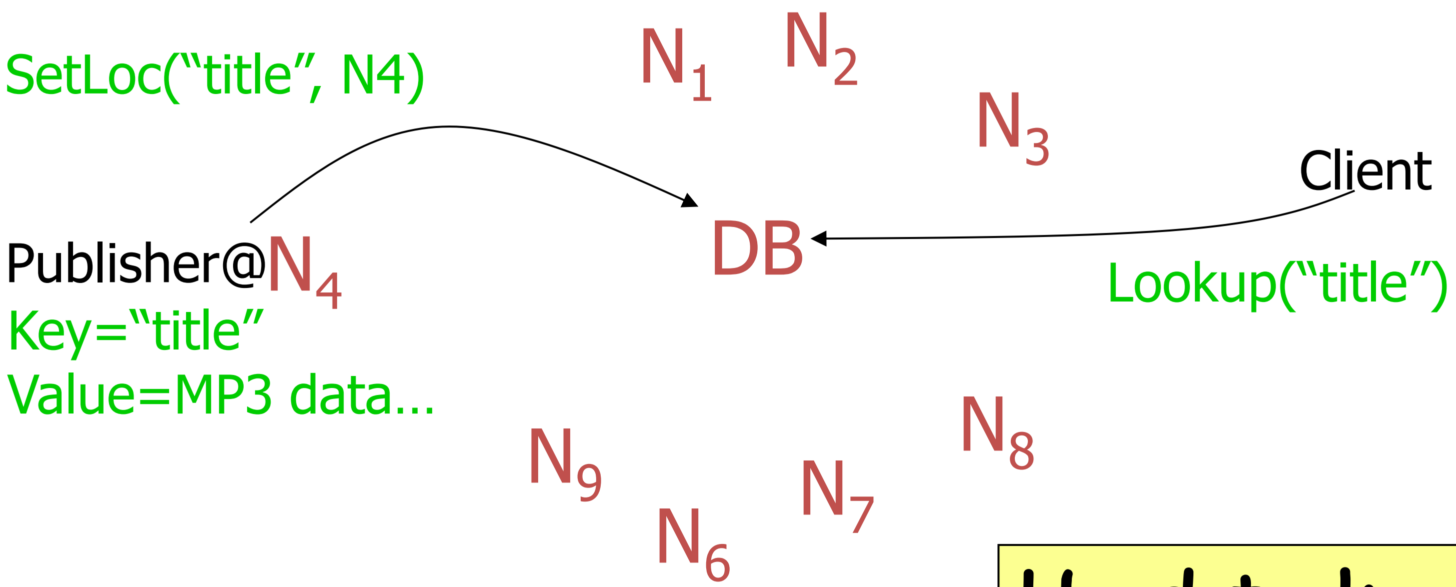
- 1000 scattered music enthusiasts
- Willing to store and serve replicas
- How do you find the data?

The Lookup Problem



Dynamic network with N nodes, how can the data be found?

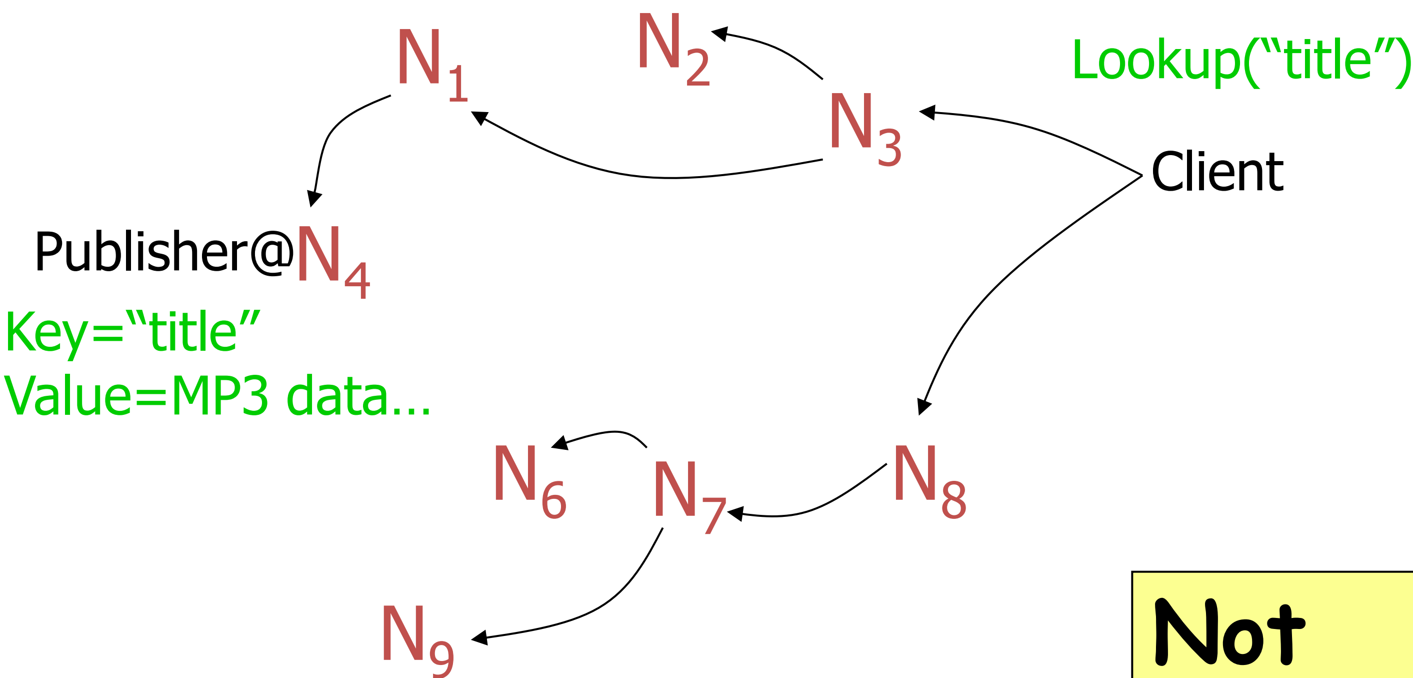
Centralized Lookup (Napster)



Hard to keep the data in the server updated

Simple, but O(N) state and a single point of failure

Flooded queries (Gnutella)



Not scalable

Robust, but worst case $O(N)$ messages per lookup

So Far

- Centralized :

  - Table size – O(n)

  - Number of hops – O(1)

- Flooded queries:

  - Table size – O(1)

  - Number of hops – O(n)

Data Lookup Approaches

- Napster approach:
  - 1 root server (or set of root servers) that know the node location of data objects
  - *not scalable, not resilient*

- Gnutella approach:
  - broadcast search to all known neighbors until the object is found
  - *scalability problems*

Data Lookup Approaches

- Superpeers (KaZaA, Gnutella Reflectors)
  - scalability through hierarchy
  - questions about resiliency
    - creating many "little Napsters"
- Freenet symmetric lookup
  - forward lookup requests to a node that is "closer" to the data object
  - focus on anonymity makes it difficult to have predictable topologies; also makes data stewardship difficult

We Want

- Efficiency : O(*log*(*N*)) messages per lookup
  - N is the total number of servers
- Scalability : O(*log*(*N*)) state per node
- Robustness : surviving massive failures

How Can It Be Done?

- How do you search in O(log(n)) time?
  - Binary search
- You need an ordered array
- How can you order nodes in a network and data items?

Hash
function

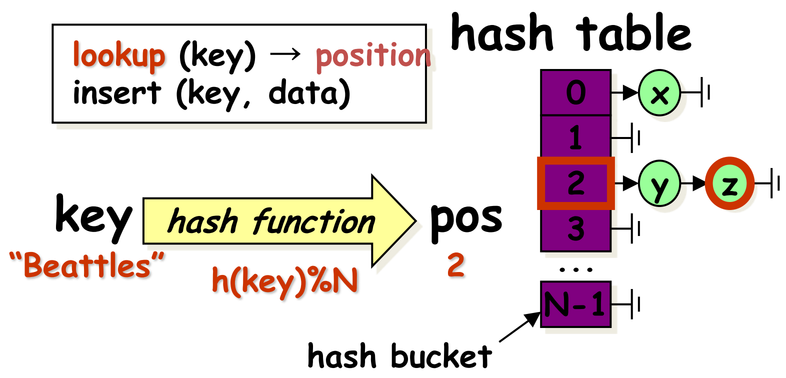
Directed Searches

- Idea
  - Assign particular nodes to hold particular content (or know where it is)
  - When a node wants this content, go to the node that is supposes to hold it (or know where it is)

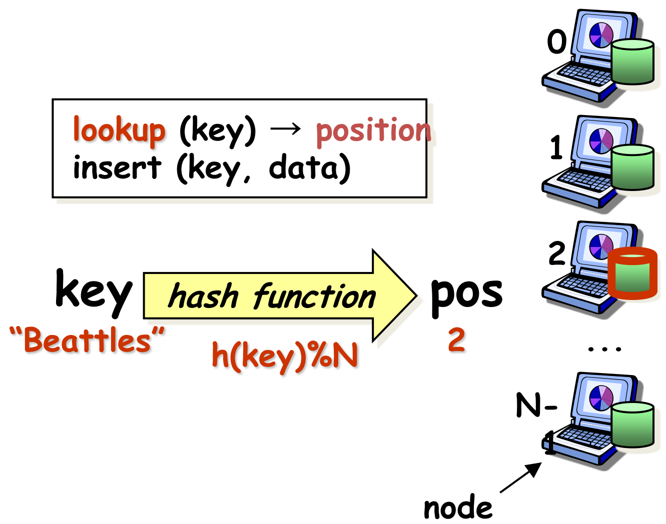
- Challenges
  - Avoid bottlenecks: distribute the responsibilities “evenly” among the existing nodes
  - Adaptation to nodes joining or leaving (or failing)
    - Give responsibilities to joining nodes
    - Redistribute responsibilities from leaving nodes

Idea: Hash Tables

- A hash table associates data with keys
  - Key is hashed to find bucket in hash table
  - Each bucket is expected to hold #items/#buckets items



- In a Distributed Hash Table (DHT), nodes are the hash buckets
  - Key is hashed to find responsible peer node
  - Data and load are balanced across nodes



DHTs: Problems

- Problem 1 (dynamicity): adding or removing nodes
  - With hash mod *N*, virtually every key will change its location!

$$h(k) \bmod N \neq h(k) \bmod (N+1) \neq h(k) \bmod (N-1)$$

- Solution: use consistent hashing
  - Define a fixed hash space
  - All hash values fall within that space and do not depend on the number of peers (hash bucket)
  - Each key goes to peer closest to its ID in hash space (according to some proximity metric)

DHTs: Problems (cont'd)

- Problem 2 (size): all nodes must be known to insert or lookup data
  - Works with *small* and *static* server populations
- Solution: each peer knows of only a few "neighbors"
  - Messages are routed through neighbors via multiple hops (overlay routing)

What Makes a Good DHT Design

- For each object, the node(s) responsible for that object should be reachable via a “short” path (small diameter)
  - The different DHTs differ fundamentally only in the routing approach
- The number of neighbors for each node should remain “reasonable” (small degree)
- DHT routing mechanisms should be decentralized (no single point of failure or bottleneck)
- Should gracefully handle nodes joining and leaving
  - Repartition the affected keys over existing nodes
  - Reorganize the neighbor sets
  - Bootstrap mechanisms to connect new nodes into the DHT
- To achieve good performance, DHT must provide low stretch
  - Minimize ratio of DHT routing vs. unicast latency

Service Discovery

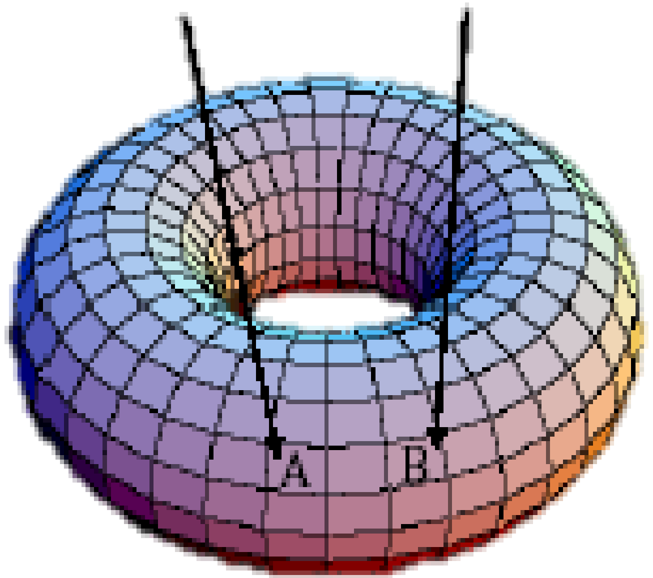
- Content Addressable Network (CAN)
  - Idea: associate to each item a unique coordinate in an (virtual) d-dimensional Cartesian space; each node owns a subspace
- Using Chord as Resolver Overlay (Chord)
  - Different from CAN: storage scheme is a ring, m bit identifier space for both keys and nodes (In Dr. Chen's Lectures)
- Both CAN and Chord are called distributed hash tables (DHT)
- other DHT Algorithms
  - Tapestry (Zhao et al)
  - Skip Graphs (Aspnes and Shah)

Example: CAN

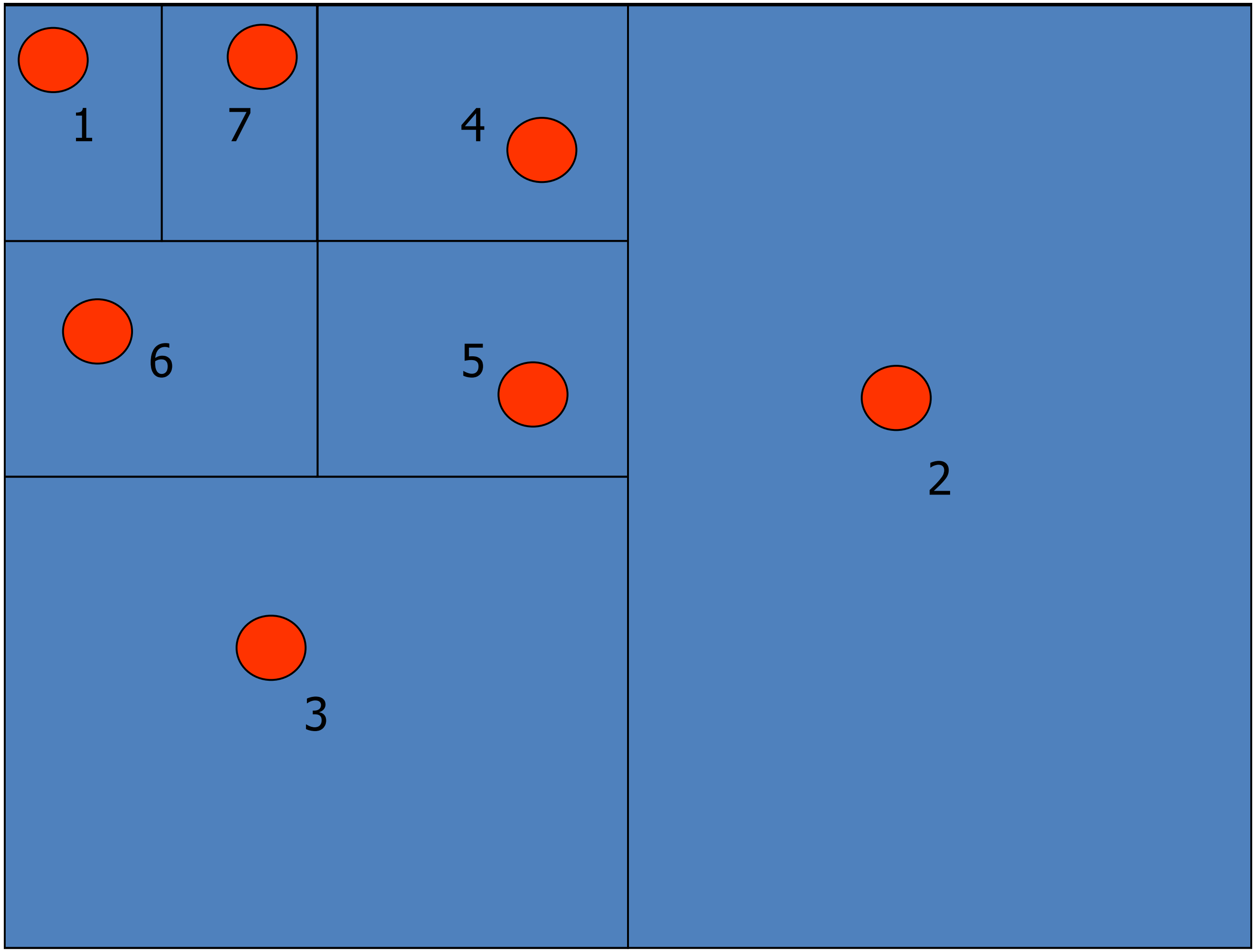
- A hash-based P2P file indexing and lookup scheme
- Decentralization
- While Gnutella, Freenet, Kazaa find data in $O(n)$ time, CAN can find data in $O(n^{1/d})$ time ($d > 1$)
- Source:
  - S. Ratnasamy, P. Francis, M. Handley, R. Karp, S. Shenker (UC Berkeley and ACIRI). "A Scalable Content-Addressable Network". ACM SIGCOMM, 2001

CAN: Zone and Key

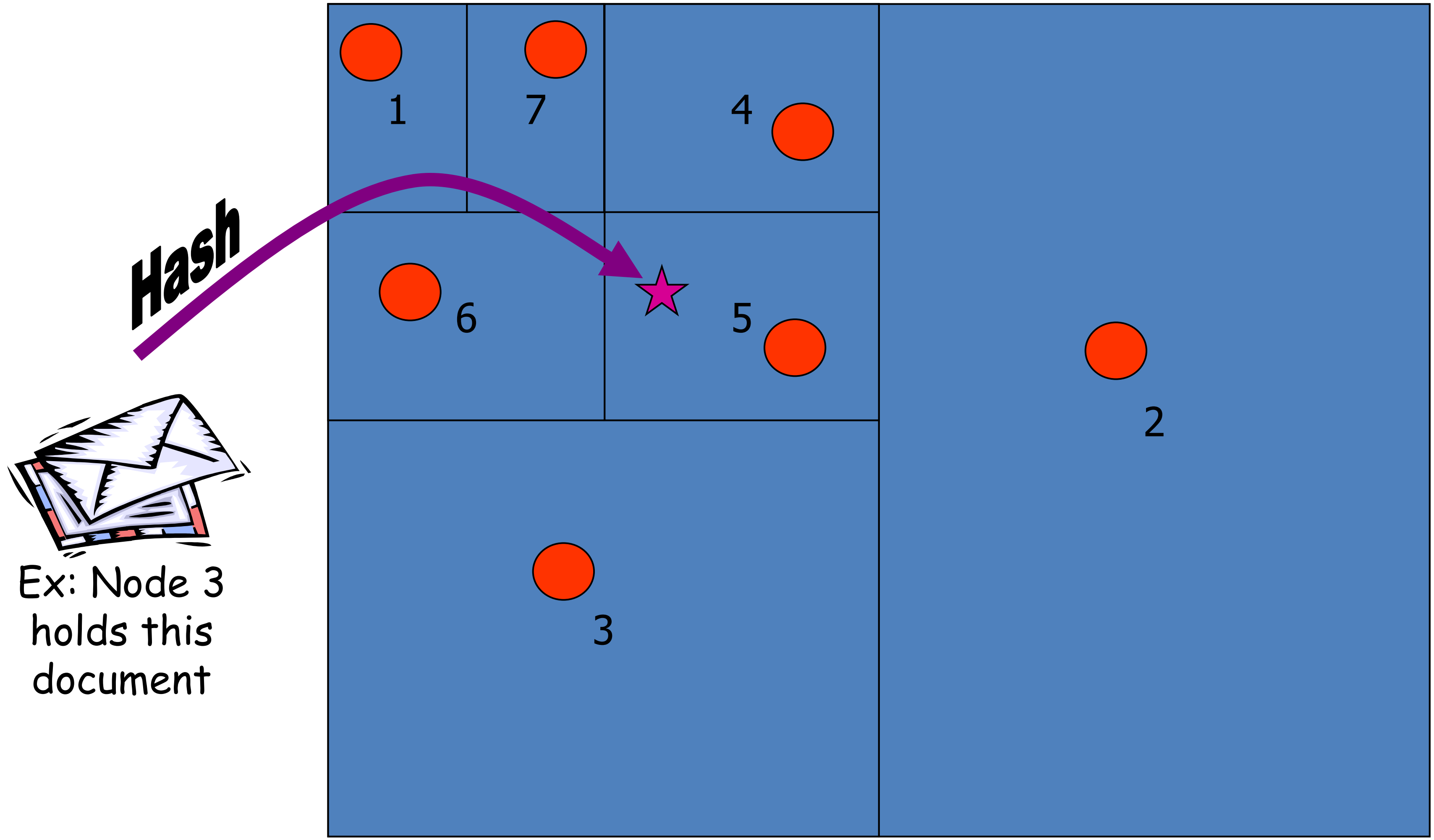
- Use a virtual *d*-dimensional coordinate space
  - $[0, 1]^d = [0, 1] \times [0, 1] \times .. \times [0, 1]$
  - called a d-torus
- A peer is mapped to a "zone" of this d-torus and said to "own this zone"
- Each file F is identified with key K_F
- A hash function h maps a key to a point in the d-torus $K \rightarrow (x_1, x_2, .., x_d) \in [0, 1]^d$, where $0 \le x_2 \le 1$.



Example



Association ID ⇨ node



Ex: Node 3 holds this document

CAN: Routing

- A CAN node maintains a coordinate routing table that holds the IP address and virtual coordinate zone of each of its immediate neighbors in d-torus (2d neighbors)
- d-torus is partitioned into n zones
- Using its neighbor coordinate set, a node routes a message towards its destination by simple greedy forwarding to the neighbor with coordinate closest to the destination coordinate.

CAN: routing algorithm

1. Start from some Node

2. P = hash value of the Key

3. Greedy forwarding

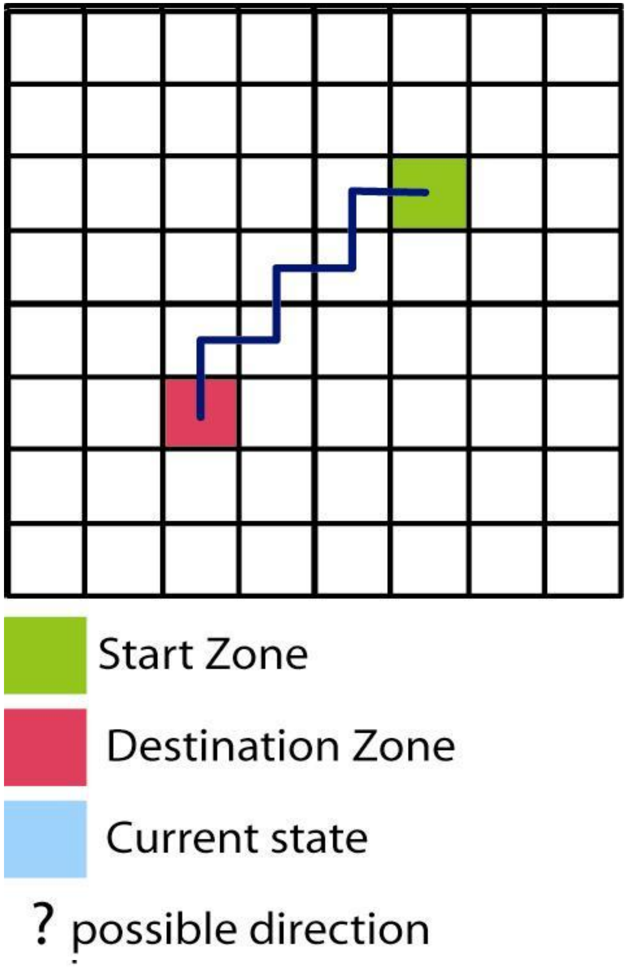
Current Node:

1. Checks whether it or its neighbors contain the point P

2. IF NOT
   a. Orders the neighbors by Cartesian distance between them and the point P
   b. Forward the search request to the closest one
   c. Repeat step 1

3. OTHERWISE

   The answer (Key, Value) pair is sent to the user



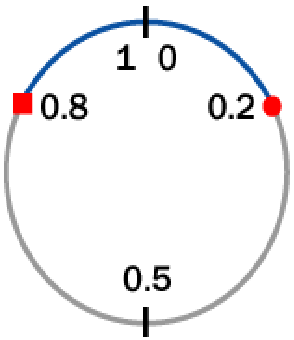
CAN: Routing Algorithm

If d-torus is partitioned into n equal zones, an average routing path goes through $(d/4)n^{1/d}$ hops, or $O(n^{1/d})$

why (d/4)n1/d hops

- Hash Table works on d-dimension Cartesian coordinate space on d-torus
  – Cyclical d-dimension Space

d-values hash function hash(K)=(x_1, …, x_d)



Example: 1-D torus

$$p1 = 0.2;\ p2 = 0.8$$

$$CartDist(p1,p2) = \sqrt{((p1\text{-}p2) \bmod 0.5)^2}$$

$$= \sqrt{(-0.6 \bmod 0.5)^2} = 0.4$$

why (d/4)n1/d hops

Example: 2-D torus

Average path length is average # hops to reach a destination node

In the case where:

1. All Zones have the same volume
2. There is no crashed node

Total path length = 0 * 1 + 1 * 2d + 2 * 4d + 3 * 6d + 4 * 7d + 5 * 6d + 6 * 4d + 7 * 2d + 8 * 1

6	5	4	3	4	5	6	7
5	4	3	2	3	4	5	6
4	3	2	1	2	3	4	5
3	2	1	0	1	2	3	4
4	3	2	1	2	3	4	5
5	4	3	2	3	4	5	6
6	5	4	3	4	5	6	7
7	6	5	4	5	6	7	8

why (d/4)n1/d hops

d-D torus

In the case where:

1. All zones have the same volume
2. There is no crashed Node

Total path length = 0 * 1 + 1 * 2d + 2 * 4d + 3 * 6d + 4 * 7d + 5 * 6d + 6 * 4d + 7 * 2d + 8 * 1

6	5	4	3	4	5	6	7
5	4	3	2	3	4	5	6
4	3	2	1	2	3	4	5
3	2	1	0	1	2	3	4
4	3	2	1	2	3	4	5
5	4	3	2	3	4	5	6
6	5	4	3	4	5	6	7
7	6	5	4	5	6	7	8

$$TPL = 0*1 + \sum_{i=1}^{\frac{n^{1/d}}{2}-1} i*2id + \frac{n^{1/d}}{2}*(n^{1/d}-1)d + \sum_{i=\frac{n^{1/d}}{2}+1}^{n^{1/d}} i*2(n^{1/d}-i)d + n^{1/d}*1$$

$$\text{Avg. path length} = \frac{\text{TPL (Total path length)}}{n(\#\text{of Nodes})} = d*\frac{n^{1/d}}{4}$$

CAN construction: New Node arrival 1

New Node, a server in the Internet wants to join the system and shares a piece of Hash Table.

1. New Node needs to get an access to the CAN
2. The system should allocate a piece of Hash Table to the New Node
3. New Node should start working in the system: provide routing

1. Finding an access point

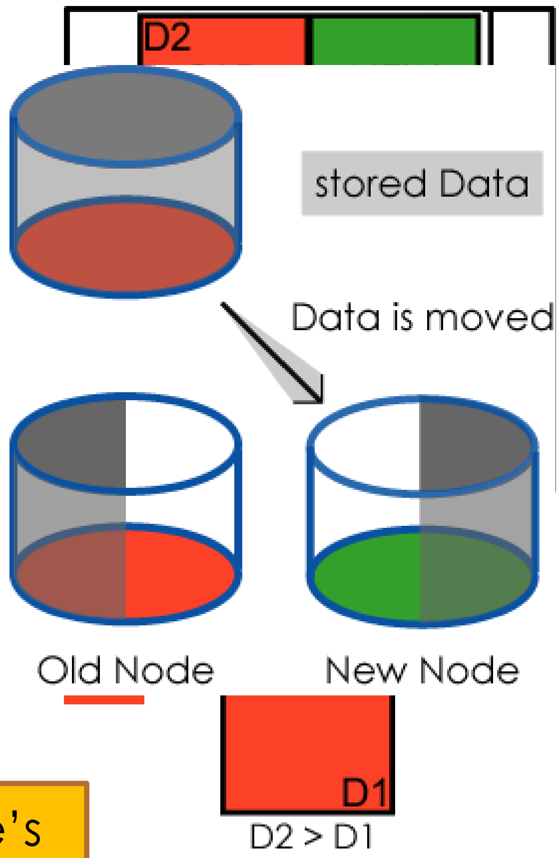
Sends a request to the CAN domain name

- Gets the IP address of one of the Nodes currently in the system
- Connects to this Node

CAN construction: New Node arrival 2

2. Finding a Zone

1. Randomly choose a point **P**
2. JOIN request is sent to the **P**-owner node
3. The request is forwarded via CAN routing
4. Desired node (P-owner) splits its Zone in half
   - One half is assigned to the New Node
   - Another half stays with Old Node
5. Zone is split along only one dimension: The greatest dim. with the lowest order
6. Hash table contents associated with New Node's Zone are moved from Old Node to the New Node



Tell me and I forget.
Show me and I remember.
Involve me and I understand.

Thank you!