Distributed System Models and Enabling Technologies

Ioan Raicu
Computer Science Department
Illinois Institute of Technology

CS 553: Cloud Computing
September 8th, 2014

The Age of Internet Computing

- Computing systems distinctions
  - Centralized Computing
  - Parallel Computing
  - Distributed Computing
  - Cloud Computing
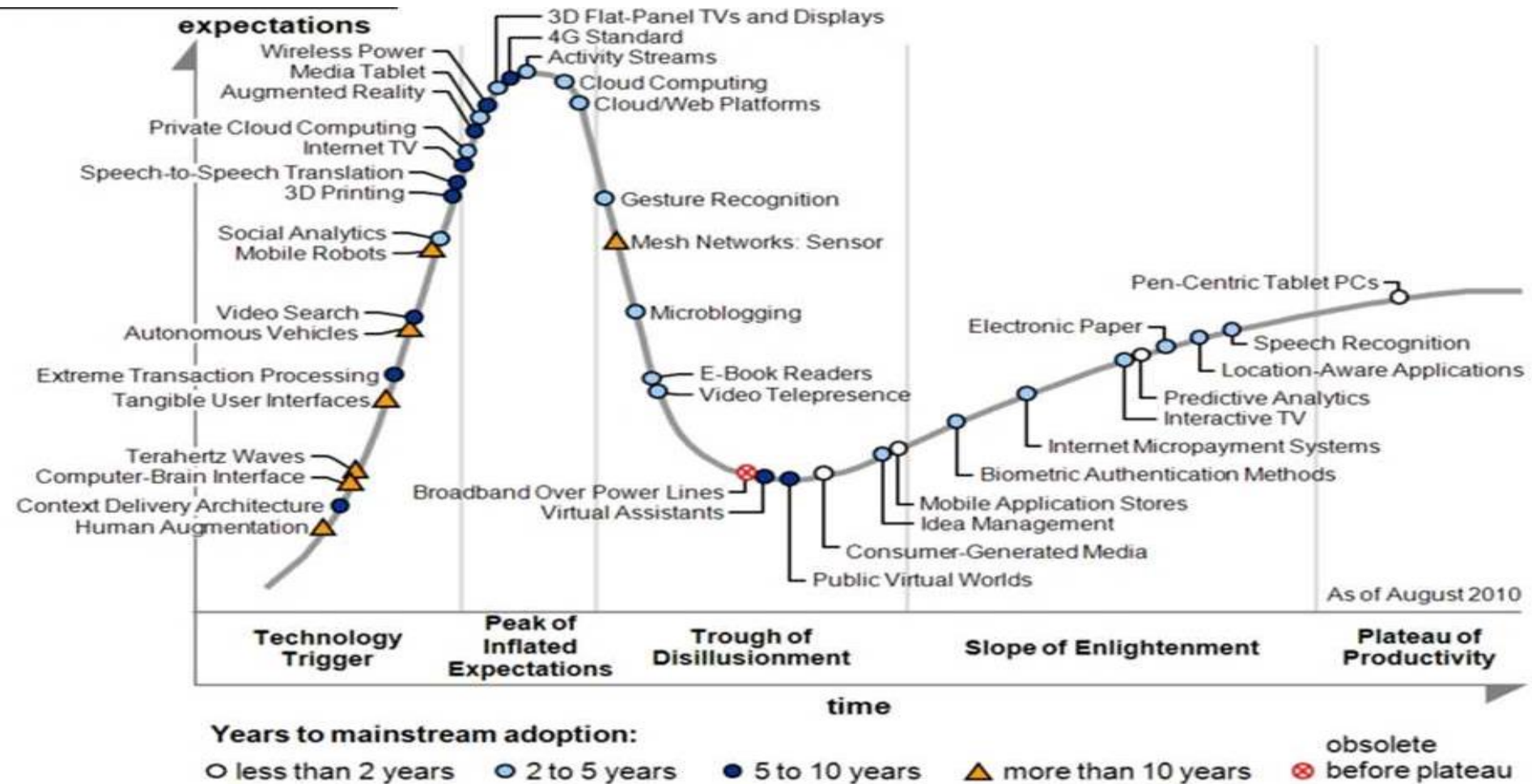- There exists significant overlap among all these

Parallel Computing Trends and New Paradigms: Degree of Parallelism

- *Bit-level parallelism* (BLP)
  - converts bit-serial processing to word-level processing gradually

- *Instruction-level parallelism* (ILP)
  - execute multiple instructions simultaneously

- *Data-level parallelism* (DLP)
  - Enabled and popularized by SIMD (*single-instruction and multiple-data*) and vector machines using vector or array types of instructions

- *Task-level parallelism* (TLP)
  - Introduced with multicore processors and *chip multiprocessors* (CMP)

- *Job-level parallelism* (JLP)
  - Parallel processing ➔ distributed processing, we will see an increase of computing granularity
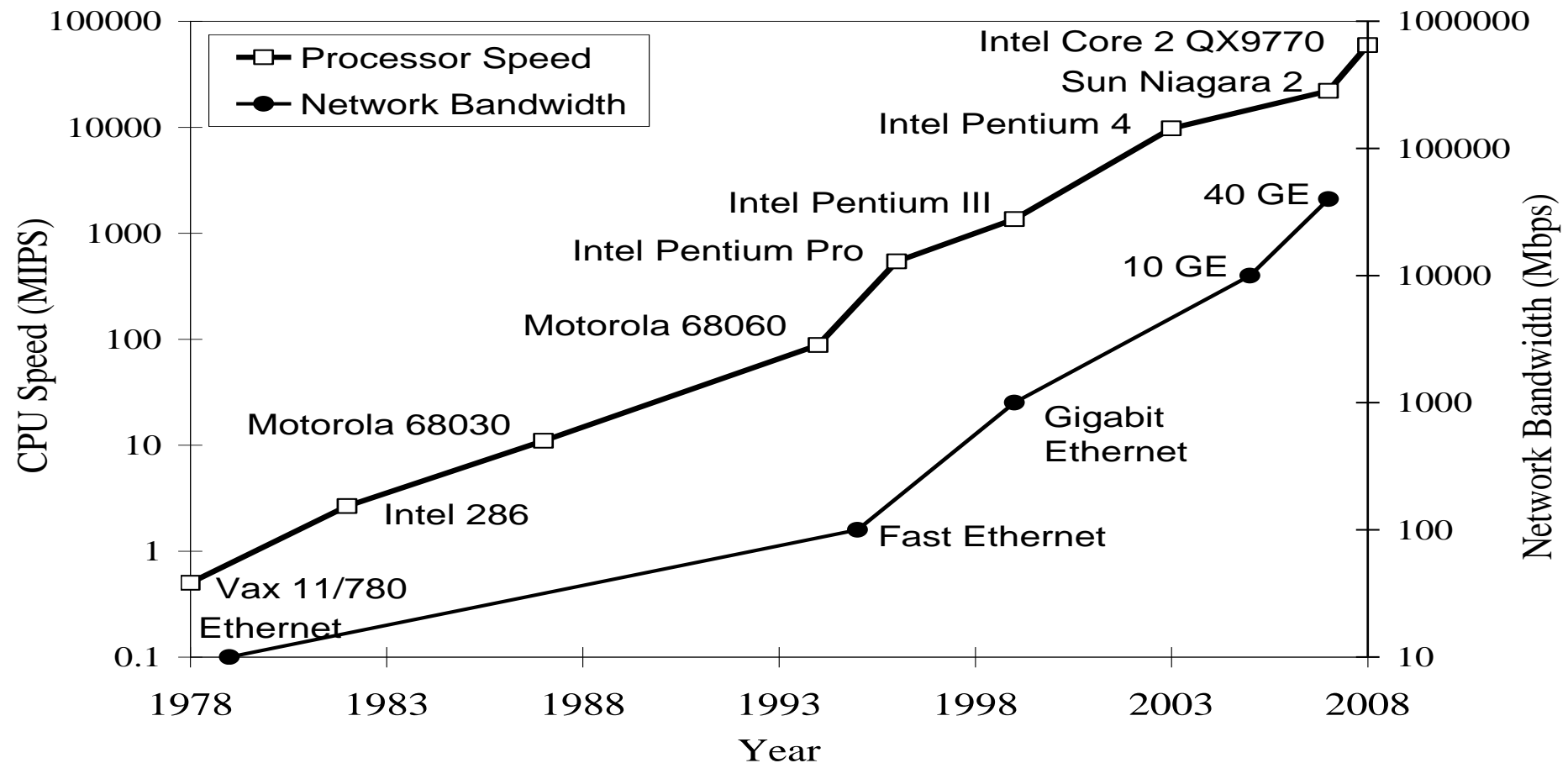
Parallel Computing Trends and New Paradigms: Applications

Domain	Specific Applications
Science and Engineering	Scientific simulations, genomic analysis, etc.
	Earthquake prediction, global warming, weather forecasting, etc.
Business, Education, service industry, and Health Care	Telecommunication, content delivery, e-commerce, etc.
	Banking, stock exchanges, transaction processing, etc.
	Air traffic control , electric power Grids, distance education, etc.
	Health care, hospital automation, telemedicine, etc.
Internet and Web Services and Government Applications	Internet search, datacenters, decision-make systems, etc.
	Traffic monitory , worm containment, cyber security, etc.
	Digital government, on-line tax return, social networking, etc.
Mission-Critical Applications	Military commend, control, intelligent systems, crisis management, etc.

Parallel Computing Trends and New Paradigms: Hype Cycle

expectations

- 3D Flat-Panel TVs and Displays
- 4G Standard
- Activity Streams
- Wireless Power
- Media Tablet
- Augmented Reality
- Cloud Computing
- Cloud/Web Platforms
- 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
- Microblogging
- Electronic Paper
- Autonomous Vehicles
- Speech Recognition
- Location-Aware Applications
- Extreme Transaction Processing
- E-Book Readers
- Video Telepresence
- Predictive Analytics
- Interactive TV
- Tangible User Interfaces
- Terahertz Waves
- Internet Micropayment Systems
- Computer-Brain Interface
- Biometric Authentication Methods
- Broadband Over Power Lines
- Context Delivery Architecture
- Virtual Assistants
- Mobile Application Stores
- Human Augmentation
- Idea Management
- 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
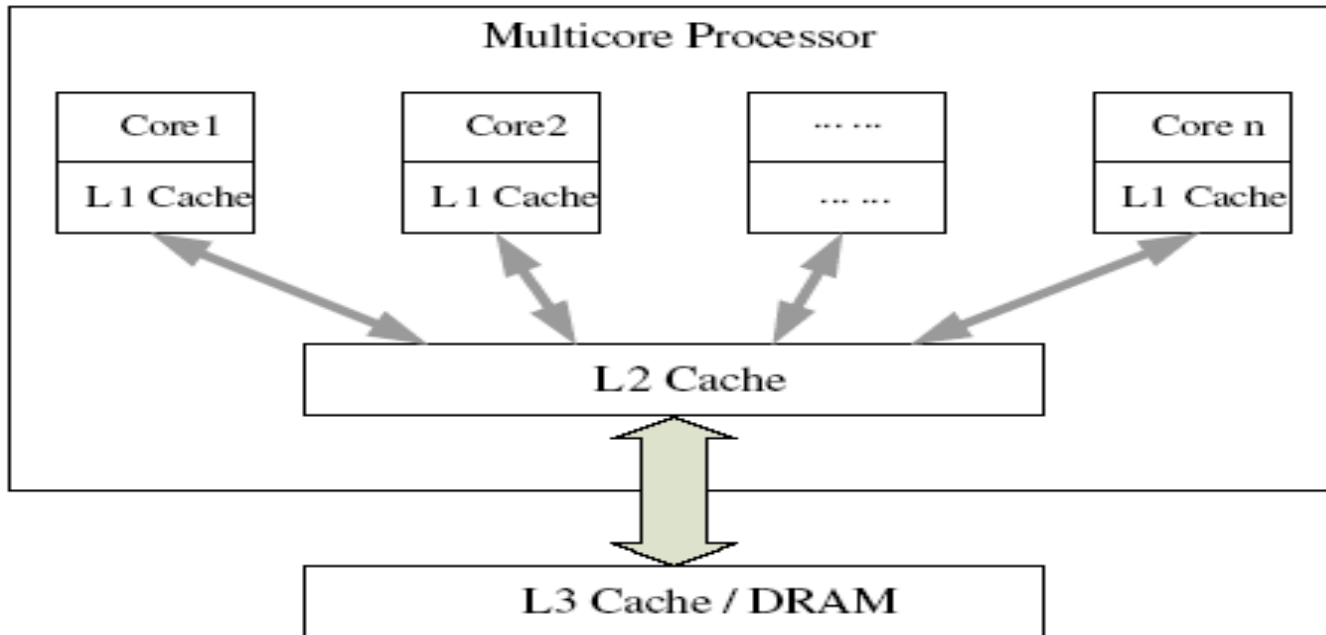- △ more than 10 years
- ⊗ obsolete before plateau

Multicore CPUs and Multithreading Technologies

- Improvement of processor and network technologies over 30 years
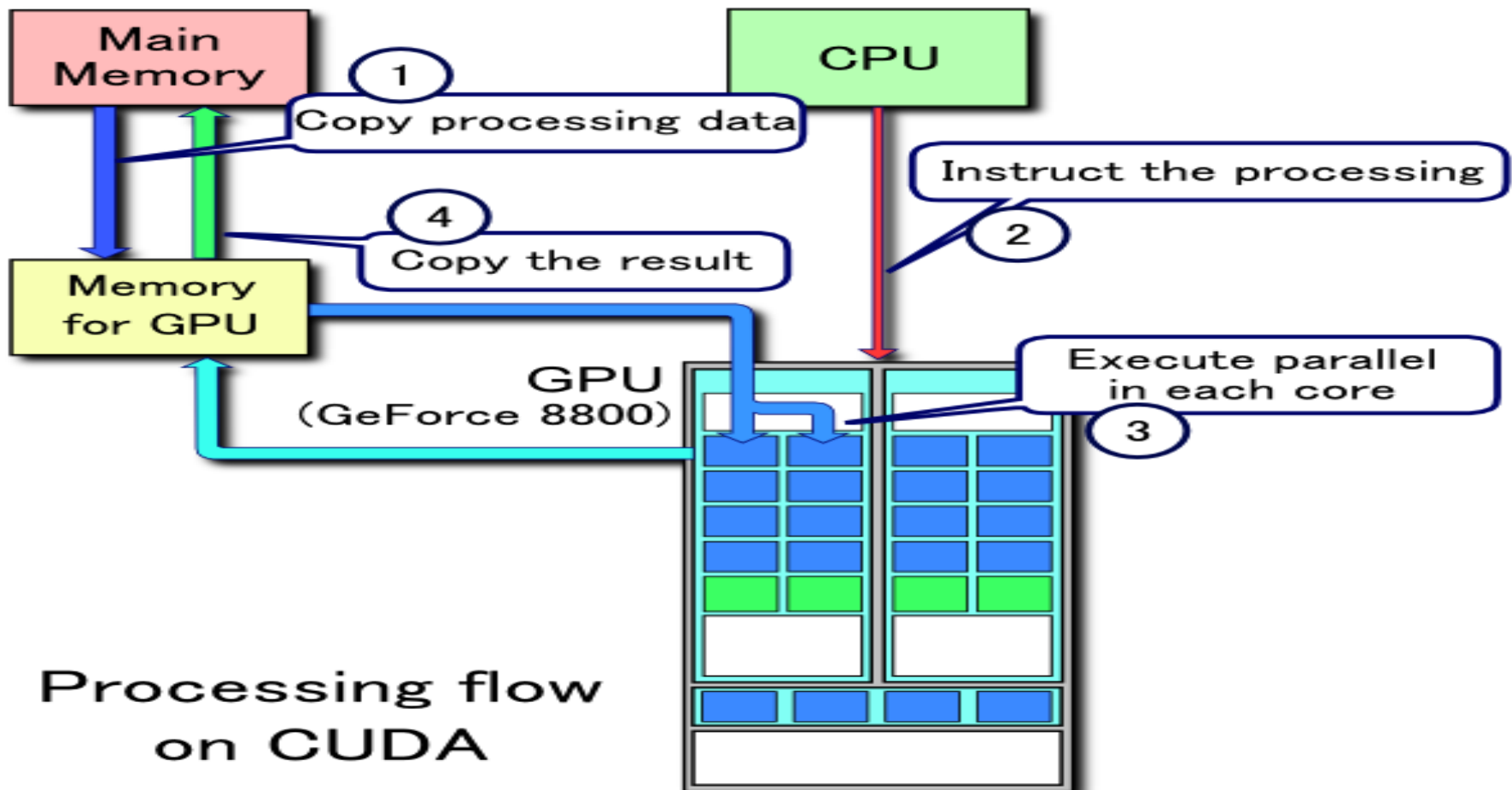
Multicore and Many-Core Architectures

- The schematic of a modern multicore CPU chip using a hierarchy of caches, where the L1 cache is private to each core. The L2 cache is also on chip and shared by all cores. The L3 cache or DRAM is off the multicore chip.
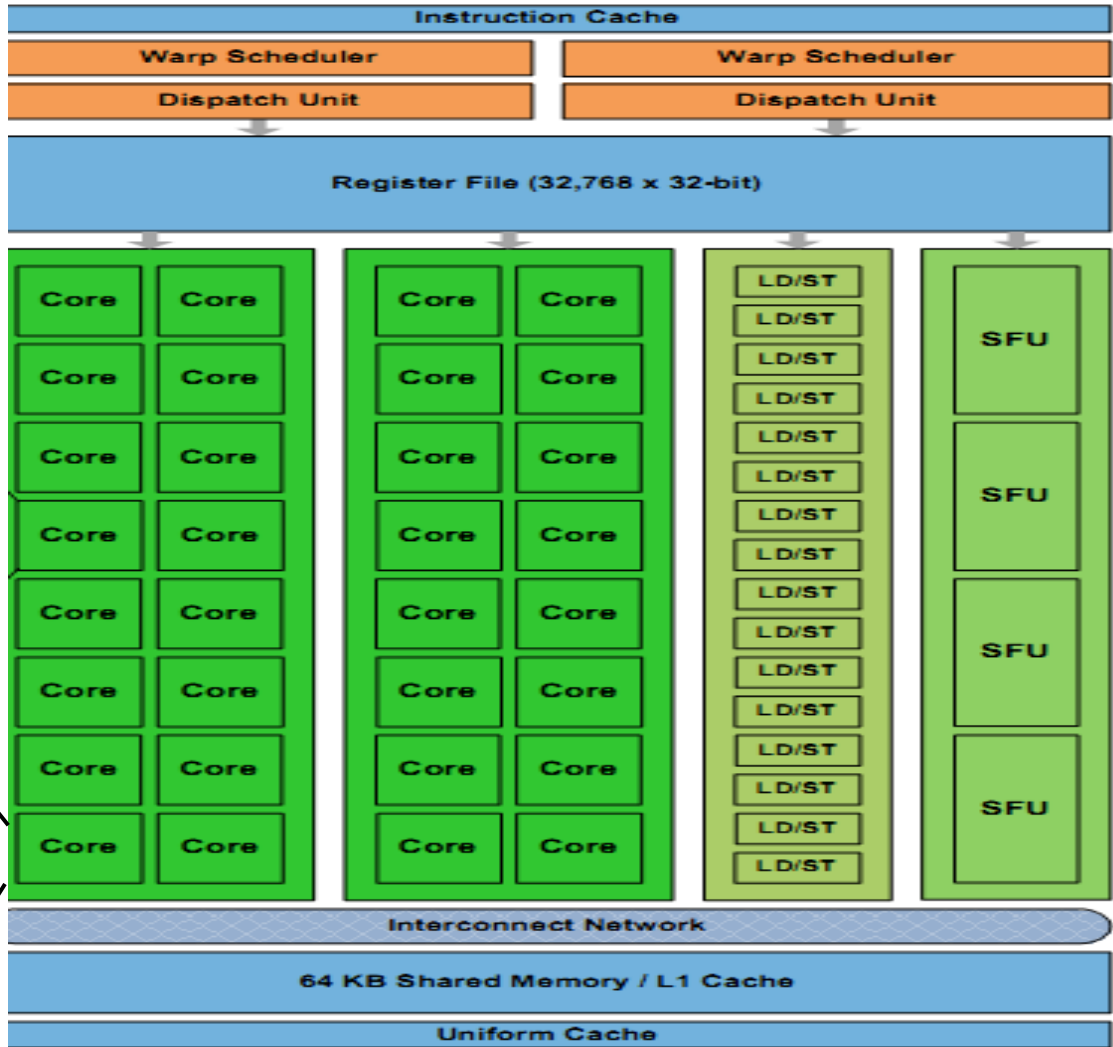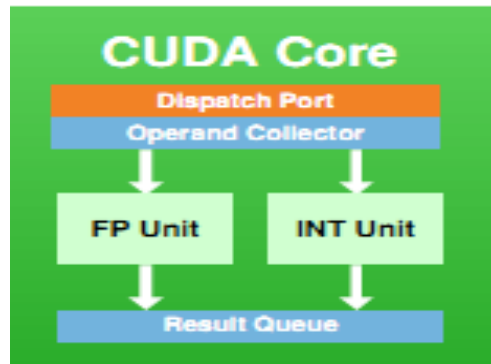
Logistics

- TBA

GPU Computing To Exascale and Beyond

GPU Computing To Exascale and Beyond

- Nvidia Fermi GPU built with 16 streaming multiprocessors (SMs) of 32 Cuda cores each

CUDA Core

| Dispatch Port |
| Operand Collector |

FP Unit | INT Unit

Result Queue

Instruction Cache

| Warp Scheduler | Warp Scheduler |
| Dispatch Unit | Dispatch Unit |

Register File (32,768 x 32-bit)

Core Core Core Core LD/ST SFU

Core Core Core Core LD/ST

Core Core Core Core LD/ST SFU

Core Core Core Core LD/ST

Core Core Core Core LD/ST SFU

Core Core Core Core LD/ST

Core Core Core Core LD/ST SFU

Core Core Core Core LD/ST

Interconnect Network

64 KB Shared Memory / L1 Cache

**Uniform Cache**

GPU Computing To Exascale and Beyond

- Future exascale systems (10^{18} flops)
  - Will likely have hybrid architectures, where GPUs will be used in combination with CPUs

- Challenges of Exascale Computing
  - energy and power
  - memory and storage
  - concurrency and locality
  - system resiliency

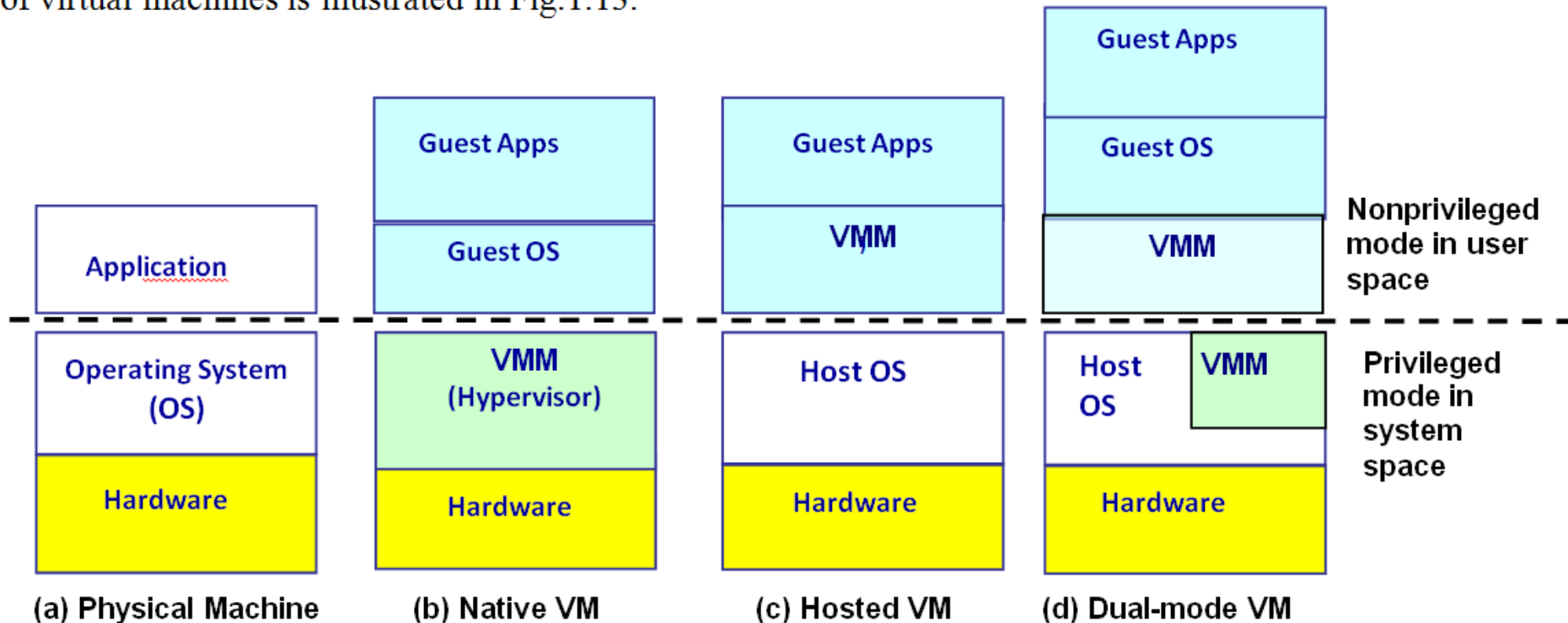# GPU Computing To Exascale and Beyond

- Power efficiency of GPUs
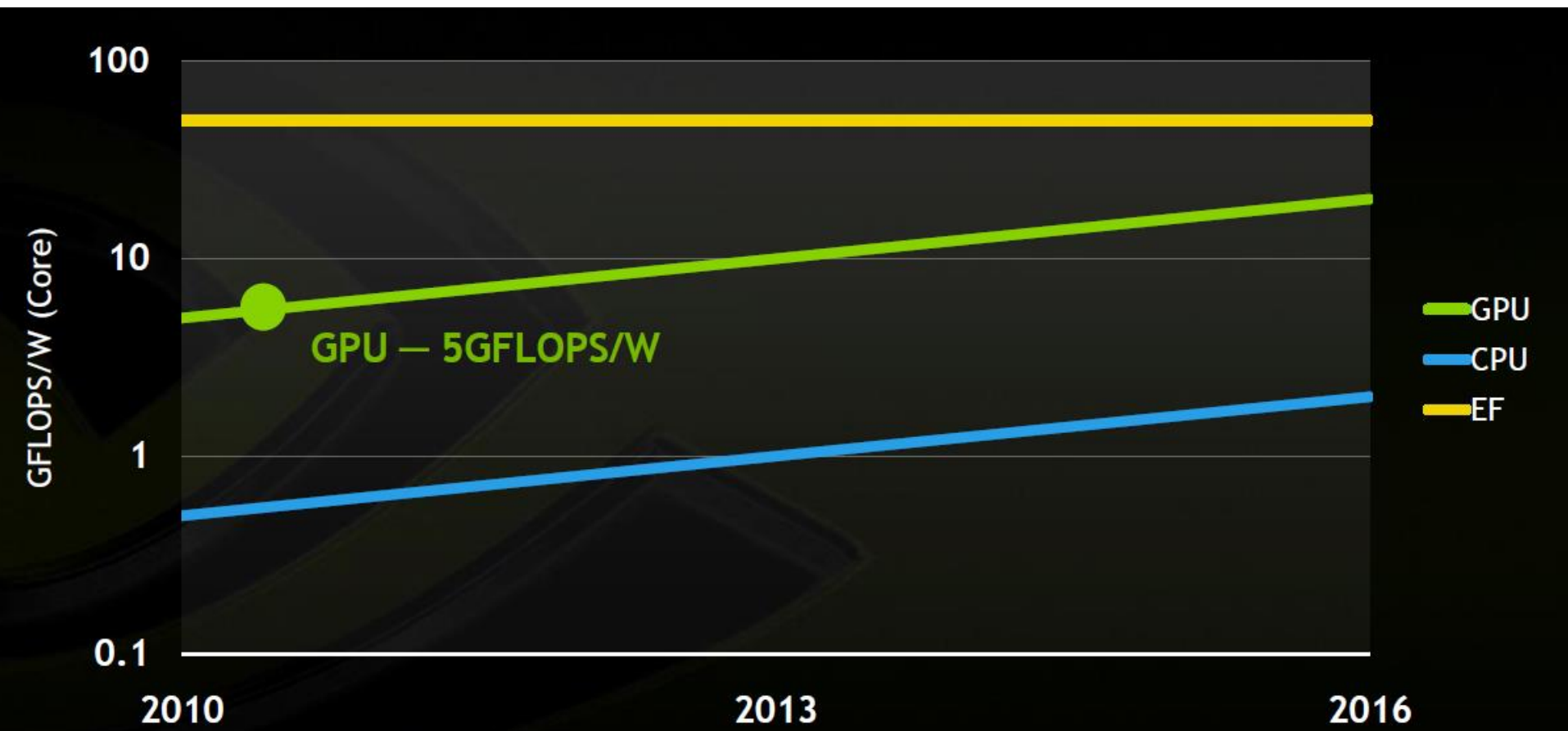
Memory, Storage, and System-Area Networking
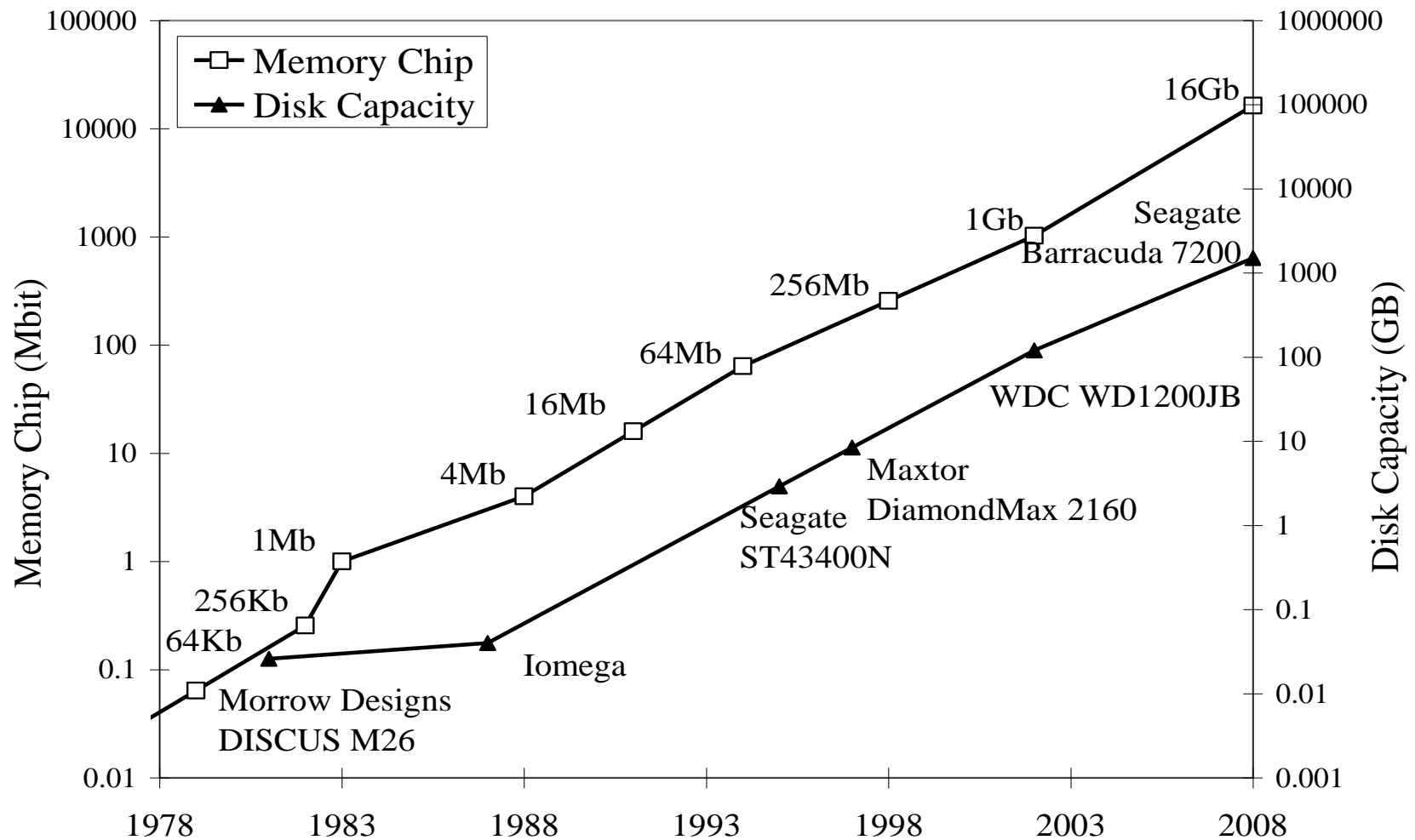
- Improvement of memory and disk technologies over 30 years

Virtual Machines and Virtualization Middleware

ol virtual machines is illustrated in Fig.1.13.

Application				

(a) Physical Machine
- Application
- Operating System (OS)
- Hardware

(b) Native VM
- Guest Apps
- Guest OS
- VMM (Hypervisor)
- Hardware

(c) Hosted VM
- Guest Apps
- VMM
- Host OS
- Hardware

(d) Dual-mode VM
- Guest Apps
- Guest OS
- VMM
- Host OS | VMM
- Hardware

Nonprivileged mode in user space

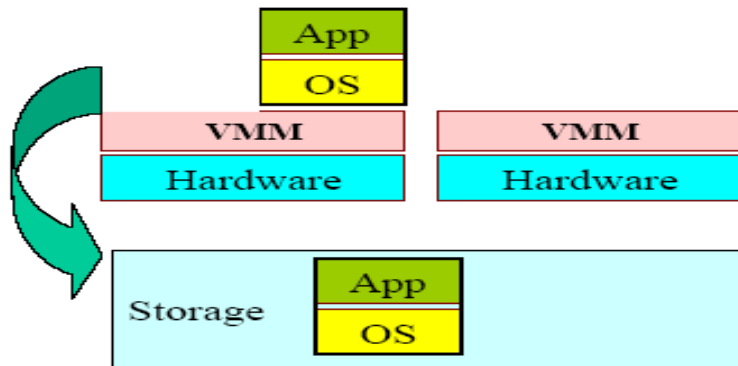**Privileged mode in system space**

- Three virtual machine (VM) architectures in Parts (b-d), compared with the traditional physical machine shown in Part (a)
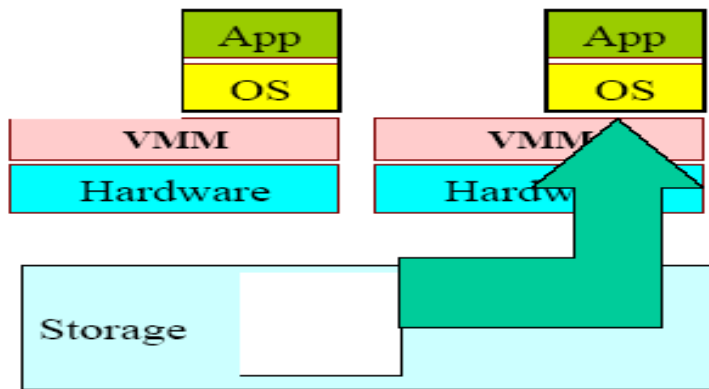
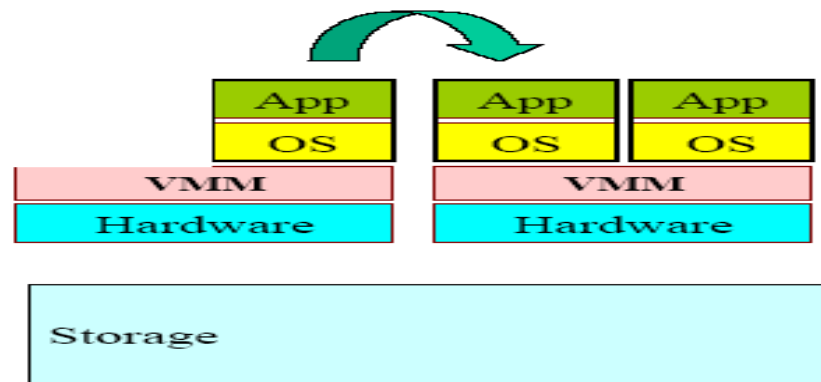# Virtual Machines and Virtualization Middleware
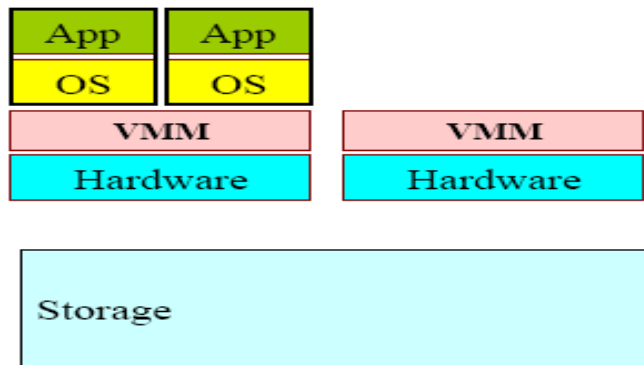
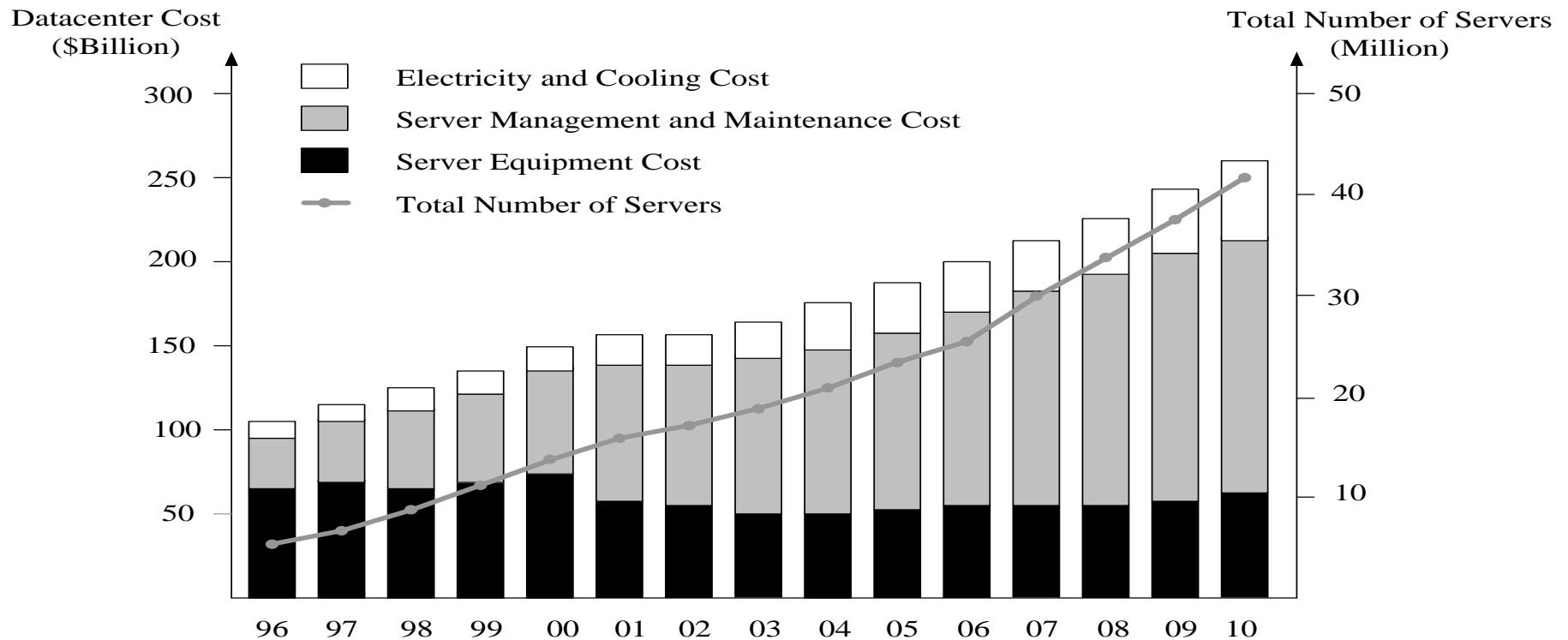
(a) Multiplexing

(b) Suspension (Storage)

(c) Provision (Resume)

(d) Life migration

- Virtual machine multiplexing, suspension, provision, and migration in a distributed computing environment

Datacenter Virtualization for Cloud Computing

- Growth and cost breakdown of datacenters over the years

System Models for Distributed and Cloud Computing

- *Clusters*
- *P2P networks*
- *Computing grids*
- *Internet clouds*

Classification of Parallel and Distributed Computing Systems

Functionality, Applications	Computer Clusters [10, 28, 35]	Peer-to-Peer Networks [42]	Data/Computational Grids [6, 44]	Cloud Platforms [1, 9, 11, 13, 30]
Architecture, Network Connectivity and Size	Network of compute nodes interconnected by SAN, LAN, or WAN, hierarchically	Flexible network of client machines logically connected by an overlay network	Heterogeneous clusters interconnected by high-speed network links over selected resource sites.	Virtualized cluster of servers over datacenters via service-level agreement
Control and Resources Management	Homogeneous nodes with distributed control, running Unix or Linux	Autonomous client nodes, free in and out, with self-organization	Centralized control, server oriented with authenticated security	Dynamic resource provisioning of servers, storage, and networks
Applications and network-centric services	High-performance computing, search engines, and web services, etc.	Most appealing to business file sharing, content delivery, and social networking	Distributed super-computing, global problem solving, and datacenter services	Upgraded web search, utility computing, and outsourced computing services
Representative Operational Systems	Google search engine, SunBlade, IBM Road Runner, Cray XT4, etc.	Gnutella, eMule, BitTorrent, Napster, KaZaA, Skype, JXTA	TeraGrid, GriPhyN, UK EGEE, D-Grid, ChinaGrid, etc.	Google App Engine, IBM Bluecloud, AWS, and Microsoft Azure

Clusters of Cooperative Computers Critical Design Issues

Features	Functional Characterization	Feasible Implementations
Availability Support	Hardware and software support for sustained high availability in cluster	Failover, failback, checkpointing, roll back recovery, non-stop OS, etc
Hardware Fault-Tolerance	Automated failure management to eliminate all single points of failure	Component redundancy, hot swapping, RAID, and multiple power supplies, etc.
Single-System Image (SSI)	Achieving SSI at functional level with hardware and software support, middleware, or OS extensions.	Hardware mechanisms or middleware support to achieve distributed shared memory (DSM) at coherent cache level.
Efficient Communications	To reduce message-passing system overhead and hide latencies	Fast message passing , active messages, enhanced MPI library, etc.
Cluster-wide Job Management	Use a global job management system with better scheduling and monitory	Apply single-job management systems such as LSF, Codine, etc
Dynamic Load Balancing	Balance the workload of all processing nodes along with failure recovery	Workload monitory, process migration, job replication and gang scheduling, etc.
Scalability and Programmability	Adding more servers to a cluster or adding more clusters to a Grid as the workload or data set increases	Use scalable interconnect, performance monitory, distributed execution environment, and better software tools

Peer-to-Peer Network Families

- **Major Categories of Peer-to-Peer Network Families**

System Features	Distributed File Sharing	Collaborative Platform	Distributed P2P Computing	Peer-to-Peer Platform
Attractive Applications	Content distribution of MP3 music, video, open software, etc.	Instant Messaging, Collaborative design and gaming	Scientific exploration and social networking	Open networks for public resources
Operational Problems	Loose security and serious on-line copyright violations	Lack of trust, disturbed by spam, privacy, and peer collusions	Security holes, selfish partners, and peer collusion	Lack of standards or protection protocols
Example Systems	Gnutella, Napster, eMule, BitTorrent, Aimster, KaZaA, etc.	ICQ, AIM, Groove, Magi, Multiplayer Games, Skype, etc.	SETI@home, Geonome@ home, etc.	JXTA, .NET, FightingAid@ home, etc.

Cloud Computing over The Internet Definition

- *A large-scale distributed computing paradigm that is driven by economics of scale, in which a pool of abstracted virtualized, dynamically-scalable, managed computing power, storage, platforms, and services are delivered on demand to external customers over the Internet. -- Ian Foster, 2008*

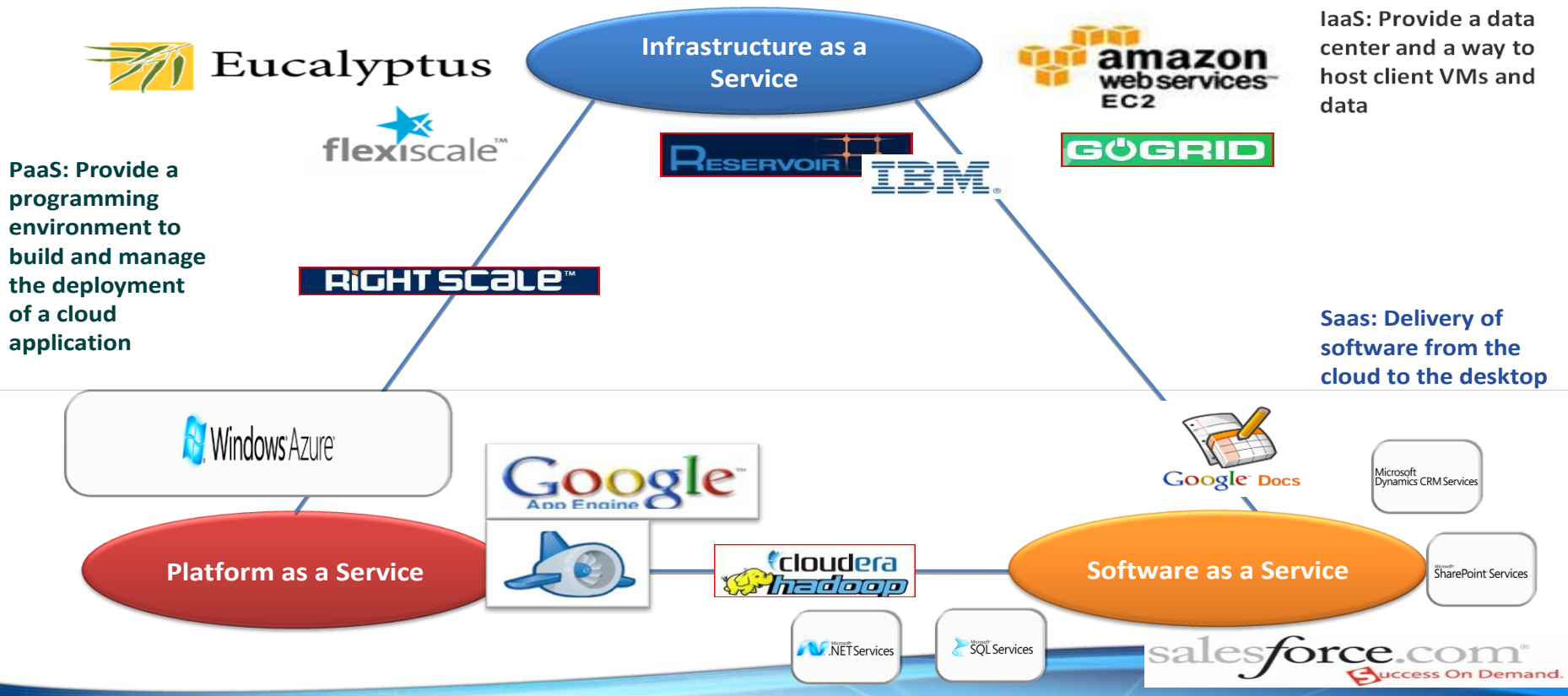# Cloud Computing over The Internet Common Characteristics

1. Cloud platform offers a scalable computing paradigm built around the datacenters.

2. Cloud resources are dynamically provisioned by datacenters upon user demand.

3. Cloud system provides compute, storage and flexible platforms for upgraded web services.

4. Cloud computing relies heavily on the virtualization of all sorts of resources.

5. Cloud computing defines a new paradigm for collective computing, data consumption and delivery of information services over the Internet.

6. Clouds stress the cost of ownership reduction in mega datacenters.

Cloud Computing over The Internet Infrastructure

Cloud Computing over The Internet Cloud Landscape

IaaS: Provide a data center and a way to host client VMs and data

PaaS: Provide a programming environment to build and manage the deployment of a cloud application

Saas: Delivery of software from the cloud to the desktop

Infrastructure as a Service

Platform as a Service

Software as a Service

Cloud Computing over The Internet
Cloud Landscape

- ***Infrastructure as a Service (IaaS):***
  - This model put together infrastructures demanded by users, namely servers, storage, networks, and datacenter fabric. The user can deploy and run on multiple VMs running guest OSes on specific applications. The user does not manage or control the underlying cloud infrastructure, but can specify when to request and release the needed resources.

- ***Platform as a Service (PaaS):***
  - This model provides the user to deploy user-built applications onto a virtualized cloud platform. PaaS include middleware, database, development tools, and some runtime supports like Web 2.0 and Java, etc. The platform include both hardware and software integrated with specific programming interfaces. The provider supplies the API and software tools (e.g., Java, python, Web2.0, .Net). The user is freed from managing the cloud infrastructure.

- ***Software as a Service (SaaS):***
  - This refers to browser-initiated application software over thousands of paid cloud customers. The SaaS model applies to business processes, industry applications, CRM (*consumer relationship management)*, ERP (*enterprise resources planning*), HR (*human resources*) and collaborative applications. On the customer side, there is no upfront investment in servers or software licensing. On the provider side, costs are rather low, compared with conventional hosting of user applications.

Parallel and Distributed Programming Models

- Parallel and Distributed Programming Models and Toolsets

Model	Objectives and Web Link	Attractive Features Implemented
MPI	Message-Passing Interface is a library of subprograms that can be called from C or Fortran to write parallel programs running on distributed computer systems [3, 28, 42]	Specify synchronous or asynchronous point-to-point and collective communication commands and I/O operations in user programs for message-passing execution
MapReduce	A web programming model for scalable data processing on large cluster over large datasets, or in web search operations [17]	A *Map* function generates a set of intermediate key/value pairs. A *Reduce* function to merge all intermediate values with the same key
Hadoop	A software library to write and run large user applications on vas datasets in business applications. http://hadoop.apache.org/core/	Hadoop is scalable, economical, efficient and reliable in providing users with easy access of commercial clusters

System Scalability

- ### *Size Scalability:*
  - This refers to achieve higher performance or performing more functionality by increasing the *machine size*. The word "size" refers to adding the number of processors;   more cache, memory, storage or I/O channels. The most obvious way to simple counting the number of processors installed. Not all parallel computer or distributed architectures are equally size-scalable. For example, IBM S2 was scaled up to 512 processors in 1997. But in 2008, the IBM BlueGene/L system scaled up to 65,000 processors.

- ### *Software Scalability:*
  - This refers to upgrades in OS or compilers, adding mathematical and engineering libraries, porting new application software, and install more user-friendly programming environment. Some software upgrade may not work with large system configurations. Testing and fine-tuning of new software on larger system is a non-trivial job.

- ### *Application scalability:*
  - This refers to the match of *problem size* scalability with the *machine size* scalability. Problem size affects the size of the data set or the workload increase. Instead of increasing machine size, users can enlarge the problem size to enhance the system efficiency or cost-effectiveness.

- ### *Technology Scalability:*
  - This refers to a system that can adapt to changes in building technologies, such as those component and networking technologies discussed in Section 3.1. Scaling a system design with new technology must consider three aspects: *time, space,* and *heterogeneity.* Time refers to generation scalability. Changing to new-generation processors, one must consider the impact to motherboard, power supply, packaging and cooling, etc. Based on the past experience, most system upgrade their commodity processors every 3 to 5 years. Space is more related to packaging and energy concerns. Technology scalability demands harmony and portability among suppliers.

Amdahl's Law

- **Speedup = $S = T / [\,\alpha T + (1-\alpha)T / n\,] = 1 / [\,\alpha + (1-\alpha)/n\,]$**

- The maximum speedup of n is achieved, only if the sequential bottleneck α is reduced to zero or the code is fully parallelizable with α = 0. As the cluster becomes sufficiently large, i.e. n → ∞, S approaches 1/ α , an upper bound on the speedup S.

- Amdahl's law teaches us that we should make the sequential bottleneck as small as possible.

- Increasing the cluster size alone may not give a good speedup in this case.

Amdahl's Law

- $E = S / n = 1 / [\, \alpha\, n + 1 - \alpha\,]$
- Fixed workload size poses efficiency problems
- **Gustafson's Law**
  - Allows variable size workloads to increase efficiency
  - For fixed workload, apply Amdahl's law
  - For scaled problems, apply Gustafson's Law

Questions

?