Introduction

Mirek Riedewald

Key Learning Goals

- What are the three Vs of Big Data?
  - Can a 1 GB dataset cause a Big-Data problem?
- What do we mean by “data center as computer” and “warehouse-scale computer”?
  - Is this about an architecture or a programming model or both?
- Why does it matter if a program accesses data in local memory vs. local disk vs. disk/memory on a machine in the same rack vs. disk/memory on a machine “across the data center”?

Key Learning Goals

- What is the impact of Moore's Law for scalable Big-Data processing?
- Why do businesses migrate their computation to the Cloud?
- Why might some businesses decide to not move all their data and computation to the Cloud?
- Name major Cloud providers.

Why Parallel Data Processing?

- Answer 1: Big Data

What is “Big Data”?

- The term “Big Data” means different things to different people. It usually refers to (i) our ability to collect large amounts of data and (ii) the promise that analyzing this data will result in new insights, e.g., scientific discoveries, more effective governance, or better business decisions.
- While probably most people associate Big Data with high volume, e.g., petabytes, even comparably smaller data is Big Data if it is produced at high velocity or shows high variety. These are known as the **three Vs** of Big Data as introduced by Gartner [Laney, Douglas. "3D Data Management: Controlling Data Volume, Velocity and Variety". Gartner. Retrieved 6 February 2001].
- Intuitively, Big Data refers to problems where the data, due to sheer volume, the high rate at which it is generated, or its complexity, overwhelms traditional approaches for analyzing or even storing it.

How Much Data is Produced Annually?

- This is difficult to estimate accurately. Probably the most thorough analysis of the amount of data generated world-wide was undertaken in 2003 by a team at UC Berkeley [http://www2.sims.berkeley.edu/research/projects/how-much-info-2003/].
- While these numbers are by now outdated, they make an impressive case for Big Data analysis, especially when we take into account that our ability to generate and collect data has rapidly increased since then.
- Take a look at selected findings next (for details please consult the report).

How Much Information Report 2003

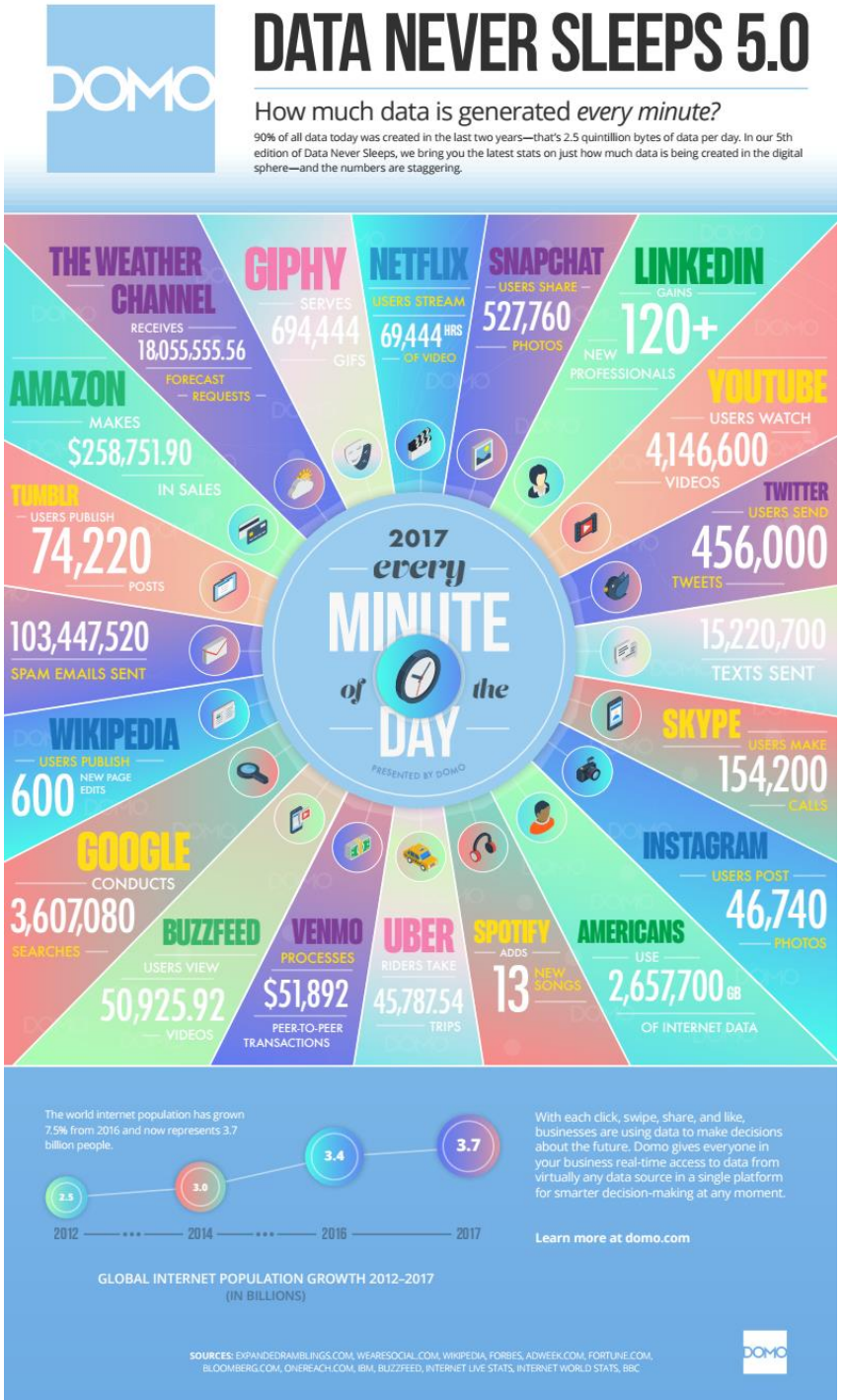
- Print, film, magnetic, and optical storage media produced 5 exabytes (10^{18} bytes) of new information in 2002.
  - This was equivalent to half a million times the size of all print collections of the US Library of Congress!
- New information stored on paper, film, magnetic, and optical media doubled between 2000 and 2003.
- Information flows through electronic channels—telephone, radio, TV, Internet—contained 18 exabytes of new information in 2002.

Stop and Think

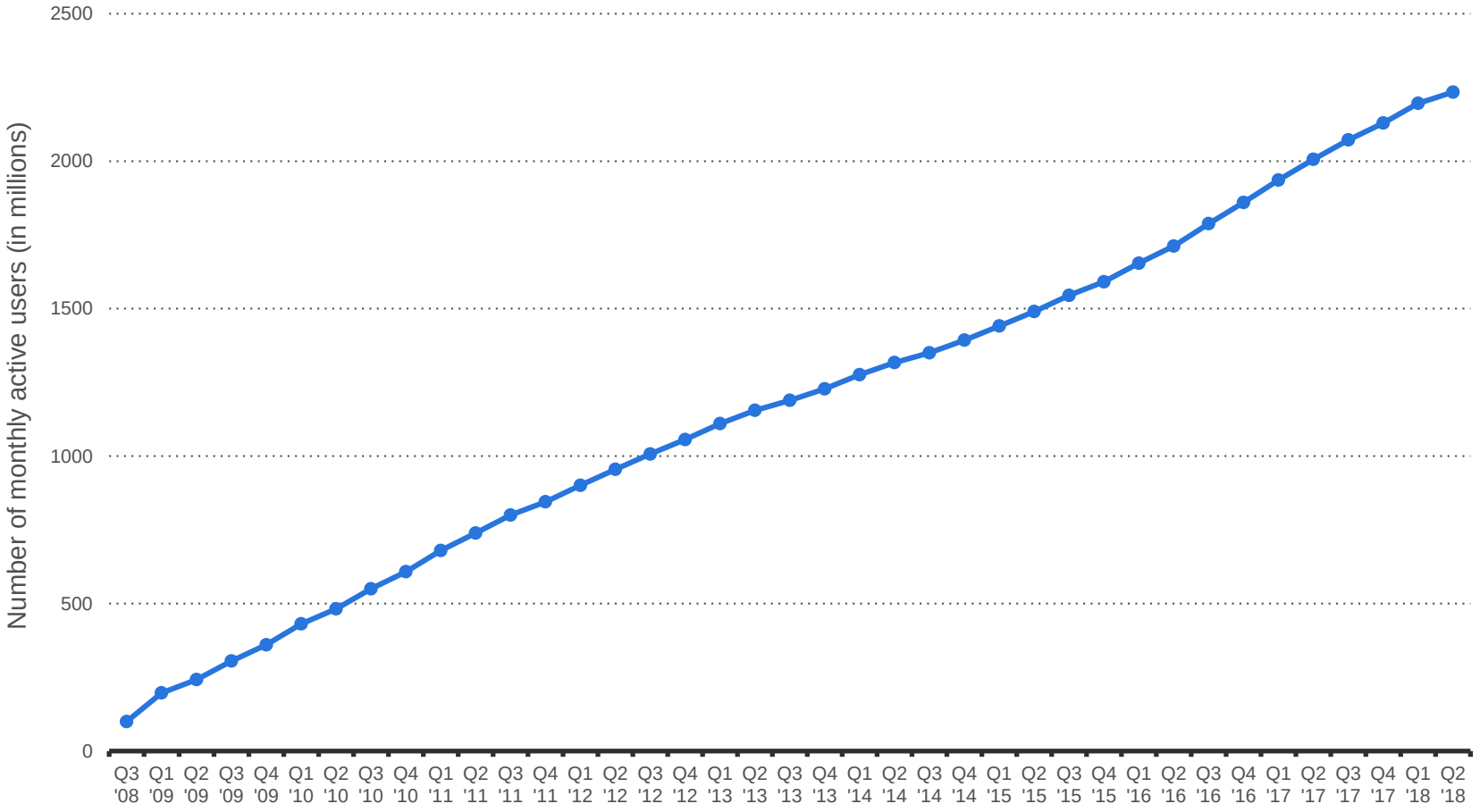
- How have things changed since 2003 and what might these numbers look like today?
- Which of the media do you think lost and which won market share?
- Can you think of new media that did not exist in 2003 or did not play a major role then?
- Which of the above numbers are affected significantly by the rise of tablets/smartphones and by social networks, e.g., Facebook?

Let us now look at some newer numbers.

Statistics Collected by domo.com



Facebook Growth



Facebook. n.d. Number of monthly active Facebook users worldwide as of 2nd quarter 2018 (in millions). Statista. Accessed August 14, 2018. Available from https://www.statista.com/statistics/264810/number-of-monthly-active-facebook-users-worldwide/

Social Network Analysis

- As a larger fraction of the world's population spends more time online, massive amounts of data about social interactions and opinions are produced, for example:
  - Homepages on the Web, personal pages in online social networks
  - Blogs, tweets, online product reviews, email
  - Companies and governments also collect data about site visits and purchases
- We can use this data to answer interesting questions about humanity and society:
  - How do my friends affect my own product reviews, purchases, or choice of friends? For instance, will the first reviews for a product set the tone for future reviews: if the others are positive, will my own review be biased to be more positive?
  - How does information spread and how is this affected by connections between people?
  - What are "friendship patterns"? A famous result is the *small-world phenomenon*, which intuitively states that even in a very large and sparse network, any two individuals are likely to be connected through a short sequence of acquaintances.
- One can also use it for other purposes, e.g., it helps governments identify dangerous organizations or simply spy on their citizen. And it helps companies make greater profits. For instance, search providers can place targeted ads based on user profiles, which they can build based on browsing history, email content, and purchase history.

Big Data in Business

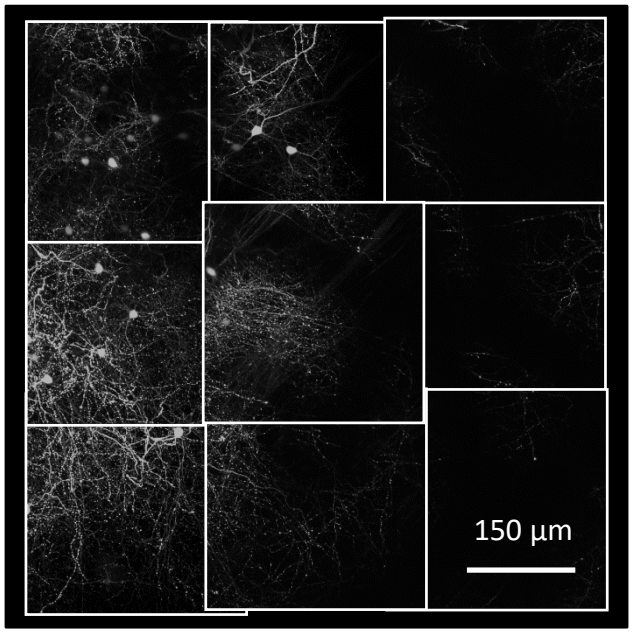
- Big Data also holds great promise for improving the way we do business as the following examples illustrate.
- Detecting fraudulent or criminal transactions for bank accounts, credit cards, and phone calls.
  - As companies collect data about legitimate and fraudulent transactions, they can construct models, e.g., using machine learning techniques. Given a new incoming transaction, these models can predict if it is legitimate.
  - In these scenarios, model training is often difficult due to data volume (high training time) and variety (making it difficult to identify actionable patterns). Keeping models up to date and making real-time predictions is challenging due to data velocity as transactions are coming in at high rates.
- Retail companies want to encourage customers to buy certain products. They analyze previous purchase behavior to find out which products are frequently bought together and which promotions will be most effective. Established data mining techniques such as association rule and frequent itemset mining are often used, but struggle to scale to Big Data.
- For marketing purposes, search companies and online social networking sites want to determine which ads to place based on search query, user interest, and other data. For instance, notice how after you search for some product, you will often see related advertisements when visiting other Web pages.
- The same data can also be used to identify key groups of customers and what defines each group. This is often done by finding clusters of similar users who are different from users in other clusters.

Data-Driven Science Examples

- The human genome has about 3 billion base pairs—now imagine how much data we have to manage and analyze for the population of Boston, the US, or India/China.
- The Large Hadron Collider (LHC) in Europe is the world's largest and most powerful particle accelerator. It was designed to perform high-energy physics experiments to answer questions related to our understanding of the Universe, e.g., verify the existence of the famous Higgs boson. The LHC produces about 15 petabytes of raw data annually. [http://home.web.cern.ch/about/computing accessed November 18, 2013]
- The Sloan Digital Sky Survey's goal is to "map the universe". Between 2000 and 2008, it obtained deep, multi-color images covering more than a quarter of the sky and created 3-dimensional maps containing more than 930,000 galaxies and more than 120,000 quasars. [http://www.sdss.org/ accessed November 18, 2013]
- Citizen science projects such as eBird (http://ebird.org/content/ebird/) are collecting and integrating tens of millions of reports about bird sightings from all over the world. Joining these reports based on time and location with data about climate, weather, human census information, habitat, elevation, and other features creates a large and rich data resource for exploring how bird populations behave and how changes might be related to climate and other factors.

Science Highlight: NCTracer

DATALAB
@Northeastern



- Prof. Riedewald and his students at the DATA Lab collaborate with scientists—Prof. Stepanyants' group in COS—on a project whose goal is to map connectivity in the brain.
- A mouse-brain scan produces about 20 terabytes of raw image data, which has to be cleaned, aligned, and turned into a graph.
- Then we want to identify interesting structural patterns and pattern changes over time as the mouse learns. (For the latter, data covers only outer brain layers in a limited region of the skull.)

Why Parallel Data Processing?

- Answer 1: Big Data
- Answer 2: hardware trends
  - Multi-core CPUs and GPUs

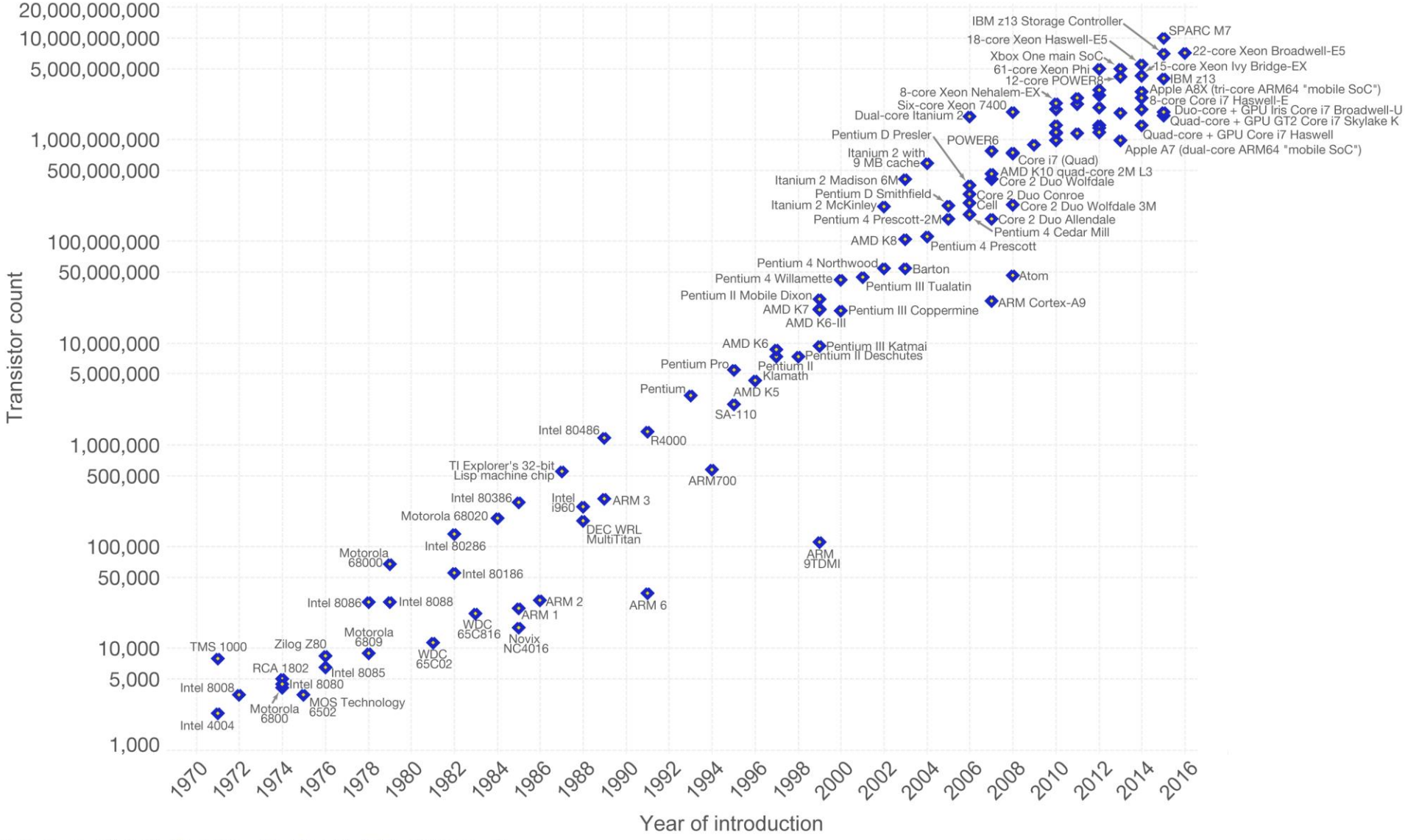
The Good Old Days: Moore's Law

- Moore's Law states that the number of transistors that can be placed inexpensively on an integrated circuit doubles about every 2 years. More transistors on a chip implies more computational resources.
- Until the early 2000s, this also meant that existing sequential programs became automatically faster at a similar rate. As a rule of thumb, you essentially just had to wait for 2 years and your program would run twice as fast.
- Hence there was little motivation to try and master the complex art of parallel programming. Parallel computing never entered the mainstream and was limited to (high-impact) niche applications, e.g., military tasks, high-energy physics, and weather forecasting.

Moore's Law – The number of transistors on integrated circuit chips (1971-2016)

Our World in Data

Moore's law describes the empirical regularity that the number of transistors on integrated circuits doubles approximately every two years. This advancement is important as other aspects of technological progress – such as processing speed or the price of electronic products – are strongly linked to Moore's law.

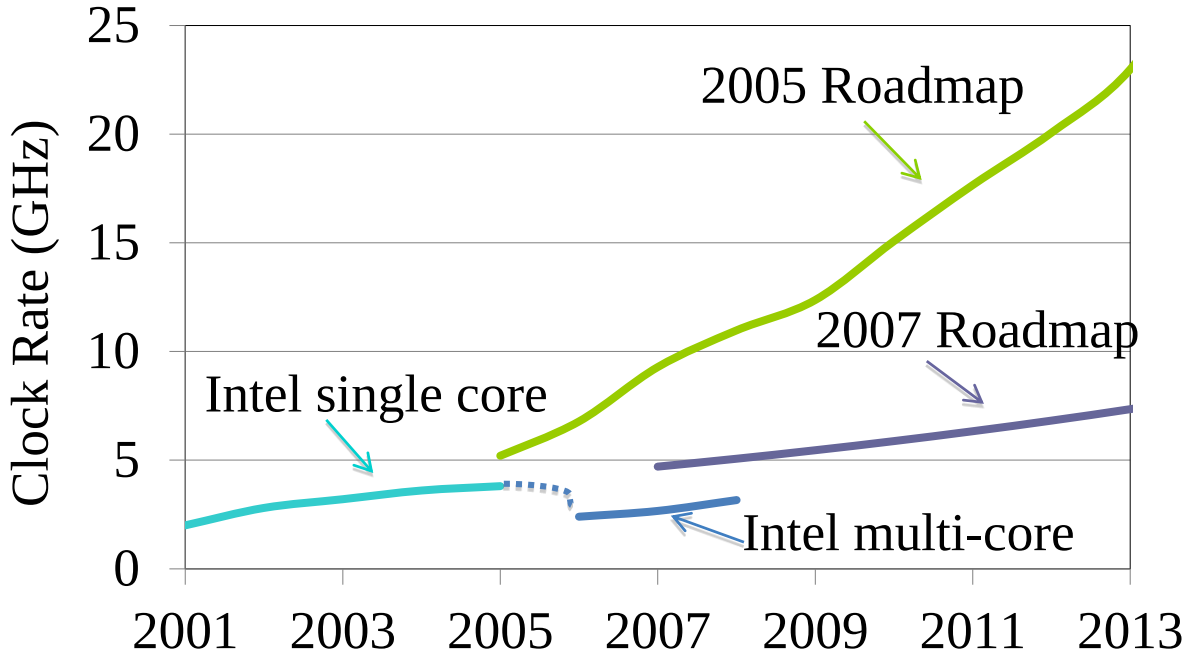


Data source: Wikipedia (https://en.wikipedia.org/wiki/Transistor_count)

The data visualization is available at OurWorldinData.org. There you find more visualizations and research on this topic.

“New” Realities

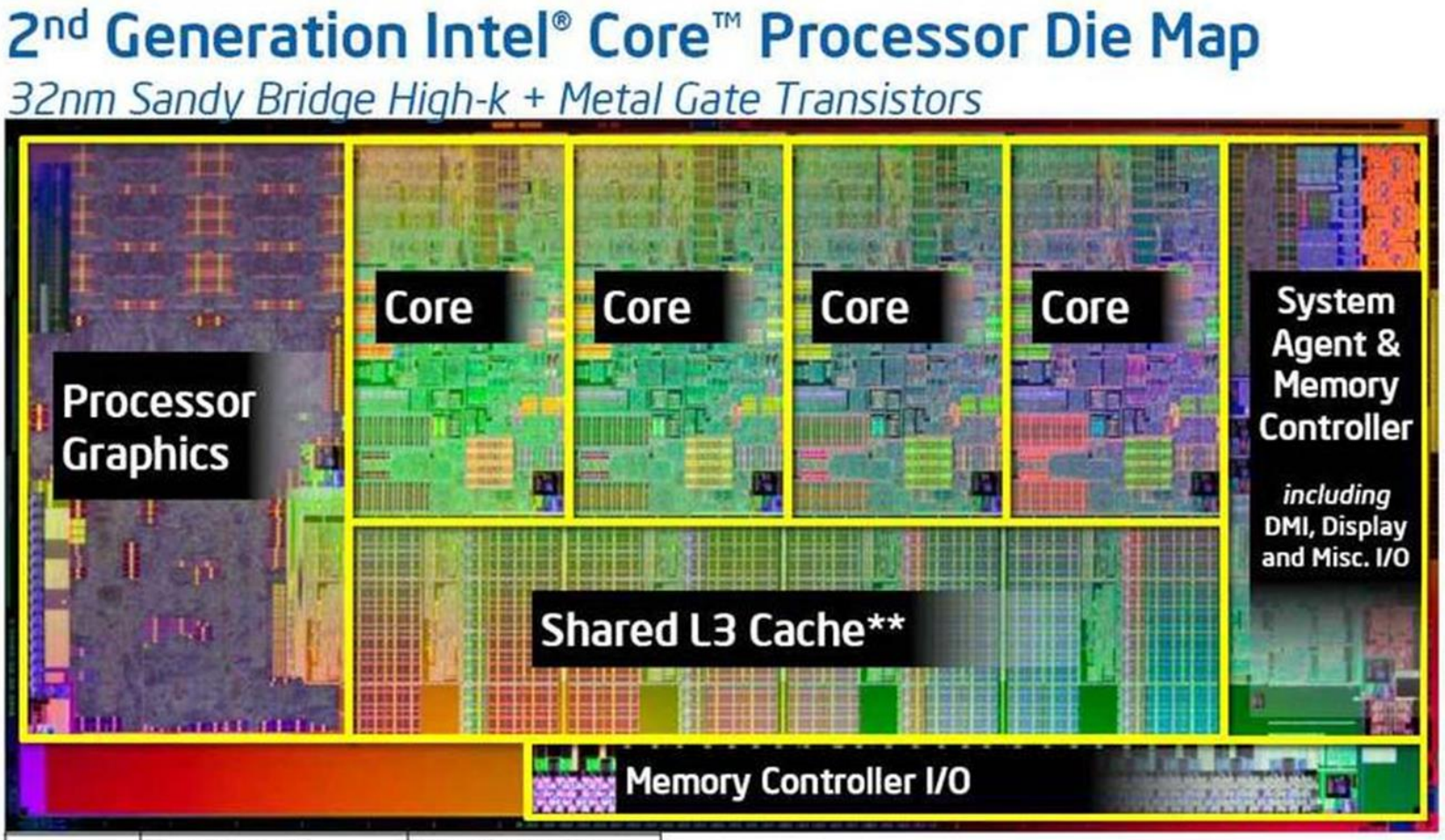
- The “party” ended around 2004, when heat and energy-related issues prevented higher clock rates. Essentially CPU clock rate has remained below 4 GHz since 2005.
- The graph below shows Intel’s initial 2005 roadmap for CPU clock rates. As problems were discovered, Intel corrected the roadmap in 2007. However, even the corrected roadmap proved overly optimistic. [Source: Dave Patterson, UC Berkeley]
- In the end, multi-core CPUs emerged as the preferred way to leverage the still-increasing transistor density while avoiding cooling problems. (Note: The are other solutions such as water-cooled CPUs. However, it is unlikely that end-users will see water-cooled CPUs in their commodity machines any time soon.



Multi-Core CPUs

- A multi-core CPU consists of several “mini-CPUs” on the same chip, called cores.
- Cores typically share some cache, the memory bus, and access to same main memory.
- To take advantage of Moore’s Law, a program now needs to keep multiple cores busy to exploit the additional transistors on a chip. To do so, we either need to run multiple applications or applications that are multi-threaded.
  - Unfortunately it is not easy to re-write existing software and make it multi-threaded.

Processor Example [Source: Intel]



Die	Number of Transistors (mio)	Die size with Scribe (mm2)
4+2	995	216
2+2	624	149
2+1	504	131

** Cache is shared across all 4 cores and processor graphics

Typical Multi-Core Properties

- Each core has some local cache (e.g., L1, L2)
- The cores share some cache (e.g., L3)
- All cores access same memory through bus
- Misses become much more expensive from L1 to L3, even more when accessing memory

- Beyond L3, access times grow rapidly as we show next.

Important Numbers [Source: Google's Jeff Dean @LADIS'09]

L1 cache reference	**0.5**
Branch mispredict	5
L2 cache reference	**7**
Mutex lock/unlock	25
Main memory reference	**100**
Compress 1 KB with Zippy	3,000
Send 2 KB over 1 Gbps network	20,000
Read 1 MB sequentially from memory	**250,000**
Round trip within same data center	500,000
Disk seek	**10,000,000**
Read 1 MB sequentially from disk	**20,000,000**
Send packet CA -> Holland -> CA	150,000,000

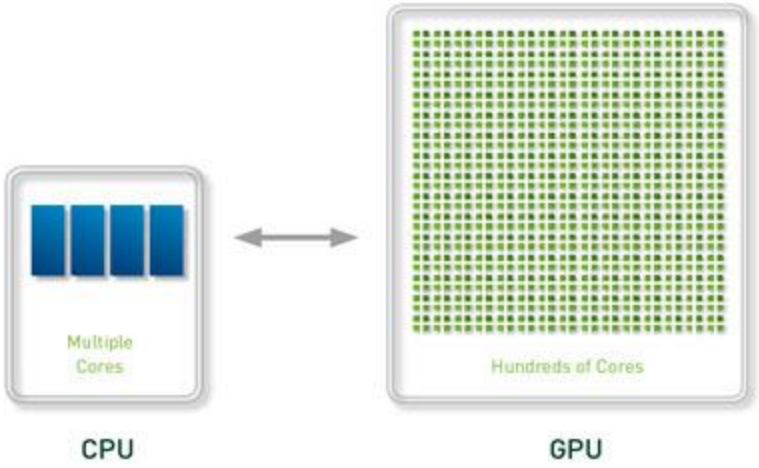
All times in ns.

Discussion of the Numbers

- The numbers on the previous page illustrate how dramatically access time increases for data located further "away" from a core:
  - It takes about ten times longer to access L2 cache than L1 cache.
  - If a record is not in cache and has to be fetched from memory, it takes even 200 times longer than L1 cache.
  - Accessing a data record on a traditional spinning hard disk takes another 100,000 times longer than a memory reference.
  - If data is accessed in a data center across the globe, wait time increases by another factor of 15.
- However, notice that the numbers discussed so far refer to the latency of access, i.e., the time between a core issuing a request for a data record until it arrives at the core. When transferring larger blocks of data, the data rate of the channel used for data transfer will also affect the time it takes until the entire block has been transferred.
  - For instance, as the numbers in the table indicate, it takes about 10,000,000 nanoseconds for the disk to position its head over the first relevant disk sector, no matter how much data will be read from there. Reading 1 MB sequentially from disk takes 20,000,000 nanoseconds: 10,000,000 nanoseconds to position the head, then another 10,000,000 nanoseconds to read the next 1 MB's worth of data. (In other words, the example disk has a latency of 10 milliseconds and a transfer rate of 100 MB per second.)

GPU vs. CPU

- GPUs are multi-core CPUs optimized for massively parallel graphics processing. They tend to have many more cores than a CPU, but each core is simpler and supports a comparably limited set of operations.
- Due to the popularity of computer games, GPUs are commodity chips with an excellent performance-to-price ratio. Hence they attracted interest in academia and industry to use them for general computations beyond the graphics-related tasks they were designed for.
- While many graphics-processing tasks are easy to parallelize, for general-purpose computational use it is challenging to write code that effectively uses 100s of "simple" cores. Parallel computing platforms and programming models were proposed to address this challenge, e.g., NVIDIA's CUDA.



Source: NVIDIA

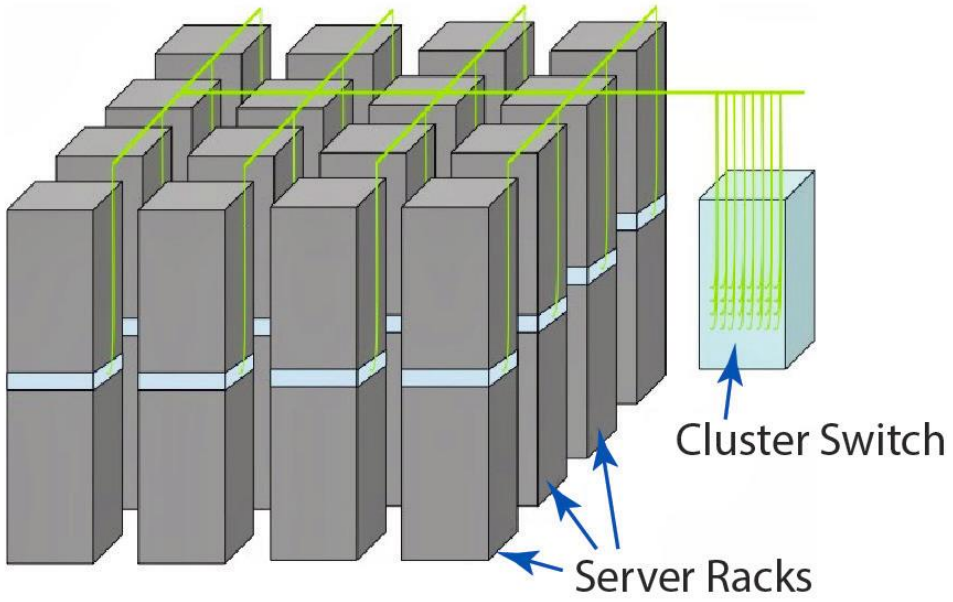
Why Parallel Processing?

- Answer 1: Big Data
- Answer 2: hardware trends
  - Multi-core CPUs and GPUs
  - Data center as a computer

Data Center as a Computer

- Data Center as a Computer, also known as **Warehouse-Scale Computer (WSC)**, is the most relevant hardware-driven trend that motivated Google to propose MapReduce.
- Large companies such as Google, Microsoft, Facebook, and Amazon are already managing hundreds to tens of thousands of servers in their data centers. These servers tend to be commodity PCs. By relying on commodity hardware, companies achieve a better cost per unit of computational capability than specialized hardware due to economies of scale. It is also easy to "scale out" by adding more machines as demand increases.
- Once we have access to thousands of machines that are linked to each other via high-speed networks, the question arises how to employ all these machines to solve Big Data analysis problems. The idea behind Data Center as a Computer and Warehouse-Scale Computer is to create an environment that enables programming of these large clusters of machines as if they were a single huge computer.

Basic Architecture

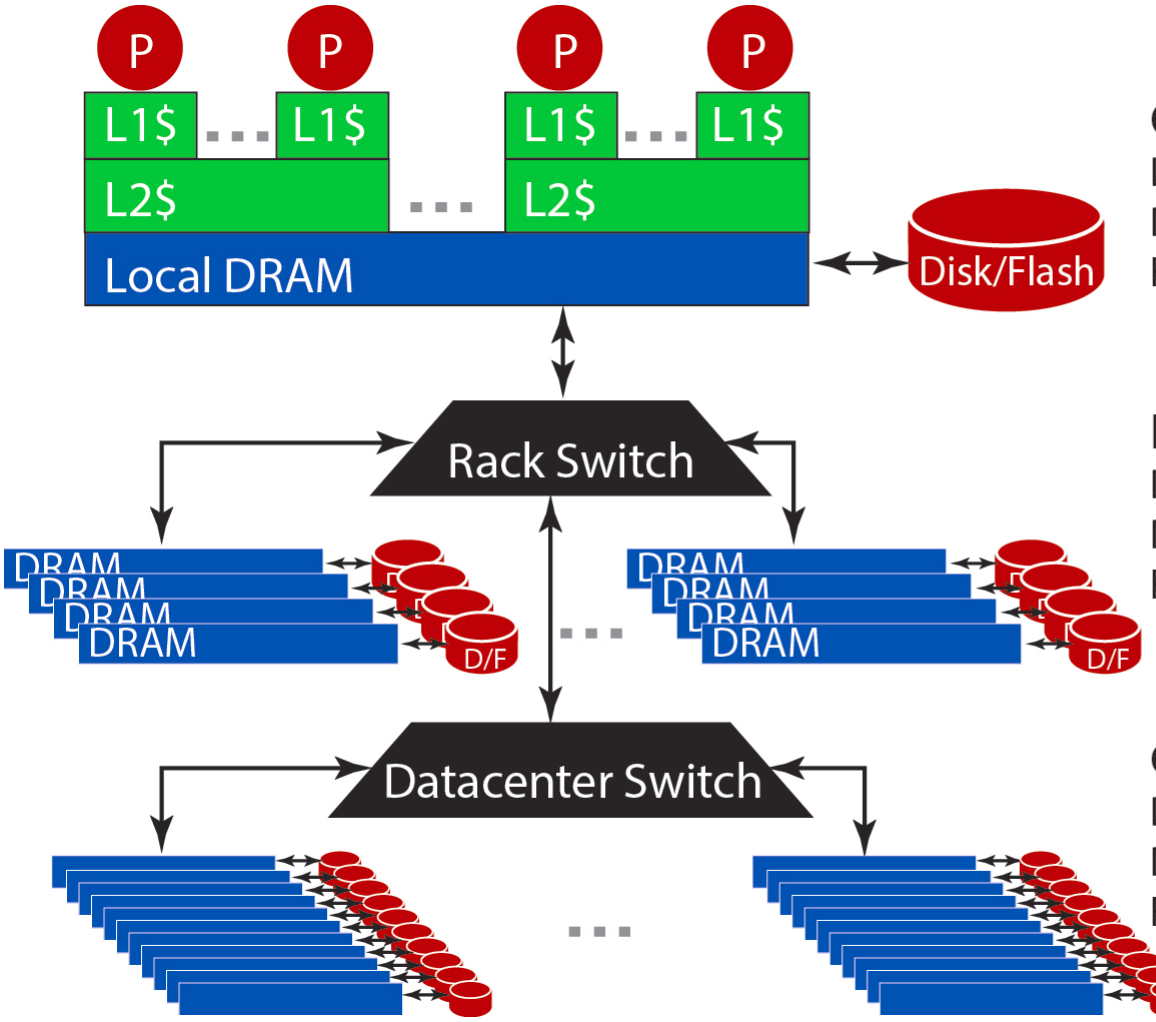


Source: Barroso, Clidaras, Holzle (2013)

Typical Specs (Barroso, Clidaras, Holzle (2013))

- Each individual machine is a low-end server in a 1U ("single unit" of height 1.75 inches/44.45 mm) enclosure in a 7' rack.
- All machines in a rack are connected to a rack-level switch with 1- or 10-Gbps links
- The different rack-level switches are connected by one or more cluster switches. Overall, more than 10,000 servers could be connected with each other this way.
- There are local (cheap) disks on each server. Disk space across machines is managed by a global distributed file system. (Discussed in another module)
- There might also be Network Attached Storage (NAS) devices for a more centralized storage solution.

Storage Hierarchy



One Server
DRAM: 16 GB, 100 ns, 20 GB/s
Disk: 2T B, 10 ms, 200 MB/s
Flash: 128 GB, 100 us, 1 GB/s

Local Rack (80 servers)
DRAM: 1 TB, 300 us, 100 MB/s
Disk: 160 TB, 11 ms, 100 MB/s
Flash: 20 TB, 400 us, 100 MB/s

Cluster (30 racks)
DRAM: 30 TB, 500 us, 10 MB/s
Disk: 4.80 PB, 12 ms, 10 MB/s
Flash: 600 TB, 600 us, 10 MB/s

Source: Barroso, Clidaras, Holzle (2013)

Storage Hierarchy Discussion

- A single server's specs look very similar to those of a laptop a student might own. There are 16 GB of main memory (DRAM) that have an access latency of 100 nano seconds. Data chunks from memory can be transferred to the local CPU at a rate of 20 GB/sec. Traditional rotating hard disks can provide up to 2 TB of storage, but latency is 100,000 times slower and bandwidth for data transfer is about 100 times lower than for memory. Flash offers a "middle ground" for non-volatile storage: latency and bandwidth are "only" 1000 and 20 times slower, respectively, than memory.
- Instead of reading data only from devices on the same server, a CPU could also pull it from another machine in the same rack or even a different rack somewhere across the same data center. As the numbers indicate, the further away the data, the higher the latency and the lower the bandwidth. Interestingly, since the network switches and links determine the maximal possible bandwidth for transferring data from a different machine, these numbers are identical for memory, disk, and flash. Also notice that the combined memory in the same rack is similar to the disk capacity of a single machine. However, access latency is better, while bandwidth is worse compared to the local disk.

Programming WSCs

- To achieve the greatest possible performance, an application should be aware of these tradeoffs. Unfortunately, asking a programmer to optimize the code "manually" is impractical. Furthermore, code optimized for one cluster would have to be rewritten for another with different parameters. On the other hand, completely ignoring these numbers and simply programming for a giant machine with 30 TB of memory and multiple PB of disk space would most likely result in poor performance. Just moving a terabyte of data between machines would take a huge amount of time.
- Several aspects are relevant for striking the right balance between high program performance and low programming complexity for WSCs.

WSC Programming Principles

1. A WSC should consist of relatively homogenous hardware and should have a system software platform with a common system management layer. The common infrastructure and services hide architectural complexity from the programmer.
2. Since the servers are commodity machines, they tend to fail. With thousands of machines, failures are the norm, not the exception. Those failures should be handled transparently by the infrastructure. This avoids cluttering user code with failure handlers.
3. The programmer needs a simple way for keeping thousands of CPUs busy, doing productive work. It would be too complex to write code that explicitly manages the behavior of individual machines. Instead, the programmer should be able to focus on the algorithm. Task scheduling and resource management should be provided by the WSC infrastructure. Similarly, programs running on a WSC need to be aware of data location, but the programmer should not have to reason about data locations when writing a program.

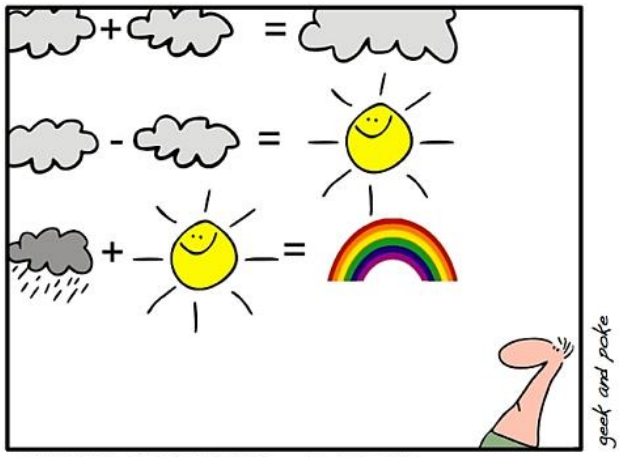
This is where systems like Hadoop MapReduce and Spark really shine! They simplify parallel programming, but still achieve high performance when executing a job in parallel on tens to thousands of machines.

More and more, parallel data processing is performed in the “Cloud”. What is the “Cloud”?

Some content based on [Brian Hayes. Cloud Computing. *Communications of the ACM* 51(7), 2008]

Cloud Overview

- There are many variations on cloud computing, but the underlying idea is always the same: instead of doing it locally, offload some computation or service to a remote provider. The cloud user accesses these remote resources and services without knowing details about underlying hardware. There are many related concepts, e.g., on-demand computing, software as a service (SaaS), and Internet as platform.
- As Cloud offerings gain popularity, they start to replace traditional “shrink-wrap” software, e.g., compare MSFT Word on a desktop PC vs. Google Docs or Office 365. This goes along with a change in pricing models. Instead of purchasing the software license once and then using the software “for free,” Cloud users often have no initial cost but pay for usage.



SIMPLY EXPLAINED – PART 17:
CLOUD COMPUTING

Cloud Computing Variants

There are three major classes of Clouds and Cloud usage (for all examples, note that Cloud offerings keep changing. Check which ones are still around and which others have been added):

1. Reserve virtual machines to create a virtual cluster. Then run your own application(s) there.
   - Examples: Amazon Web Services (including Elastic MapReduce), IBM SmartCloud, Google App Engine, Microsoft Azure, Force.com
2. Connect through a Web browser to an existing application running in the Cloud.
   - Examples: Google Docs, Acrobat.com, Microsoft Office 365
3. Build your own application on top of services offered by a Cloud provider. Typical services include database, document management, Web design, workflow management, and data analytics. This third class of Cloud usage is between the first two "extreme" points.
   - Examples: Salesforce.com, Microsoft Dynamics, IBM Tivoli Live

Cloud History: Back to the Future?

- In many ways, it seems as if Cloud computing is a step back to the 1960s world of hub-and-spoke mainframe configurations. Then “dumb” terminals were used to access a mainframe machine through phone lines.
- This all changed in the 1980s when the client-server model let PCs take over functionality and data from the mainframe. Companies such as SAP who realized this trend early on and designed their business software accordingly became global powerhouses.
- So, why is it suddenly in fashion again to go back to an old model?

Or Not Back to the Future?

- Despite the obvious similarities, the Cloud is not the same as a 1960s hub. First, clients can communicate with many servers at the same time and are more powerful than the old terminals. Second, servers also can communicate with each other.
- Still, with Cloud computing, functionality migrates away to data centers. Storage, computing, high bandwidth, and careful resource management are characteristics of the core, while end users initiate requests from the fringe.

Driving Forces Behind Cloud Computing

- Clouds can leverage economies of scale in several ways:
  - Software installation, configuration, and maintenance become easier in a homogeneous environment.
  - Less personnel is needed for hardware maintenance as well, compared to each company running their own tech department.
  - A single instance of a text processor or spreadsheet program cannot utilize 100 cores, but 100 instances can!
- The Cloud enables elasticity, i.e., the ability to grow and shrink capacity on demand. In many businesses there is a great difference between average and peak demand. Sizing infrastructure for peak demand is a waste of resources for most of the time. On the other hand, not being able to handle the load could result in lost business opportunity or customers. (Just imagine if you wanted to buy a product from Amazon and the Web site says “sorry, you cannot complete the purchase because our servers are overloaded.”) With Cloud computing it becomes comparably easy to deal with this issue: there is no major initial investment cost and the business pays according to usage.

Cloud Challenges

- Scalability: Cloud providers have to support many users and many applications
- Complex interactions: a client might invoke programs on multiple servers and each server might interact with multiple clients.
- A browser is limited in functionality compared to a traditional operating system. Hence applications accessed through a browser are inherently limited in the way the interact with the user and manage data.
- Despite all economies of scale, a Cloud provider might have to deal with a more heterogeneous environment than any single company. It might have to support a database backend running SQL, and JavaScript and HTML at the client. A server app might be written in PHP, Java, or Python.

The Cloud's Biggest Problem

- Arguably the biggest challenges for Cloud computing are related to privacy, security, and reliability.
  - What happens if the service is not accessible?
  - Who owns the data? Will the Cloud user lose access to the data if the bill is not paid?
  - Can the Cloud provider guarantee that deleted documents are really gone?
  - How well does the Cloud provider protect the data, e.g., against leaking to a third party or against government access? For instance, if software by competing companies runs on the same physical machines, could an infrastructure bug allow data to be shared? Would a Cloud provider contest government requests for private data as much as the company whose customers are affected? With events such as the revelations by Edward Snowden, such questions are more relevant than ever.

References

- Barroso, L.A., Clidaras, J., Hölzle, U.: The Datacenter as a Computer: An Introduction to the Design of Warehouse-Scale Machines. Synthesis Lectures on Computer Architecture **8**(3), 1—154 (2013)
  - https://scholar.google.com/scholar?cluster=16215025499256569000&hl=en&as_sdt=0,22