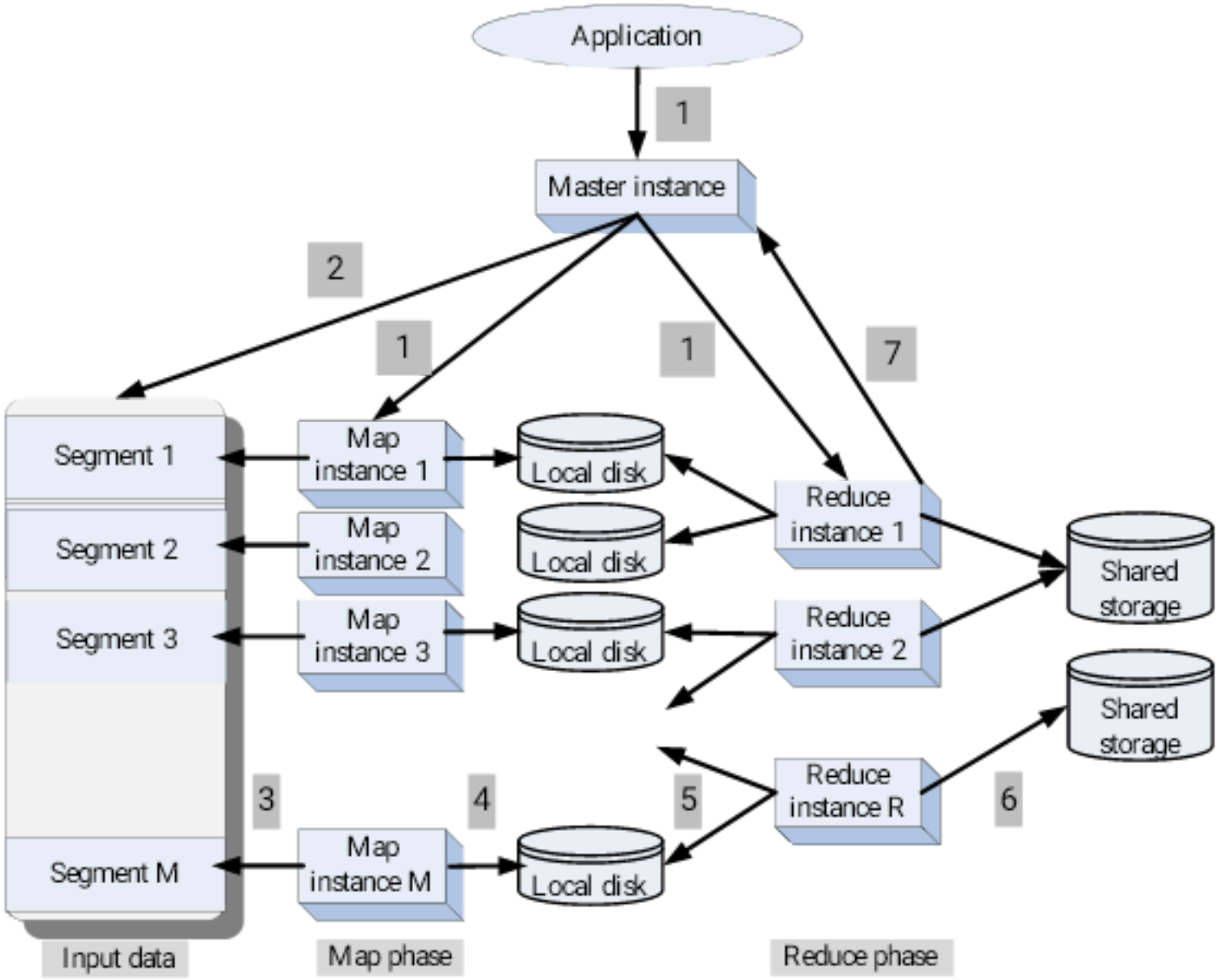
MapReduce philosophy

1. An application starts a master instance, M worker instances for the *Map phase* and later R worker instances for the *Reduce phase*.
2. The master instance partitions the input data in M *segments*.
3. Each *map instance* reads its input data segment and processes the data.
4. The results of the processing are stored on the local disks of the servers where the map instances run.
5. When all map instances have finished processing their data, the R reduce instances read the results of the first phase and merge the partial results.
6. The final results are written by the reduce instances to a shared storage server.
7. The master instance monitors the reduce instances and when all of them report task completion the application is terminated.

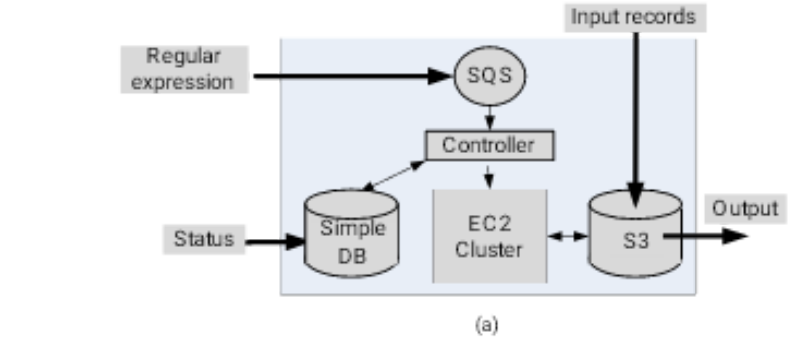

Application
1
Master instance
2
1
1
7
Segment 1
Segment 2
Segment 3
Segment M
Map instance 1
Map instance 2
Map instance 3
Map instance M
Local disk
Local disk
Local disk
Local disk
Reduce instance 1
Reduce instance 2
Reduce instance R
Shared storage
Shared storage
3
4
5
6
Input data
Map phase
Reduce phase

Case study: GrepTheWeb

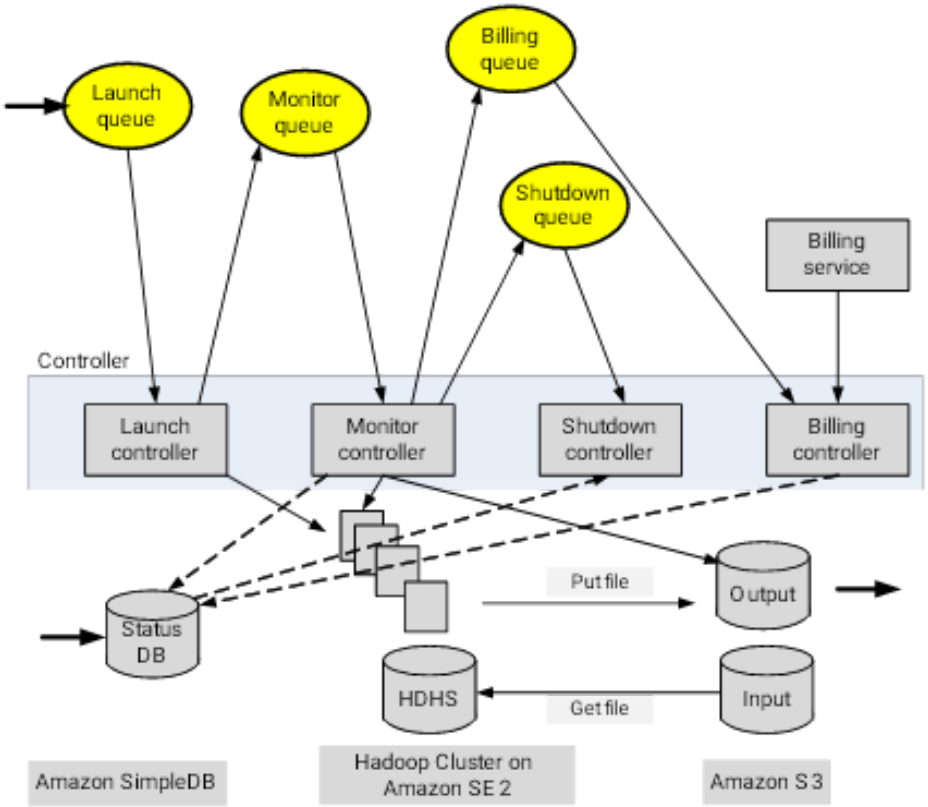
- The application illustrates the means to
  - create an on-demand infrastructure.
  - run it on a massively distributed system in a manner that allows it to run in parallel and scale up and down, based on the number of users and the problem size.
- GrepTheWeb
  - Performs a search of a very large set of records to identify records that satisfy a regular expression.
  - It is analogous to the Unix *grep* command.
  - The source is a collection of document URLs produced by the Alexa Web Search, a software system that crawls the web every night.
  - Uses message passing to trigger the activities of multiple controller threads which launch the application, initiate processing, shutdown the system, and create billing records.

(a) The simplified workflow showing the inputs:
- the regular expression.
- the input records generated by the web crawler.
- the user commands to report the current status and to terminate the processing.

(b) The detailed workflow. The system is based on message passing between several queues; four controller threads periodically poll their associated input queues, retrieve messages, and carry out the required actions



(a)



(b)

Chapter 5 – Cloud Resource Virtualization

Contents

- Virtualization.
- Layering and virtualization.
- Virtual machine monitor.
- Virtual machine.
- Performance and security isolation.
- Architectural support for virtualization.
- x86 support for virtualization.
- Full and paravirtualization.

Motivation

- There are many physical realizations of the fundamental abstractions necessary to describe the operation of a computing systems.
  - Interpreters.
  - Memory.
  - Communications links.
- Virtualization is a basic tenet of cloud computing, it simplifies the management of physical resources for the three abstractions.
- The state of a virtual machine (VM) running under a virtual machine monitor (VMM) can de saved and migrated to another server to balance the load.
- Virtualization allows users to operate in environments they are familiar with, rather than forcing them to idiosyncratic ones.

Motivation (cont'd)

- Cloud resource virtualization is important for:
  - System security, as it allows isolation of services running on the same hardware.
  - Performance and reliability, as it allows applications to migrate from one platform to another.
  - The development and management of services offered by a provider.
  - Performance isolation.

Virtualization

- Simulates the interface to a physical object by:
  - Multiplexing: creates multiple virtual objects from one instance of a physical object. Example - a processor is multiplexed among a number of processes or threads.
  - Aggregation: creates one virtual object from multiple physical objects. Example - a number of physical disks are aggregated into a RAID disk.
  - Emulation: constructs a virtual object from a different type of a physical object. Example - a physical disk emulates a Random Access Memory (RAM).
  - Multiplexing and emulation. Examples - virtual memory with paging multiplexes real memory and disk; a virtual address emulates a real address.

Layering

- Layering – a common approach to manage system complexity.
  - Minimizes the interactions among the subsystems of a complex system.
  - Simplifies the description of the subsystems; each subsystem is abstracted through its interfaces with the other subsystems.
  - We are able to design, implement, and modify the individual subsystems independently.
- Layering in a computer system.
  - Hardware.
  - Software.
    - Operating system.
    - Libraries.
    - Applications.

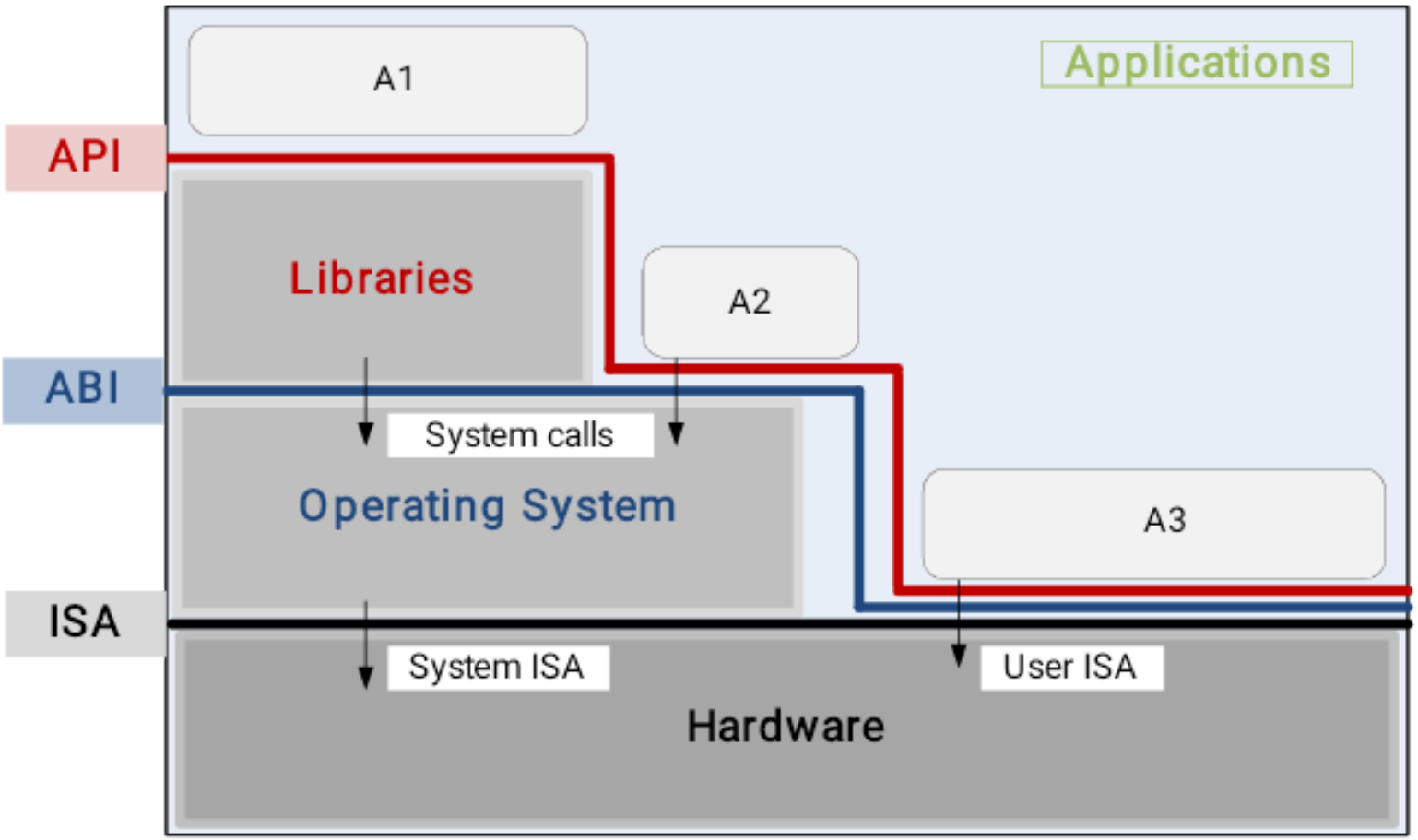
Interfaces

- Instruction Set Architecture (ISA) – at the boundary between hardware and software.

- Application Binary Interface (ABI) – allows the ensemble consisting of the application and the library modules to access the hardware; the ABI does not include privileged system instructions, instead it invokes system calls.

- Application Program Interface (API) - defines the set of instructions the hardware was designed to execute and gives the application access to the ISA; it includes HLL library calls which often invoke system calls.

gives applications the access to Hll which in turn can invoke system calls to the OS or can give access to ISA



Application Programming Interface, Application Binary Interface, and Instruction Set Architecture . An application uses library functions (A1), makes system calls (A2), and executes machine instructions (A3).

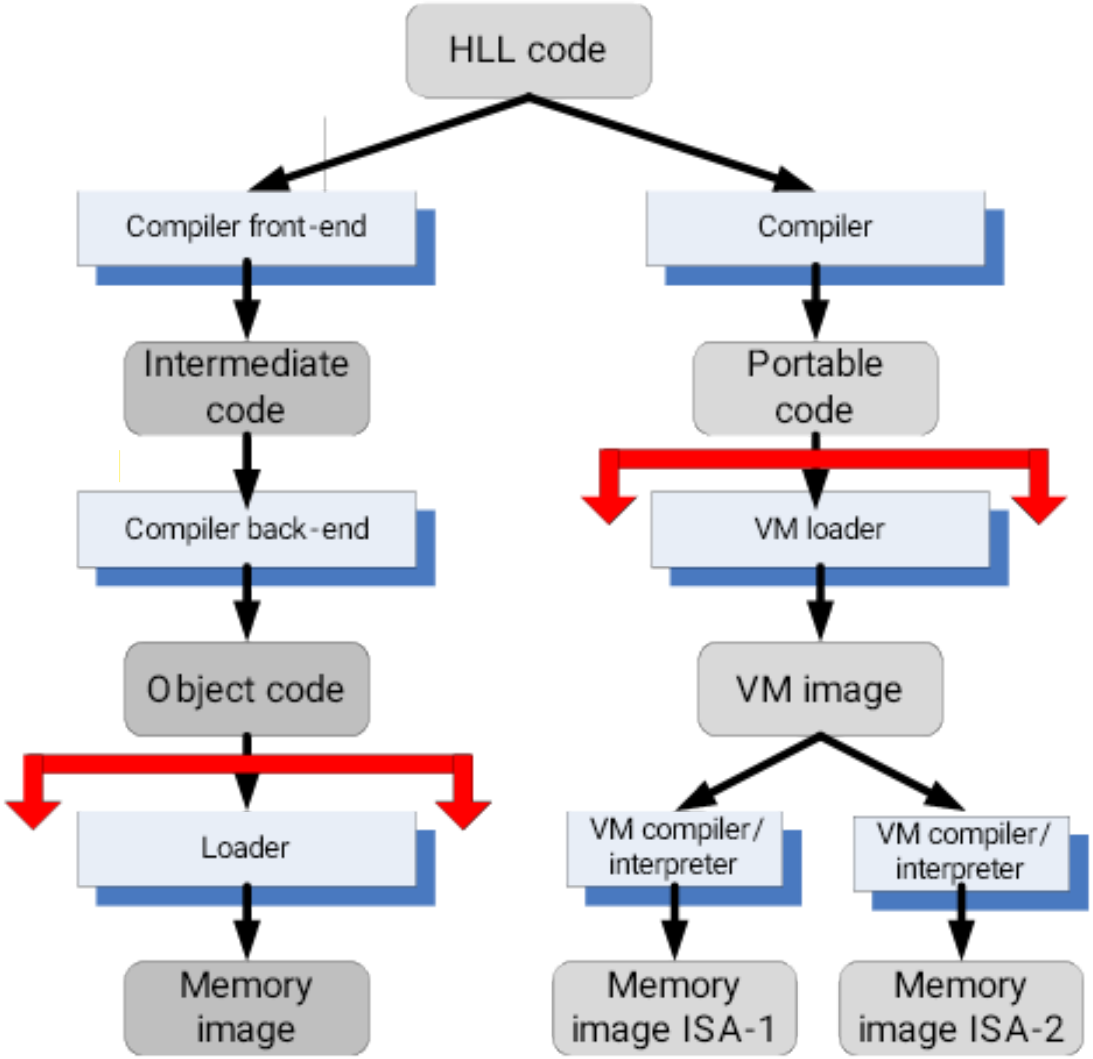
Code portability

- Binaries created by a compiler for a specific ISA and a specific operating systems are not portable.

- It is possible, though, to compile a HLL program for a virtual machine (VM) environment where portable code is produced and distributed and then converted by binary translators to the ISA of the host system.

A dynamic binary translation converts blocks of guest instructions from the portable code to the host instruction and leads to a significant performance improvement, as such blocks are cached and reused

The HLL code is processed by a compiler or interpreter that is specifically designed for the target VM environment. Instead of translating the code into machine code for a specific ISA/OS, the compiler generates bytecode or intermediate code that is understood by the VM.


HLL code
Compiler front-end
Compiler
Intermediate code
Portable code
Compiler back-end
VM loader
Object code
VM image
Loader
VM compiler/ interpreter
VM compiler/ interpreter
Memory image
Memory image ISA-1
Memory image ISA-2

Virtual machine monitor (VMM / hypervisor)

- Partitions the resources of computer system into one or more virtual machines (VMs). Allows several operating systems to run concurrently on a single hardware platform.

- A VMM allows
  - Multiple services to share the same platform.
  - Live migration - the movement of a server from one platform to another.
  - System modification while maintaining backward compatibility with the original system.
  - Enforces isolation among the systems, thus security.

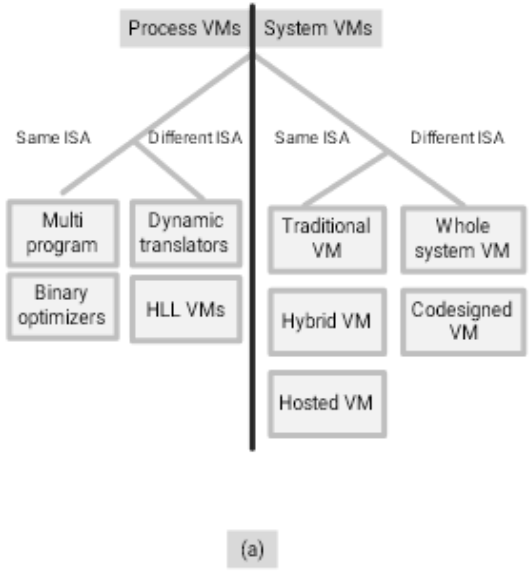
VMM virtualizes the CPU and the memory

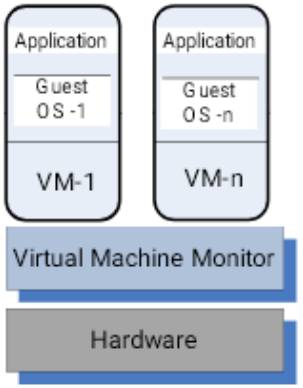
- A VMM
  - Traps the privileged instructions executed by a guest OS and enforces the correctness and safety of the operation.
  - Traps interrupts and dispatches them to the individual guest operating systems.
  - Controls the virtual memory management.
  - Maintains a shadow page table for each guest OS and replicates any modification made by the guest OS in its own shadow page table. This shadow page table points to the actual page frame and it is used by the Memory Management Unit (MMU) for dynamic address translation.
  - Monitors the system performance and takes corrective actions to avoid performance degradation. For example, the VMM may swap out a Virtual Machine to avoid thrashing.

Virtual machines (VMs)

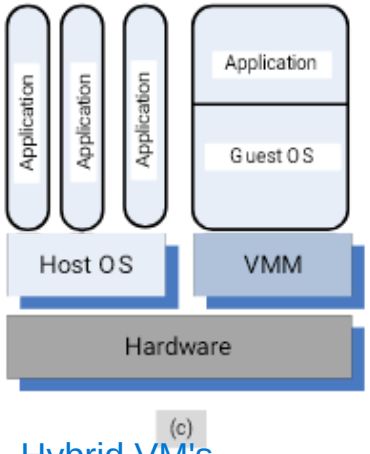
- VM - isolated environment that appears to be a whole computer, but actually only has access to a portion of the computer resources.
- Process VM - a virtual platform created for an individual process and destroyed once the process terminates.
- System VM - supports an operating system together with many user processes.
- Traditional VM - supports multiple virtual machines and runs directly on the hardware.
- Hybrid VM - shares the hardware with a host operating system and supports multiple virtual machines.
- Hosted VM - runs under a host operating system.

Traditional, hybrid, and hosted VMs

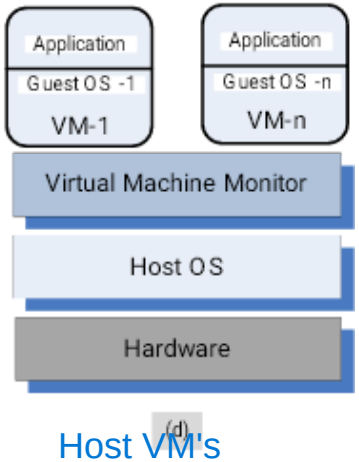




Traditional VM's



Hybrid VM's



Host VM's

Name	Host ISA	Guest ISA	Host OS	guest OS	Company
Integrity VM	*x86*-64	*x86*-64	HP-Unix	Linux,Windows HP Unix	HP
Power VM	Power	Power	No host OS	Linux, AIX	IBM
z/VM	z-ISA	z-ISA	No host OS	Linux on z-ISA	IBM
Lynx Secure	*x86*	*x86*	No host OS	Linux, Windows	LinuxWorks
Hyper-V Server	*x86*-64	*x86*-64	Windows	Windows	Microsoft
Oracle VM	*x86*, *x86*-64	*x86*, *x86*-64	No host OS	Linux, Windows	Oracle
RTS Hypervisor	*x86*	*x86*	No host OS	Linux, Windows	Real Time Systems
SUN xVM	*x86*, SPARC	same as host	No host OS	Linux, Windows	SUN
VMware EX Server	*x86*, *x86*-64	*x86*, *x86*-64	No host OS	Linux, Windows Solaris, FreeBSD	VMware
VMware Fusion	*x86*, *x86*-64	*x86*, *x86*-64	MAC OS *x86*	Linux, Windows Solaris, FreeBSD	VMware
VMware Server	*x86*, *x86*-64	*x86*, *x86*-64	Linux, Windows	Linux, Windows Solaris, FreeBSD	VMware
VMware Workstation	*x86*, *x86*-64	*x86*, *x86*-64	Linux, Windows	Linux, Windows Solaris, FreeBSD	VMware
VMware Player	*x86*, *x86*-64	*x86*, *x86*-64	Linux Windows	Linux, Windows Solaris, FreeBSD	VMware
Denali	*x86*	*x86*	Denali	ILVACO, NetBSD	University of Washington
Xen	*x86*, *x86*-64	*x86*, *x86*-64	Linux Solaris	Linux, Solaris NetBSD	University of Cambridge

Performance and security isolation

- The run-time behavior of an application is affected by other applications running concurrently on the same platform and competing for CPU cycles, cache, main memory, disk and network access. Thus, it is difficult to predict the completion time!

- Performance isolation - a critical condition for QoS guarantees in shared computing environments.

- A VMM is a much simpler and better specified system than a traditional operating system. Example - Xen has approximately 60,000 lines of code; Denali has only about half, 30,000.

- The security vulnerability of VMMs is considerably reduced as the systems expose a much smaller number of privileged functions.

Computer architecture and virtualization

- Conditions for efficient virtualization:
  - A program running under the VMM should exhibit a behavior essentially identical to that demonstrated when running on an equivalent machine directly.
  - The VMM should be in complete control of the virtualized resources.
  - A statistically significant fraction of machine instructions must be executed without the intervention of the VMM.
- Two classes of machine instructions:
  - Sensitive - require special precautions at execution time:
    - Control sensitive - instructions that attempt to change either the memory allocation or the privileged mode.
    - Mode sensitive - instructions whose behavior is different in the privileged mode.
  - Innocuous - not sensitive.

Full virtualization and paravirtualization

- Full virtualization – a guest OS can run unchanged under the VMM as if it was running directly on the hardware platform.
  - Requires a virtualizable architecture.
  - Examples: Vmware.

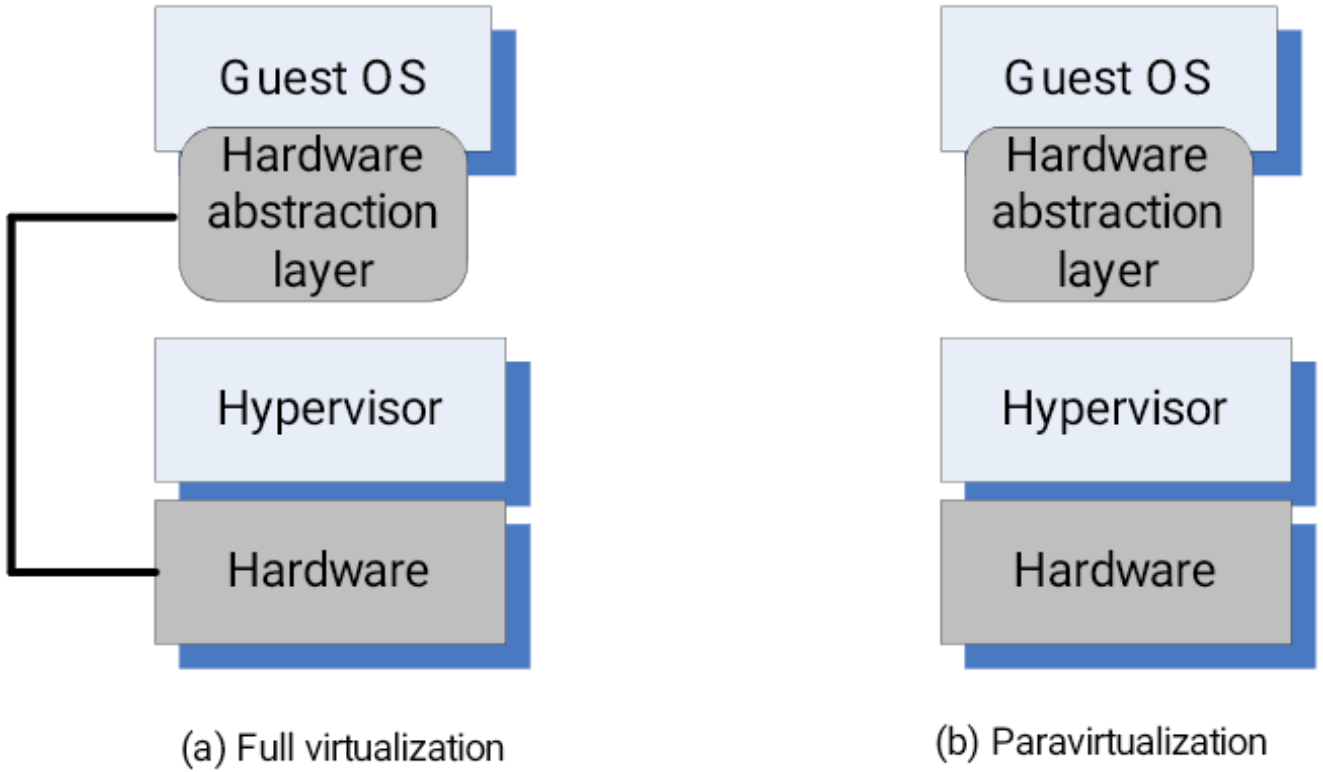
OS is having the illusion of running on its dedicated hardware.

- Paravirtualization - a guest operating system is modified to use only instructions that can be virtualized. Reasons for paravirtualization:
  - Some aspects of the hardware cannot be virtualized.
  - Improved performance.
  - Present a simpler interface.

  Examples: Xen, Denaly

In para-virtualization, the guest operating systems are specifically modified or "enlightened" to be aware that they are running in a virtualized environment.

Full virtualization and paravirtualization



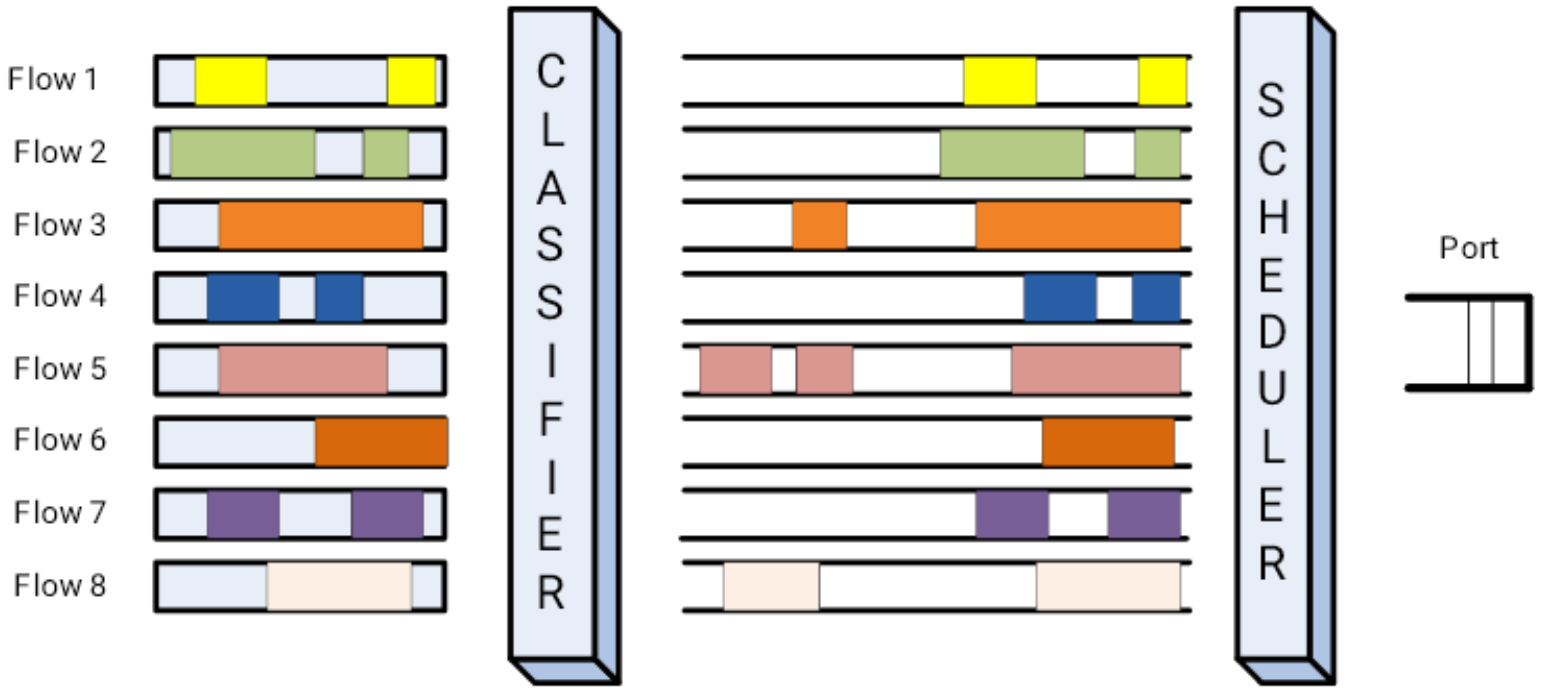
Chapter 7 – Networking Support

Contents

- Network management; class-based queuing.
- Cloud interconnection networks.
- Storage area networks.
- Content delivery networks.
- Scale free Networks
- Epidemic Algorithm

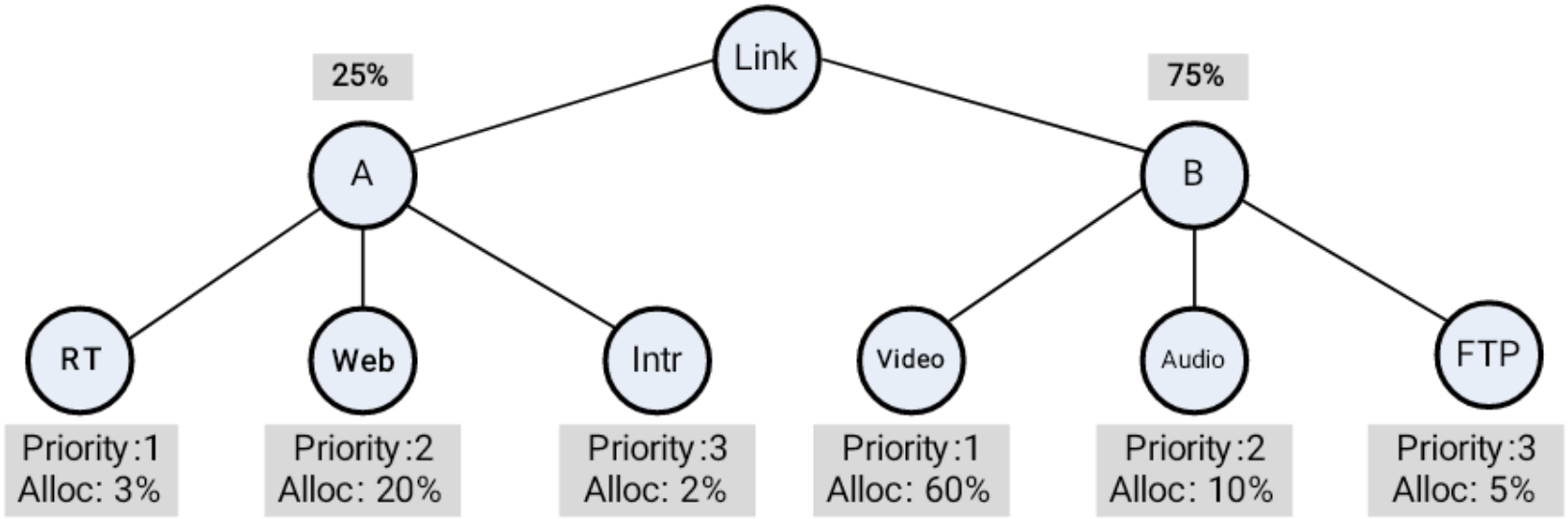
Class-Based Queuing (CBQ)

- The objectives of CBQ are to support:
  - Flexible link sharing for applications which require bandwidth guarantees such as VoIP, video-streaming, and audio-streaming.
  - Some balance between short-lived network flows, such as web searches, and long-lived ones, such as video-streaming or file transfers.
- CBQ aggregates the connections and constructs a tree-like hierarchy of classes with different priorities and throughput allocations. CBQ uses several functional units:
  - a classifier which uses the information in the packet header to assign arriving packets to classes.
  - an estimator of the short-term bandwidth for the class.
  - a selector/scheduler which identifies the highest priority class to send next and, if multiple classes have the same priority, to schedule them on a round-robin base.
  - a delayer to compute the next time when a class that has exceeded its link allocation is allowed to send.



Class-Based Queuing (CBQ) - packets are first classified into flows and then assigned to a queue dedicated to the flow; queues are serviced one packet at a time in round-robin order and empty queues are skipped

Class-Based Queuing (CBQ)



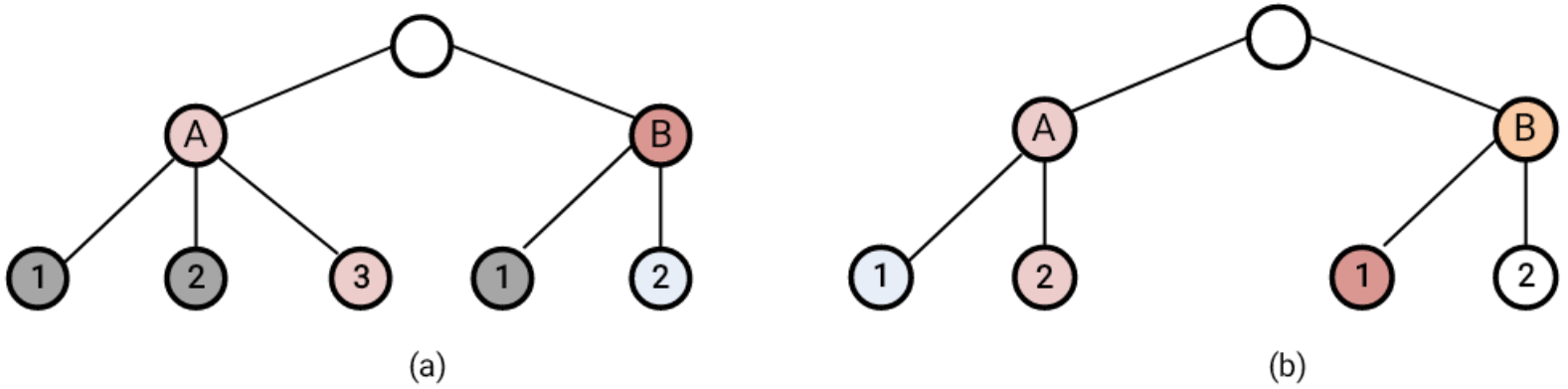
CBQ link sharing for two groups: A□short-lived and B□long-lived traffic, allocated 25% and 75% of the link capacity. There are three classes with priorities 1, 2, and 3: (i) Real-time (RT) and the video streaming have priority 1 and are allocated 3% and 60%, respectively, (ii) Web transactions and audio streaming have priority 2 and are allocated 20% and 10%, respectively; (iii) In interactive (Intr) and file transfer (FTP) applications have priority 3 and are allocated 2% and 5%, respectively.

Class-Based Queuing (CBQ)

- A class is
  - overlimit if over a certain recent period it has used more than its bandwidth allocation (in bytes per second).
  - underlimit if it has used less.
  - atlimit if it has used exactly its allocation.

- A leaf class is
  - satisfied if it is underlimit and has a persistent backlog.
  - unsatisfied otherwise.

- A non-leaf class is unsatisfied if it is underlimit and has some descendent class with a persistent backlog.



There are two groups A and B and three types of traffic, e.g., web, real-time, and interactive, denoted as 1, 2, and 3. (a) Group A and class A.3 traffic are underlimit and unsatisfied; classes A.1, A.2 and B.1 are overlimit, unsatisfied and with persistent backlog and have to be regulated; (b) Group A is underlimit and unsatisfied; Group B is overlimit and needs to be regulated; class A.1 traffic is underlimit; class A.2 is overlimit and with persistent backlog; class B.1 traffic is overlimit and with persistent backlog and needs to be regulated.

Hierarchical Token Buckets (HTB)

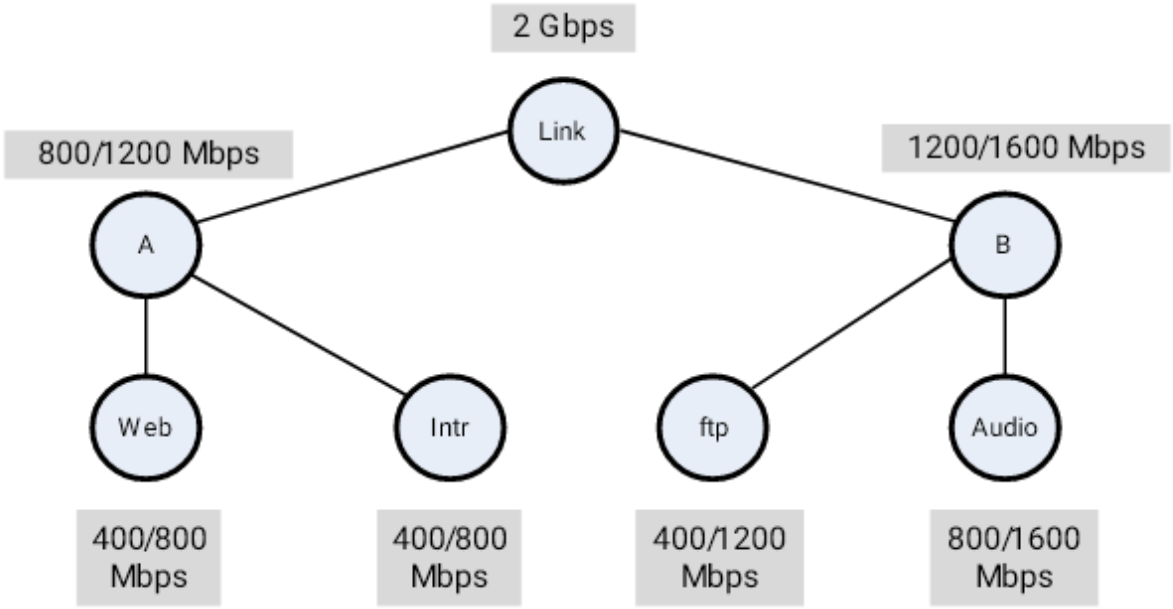
- Hierarchical Token Buckets (HTB) is a link sharing algorithm inspired by CBQ.

- The Linux kernel implements HTB.

- Each class has
  - An assured rate (AR).
  - A ceil rate (CR).

- HTB supports borrowing If a class C needs a rate above its AR it tries to borrow from its parent; then the parent examines its children and, if there are classes running at a rate lower that their AR, the parent can borrow from them and reallocate it to class C.

Hierarchical Token Buckets (HTB)



HTB packet scheduling uses for every node a ceil rate in addition to the assured rate.

Cloud interconnection networks

- While processor and memory technology have followed Moore's law, the interconnection networks have evolved at a slower pace and have become a major factor in determining the overall performance and cost of the system.
- The networking infrastructure is organized hierarchically: servers are packed into racks and interconnected by a top of the rack router; the rack routers are connected to cluster routers which in turn are interconnected by a local communication fabric.
- The networking infrastructure of a cloud must satisfy several requirements:
  - Scalability.
  - Low cost.
  - Low-latency.
  - High bandwidth.
  - Provide location transparent communication between servers.

Location transparent communication

- Every server should be able to communicate with every other server with similar speed and latency.
- Applications need not be location aware.
- It also reduces the complexity of the system management.
- In a hierarchical organization *true location transparency is not feasible* and cost considerations ultimately decide the actual organization and performance of the communication fabric.

Interconnection networks - InfiniBand

- Interconnection network used by supercomputers and computer clouds.
  - Has a switched fabric topology designed to be scalable.
  - Supports several signaling rates.
  - The energy consumption depends on the throughput.
  - Links can be bonded together for additional throughput.
- The data rates.
  - single data rate (SDR) - 2.5 Gbps in each direction per connection.
  - double data rate (DDR) - 5 Gbps.
  - quad data rate (QDR) – 10 Gbps.
  - fourteen data rate (FDR) – 14.0625 Gbps.
  - enhanced data rated (EDR) – 25.78125 Gbps.
- Advantages.
  - high throughput, low latency.
  - supports quality of service guarantees and failover - the capability to switch to a redundant or standby system

Routers and switches

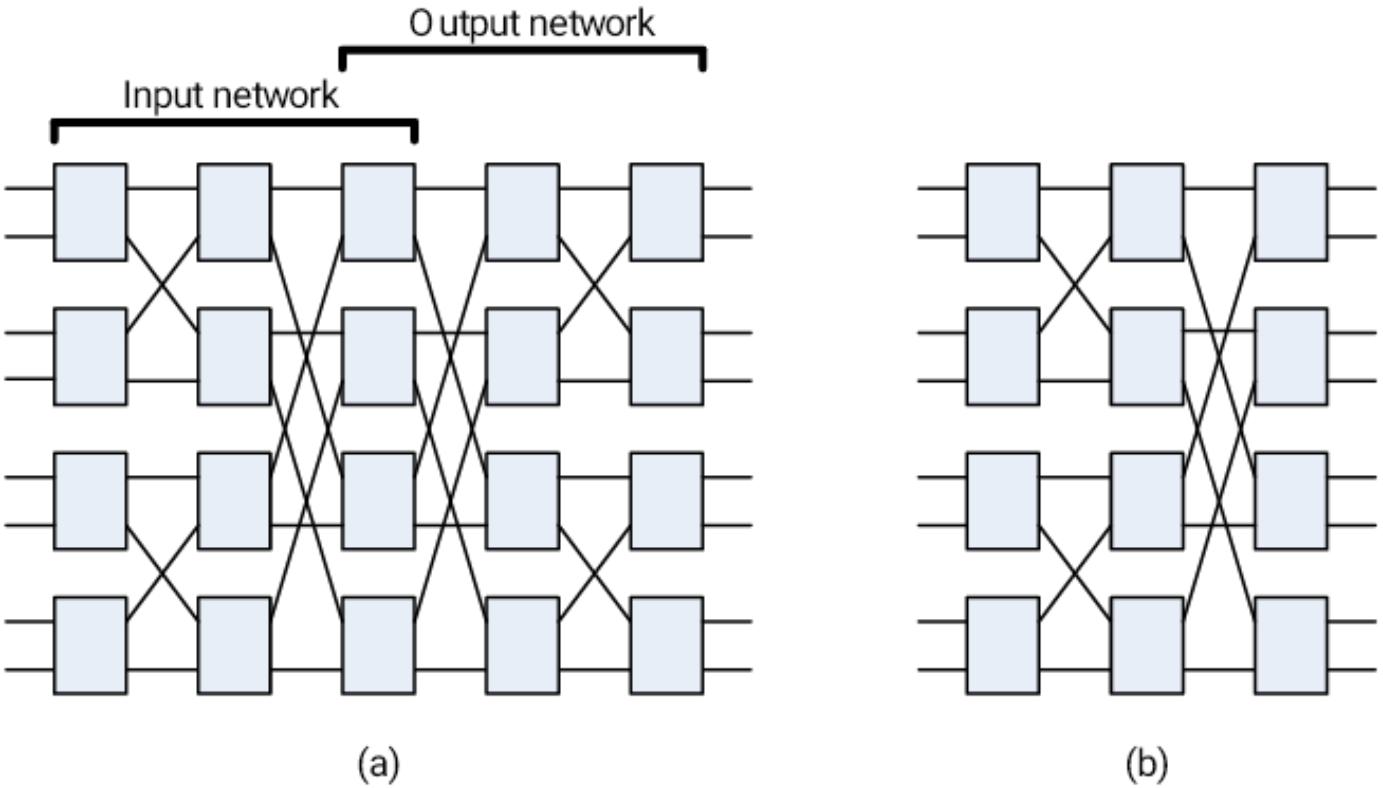
- The cost of routers and the number of cables interconnecting the routers are major components of the cost of interconnection network.
- Better performance and lower costs can only be achieved with innovative router architecture wire density has scaled up at a slower rate than processor speed and wire delay has remained constant .
- Router – switch interconnecting several networks.
  - low-radix routers – have a small number of ports; divide the bandwidth into a smaller number of wide ports.
  - high-radix routers - have a large number of ports; divide the bandwidth into larger number of narrow ports
- The number of intermediate routers in high-radix networks is reduced; lower latency and reduced power consumption.
- The pin bandwidth of the chips used for switching has increased by approximately an order of magnitude every 5 years during the past two decades.

Network characterization

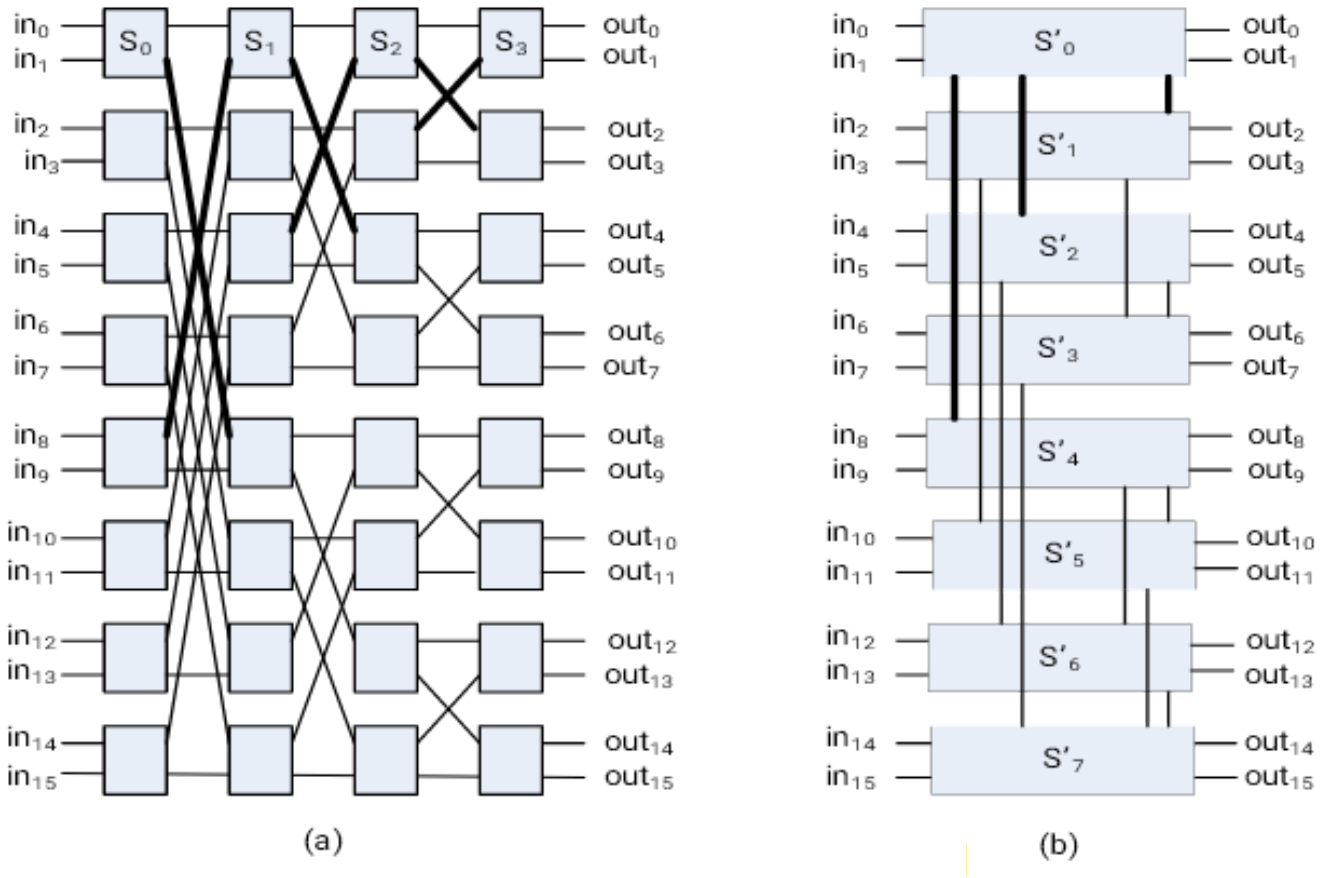
- The diameter of a network is the average distance between all pairs of nodes; if a network is fully-connected its diameter is equal one.
- When a network is partitioned into two networks of the same size, the bisection bandwidth measures the communication bandwidth between the two.
- The cost.
- The power consumption.

Clos networks

- Butterfly network the name comes from the pattern of inverted triangles created by the interconnections, which look like butterfly wings.
    - Transfers the data using the most efficient route, but it is blocking, it cannot handle a conflict between two packets attempting to reach the same port at the same time.
- Clos Multistage nonblocking network with an odd number of stages.
  - Consists of two butterfly networks. The last stage of the input is fused with the first stage of the output.
  - All packets overshoot their destination and then hop back to it; most of the time, the overshoot is not necessary and increases the latency, a packet takes twice as many hops as it really needs.
- Folded Clos topology the input and the output networks share switch modules. Such networks are called fat tree.
  - Myrinet, InfiniBand, and Quadrics implement a fat-tree topology.



- a) A 5-stage Clos network with radix-2 routers and unidirectional channels; the network is equivalent to two back-to-back butterfly networks.
- (b) The corresponding folded-Clos network with bidirectional channels; the input and the output networks share switch modules.

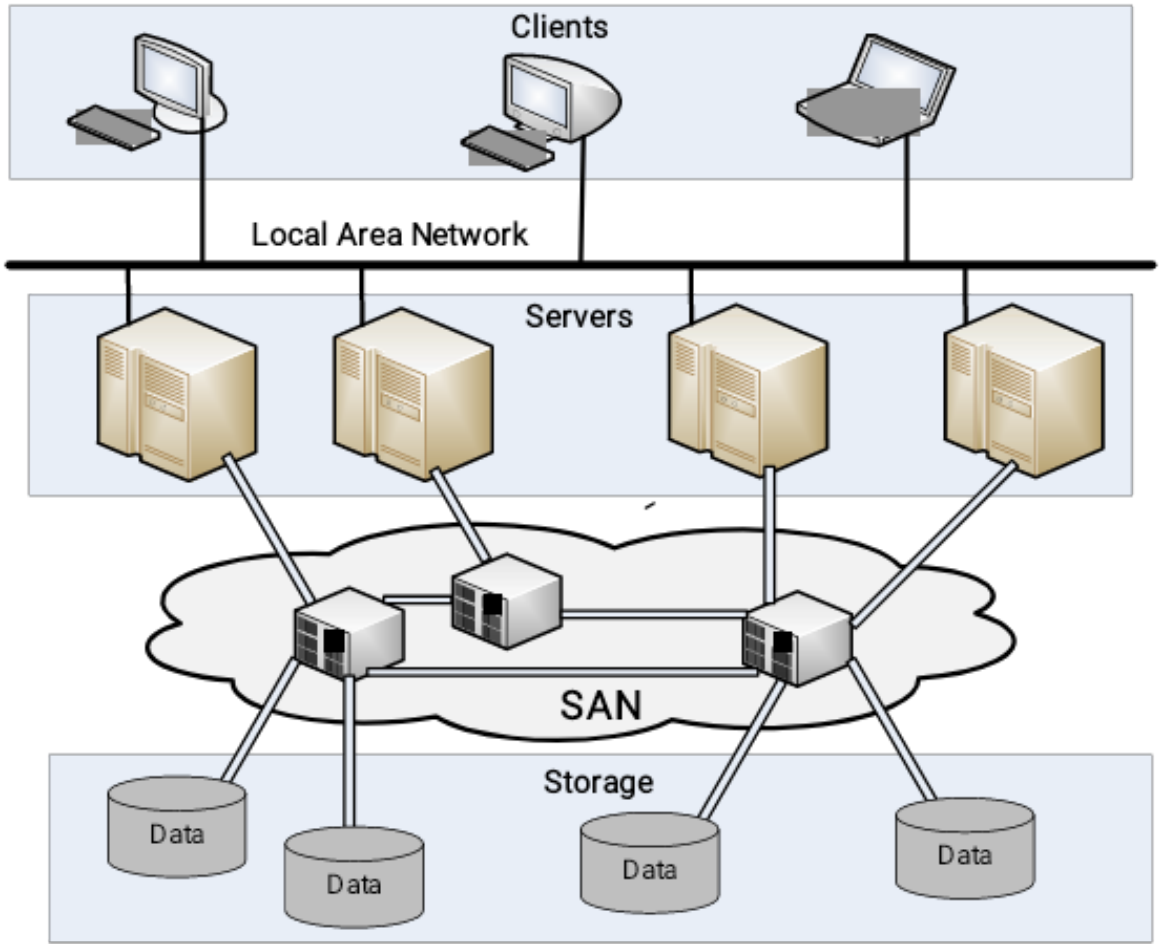


(a) A 2-ary 4-fly butterfly with unidirectional links.

(b) The corresponding 2-ary 4-flat flattened butterfly is obtained by combining the four switches S_0, S_1, S_2, and S_3, in the first row of the traditional butterfly into a single switch S_0', and by adding additional connections between switches

Storage area networks

- Specialized, high-speed network for data block transfers between computer systems and storage elements.
- Consists of a communication infrastructure and a management layer.
- The Fiber Channel (FC) is the dominant architecture of SANs.
- FC it is a layered protocol.

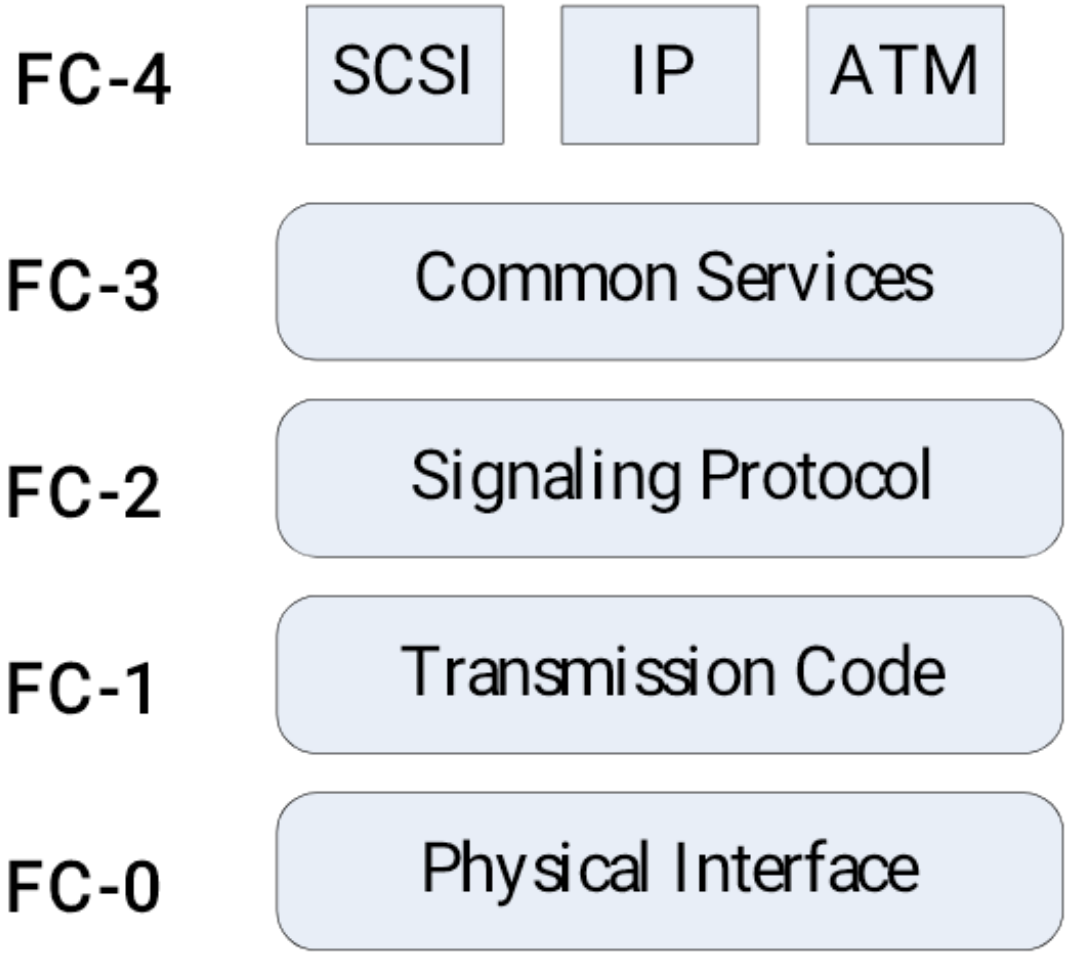


A storage area network interconnects servers to servers, servers to storage devices, and storage devices to storage devices.

FC protocol layers

- Three lower-layer protocols: FC-0, the physical interface; FC-1, the transmission protocol responsible for encoding/decoding; and FC-2, the signaling protocol responsible for framing and flow control.
  - FC-0 uses laser diodes as the optical source and manages the point-to-point fiber connections.
  - FC-1 controls the serial transmission and integrates data with clock information.
  - FC-2 handles the topologies, the communication models, the classes of service, sequence and exchange identifiers, and segmentation and reassembly.

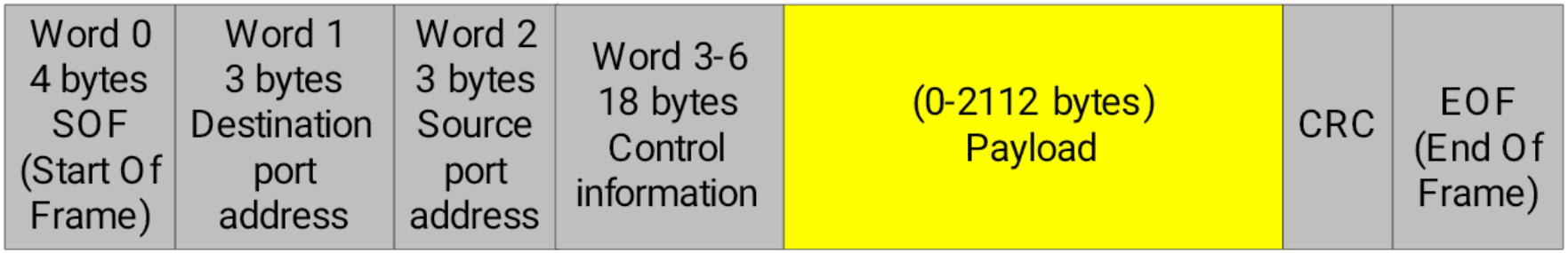
- Two upper-layer protocols:
  - FC-3 is common services layer.
  - FC-4 is the protocol mapping layer.



FC (Fiber Channel) protocol layers

FC classes of service

- Class 1 – rarely used blocking connection-oriented service.
- Class 2 – acknowledgments ensure that the frames are received; allows the fabric to multiplex several messages on a frame-by-frame basis; does not guarantee in-order delivery.
- Class 3 – datagram connection; no acknowledgments.
- Class 4 – connection-oriented service for multimedia applications; virtual circuits (VCs) established between ports, in-order delivery, acknowledgment of delivered frames; the fabric is responsible for multiplexing frames of different VCs. Guaranteed QoS, bandwidth and latency.
- Class 5 – isochronous service for applications requiring immediate delivery, without buffering.
- Class 6 – supports dedicated connections for a reliable multicast.
- Class 7 – similar to Class 2, used for the control and management of the fabric; connectionless service with notification of non-delivery.



The format of a FC frame

FC networks

- An FC device has a unique id called the WWN (World Wide Name), a 64 bit address, the equivalent of the MAC address.
- Each port in the switched fabric has its own unique 24-bit address consisting of: the domain (bits 23 - 16), the area (bits 15 - 08), and the port physical address (bits 07-00).
- A switch assigns dynamically and maintains the port addresses.
- When a device with a WWN logs into the switch on a port, the switch assigns the port address to that device and maintains the correlation between that port address and the WWN address of the device using a Name Server.
- The Name Server is a component of the fabric operating system, running on the switch.

Content delivery networks (CDNs)

- CDNs are designed to support scalability, to increase reliability and performance, and to provide better security. In 2013, Internet video is expected to generate over 18 exabytes of data per month.
- A CDN receives the content from an origin server, then replicates it to its edge cache servers; the content is delivered to an end-user from the "closest" edge server.
- A CDN can deliver static content and/or live or on-demand streaming media.
  - Static content - media that can be maintained using traditional caching technologies as changes are infrequent. Examples: HTML pages, images, documents, software patches, audio and video files.
  - Live media - live events when the content is delivered in real time from the encoder to the media server.
- Protocols used by CDNs: Network Element Control Protocol (NECP), Web Cache Coordination Protocol (WCCP), SOCKS, Cache Array Routing Protocol (CARP), Internet Cache Protocol (ICP), Hypertext Caching Protocol (HTCP), and Cache Digest.

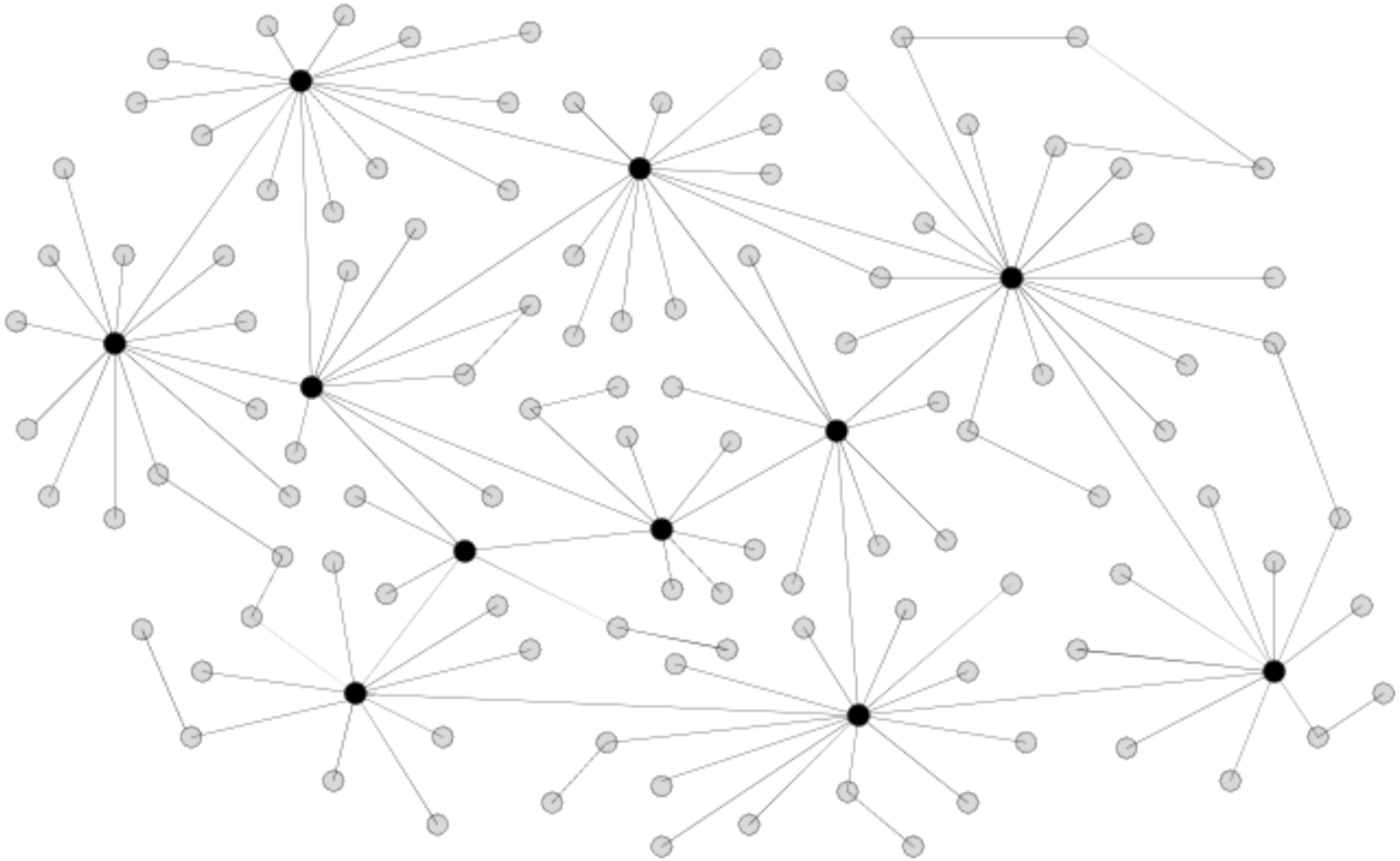
CDN design and performance

- Design and policy decisions for a CDNs.
  - The placement of the edge servers.
  - The content selection and delivery.
  - The content management.
  - The request routing policies.

- Critical metrics for CDN performance
  - Cache hit ratio - the ratio of the number of cached objects versus total number of objects requested.
  - Reserved bandwidth for the origin server.
  - Latency - based on the perceived response time by the end users.
  - Edge server utilization.
  - Reliability - based on packet-loss measurements.

Scale-free networks

- The degree distribution of scale-free networks follows a power law.
- Many physical and social systems are interconnected by a scale-free network. Empirical data available for power grids, the web, the citation of scientific papers, or social networks confirm this trend.
- The majority of the vertices of a scale-free network:
  - Are directly connected with the vertices with the highest degree.
  - Have a low degree and only a few vertices are connected to a large number of edges.



A scale-free network is non-homogeneous; the majority of vertices have a low degree and only a few vertices are connected to a large number of edges; the majority of the vertices are directly connected with the highest degree ones.

Epidemic algorithms

- Epidemic algorithm mimic the transmission of infectious diseases and are often used in distributed systems to accomplish tasks such as:
  - disseminate information, e.g., topology information.
  - compute aggregates, e.g., arrange the nodes in a gossip overlay into a list sorted by some attributes in logarithmic time.
  - manage data replication in a distributed system.
- *Game of life* is a popular epidemic algorithm invented by John Conway.
- Several classes of epidemic algorithms exist. The concepts used to classify these algorithms
  - Susceptible (S),
  - Infective (I),
  - Recovered (R)

  refer to the state of the population subject to infectious disease and, by extension, to the recipient of information in a distributed system.

Types of epidemic algorithms

- Susceptible-Infective (SI) algorithms apply when the entire population is susceptible to be infected; once an individual becomes infected it remains in that state until the entire population is infected.

- Susceptible-Infectious-Recover (SIR) based on the model developed by Kermack and McKendrik which assumes
  - the following transition from one state to another S I R;
  - that the size of the population is fixed (S(t) + I(t) + R(t) =N.

- Susceptible-Infective-Susceptible (SIS) algorithms are particular cases of SIR models when individuals recover from the disease without immunity. If p=R(r)/I(r), then the number of newly infected grows until (1-p)/2 are infected and then decreases exponentially to (1-p).