Introduction to Cloud Computing

Systems II

15-319, spring 2010

6th Lecture, Jan 28th

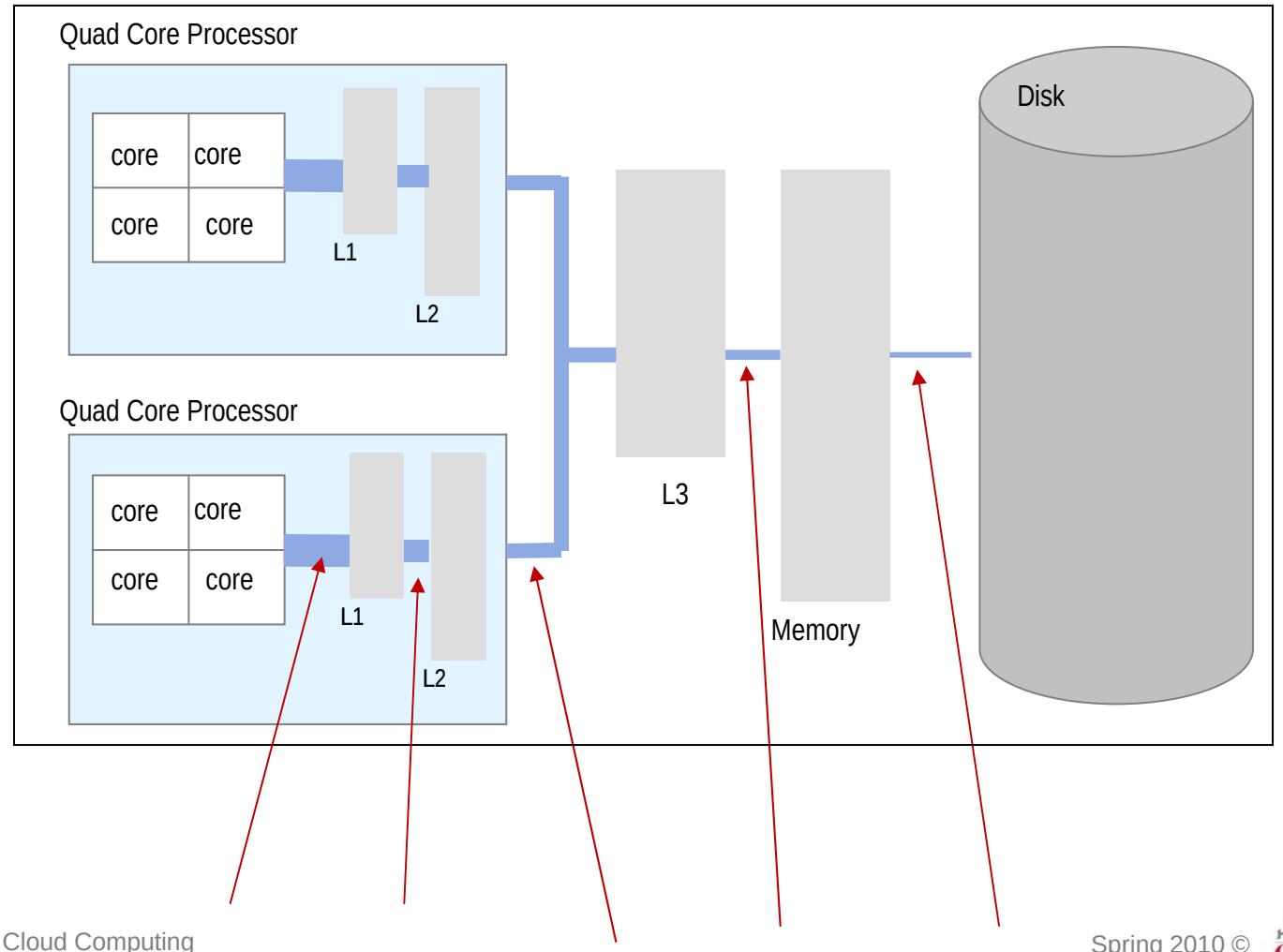
Majd F. Sakr

Lecture Motivation

- System Impact on Performance
- System Considerations
- Performance Evaluation

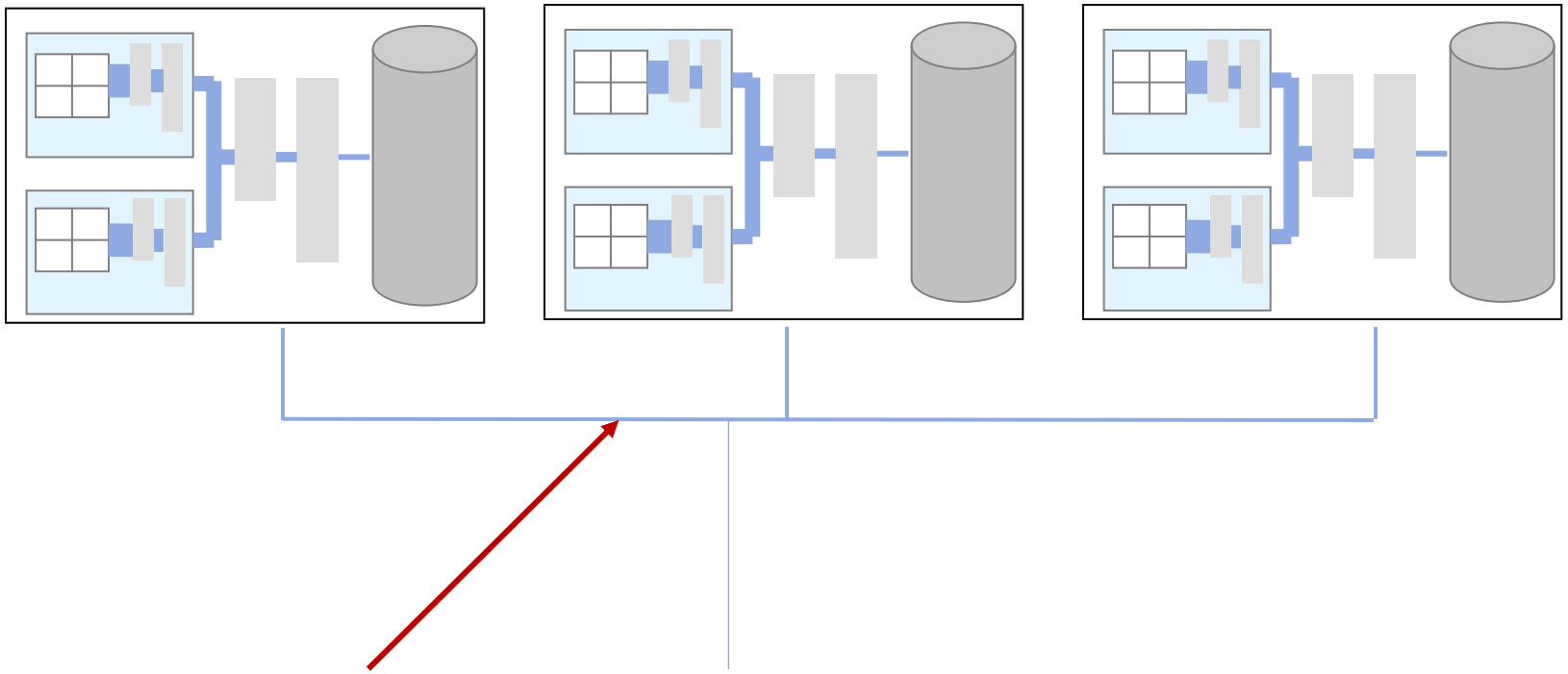
Performance Bottlenecks

- Consider bandwidth and latency between these layers



Performance Bottlenecks

- **Consider bandwidth and latency of all layers**



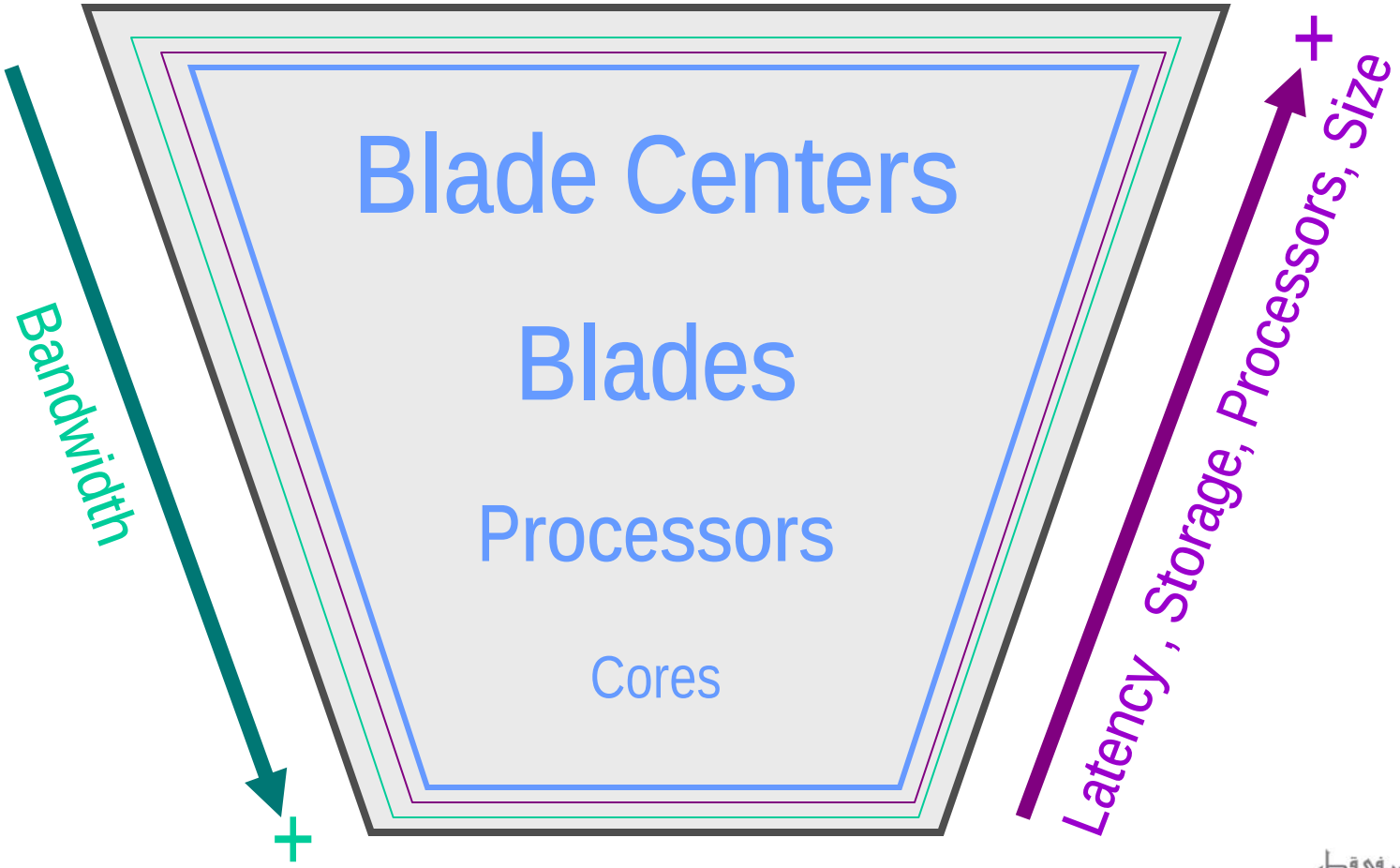
Performance Bottlenecks

- Consider bandwidth and latency of all of these pipes

How about a network between different blade centers??

System Design Performance Considerations

Where does your application live? across….



System Design Performance Considerations

- Number of blade Centers
- Number of blades in each Blade Center
- Number of processors in each Blade or Motherboard
- Number of cores in each processor
- Processor Frequency
- Per core:
  - Order of superscalars
  - Number of Registers
  - Bandwidth between register
  - File and the Functional Unit
- Memory design
- Bandwidth between all these components

Blade Centers

- **A blade center OR group of blade centers could be a “cloud”**
- **This # affects the blade centers’ networking issues:**
  - More bandwidth & less latency means “better communication”
  - If the blade centers are connected to form a “cloud” for intensive computation:
    - Processes are tightly coupled: There is a lot of communication to achieve cooperative work
    - Maximizing network performance would maximize the cloud efficiency and productivity

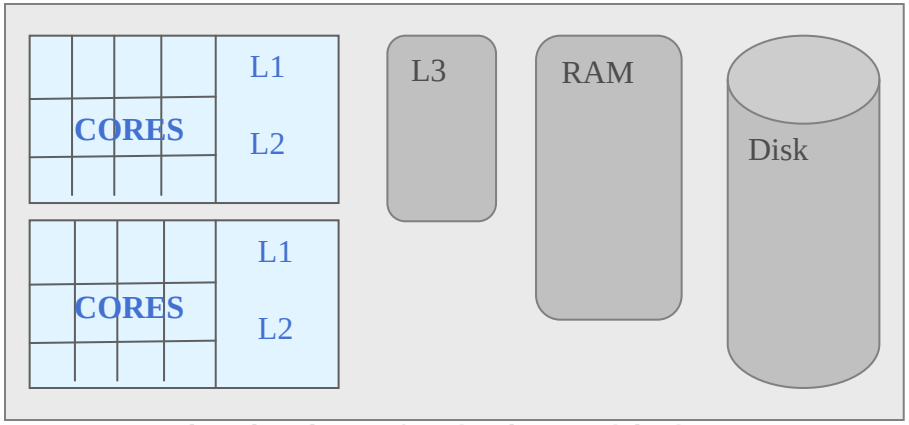
Blades/Blade Center

- **Again it is a matter of intra-blade center networking**
  - bandwidth and latency issues in the link connecting the blades, like the *backplane*

Processors/Blade

- **Each Processor has cores, L1 & L2 caches.**
  - Both caches connected to/share/retrieve/write data to same L3 cache.
  - The network pipe between processors & L3 cache requires bandwidth and latency good enough to serve all processors.

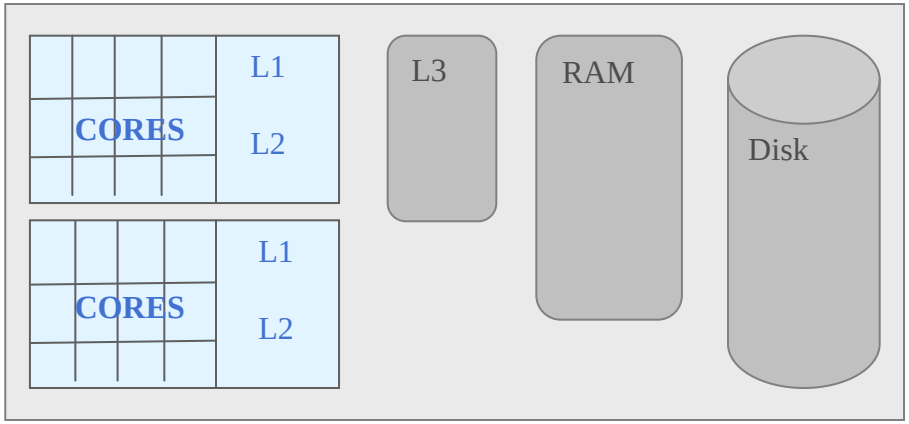
- **Also, A delay comes with needing a connection/internal bus between processors**
  - Especially if they were to do parallel processing and exchange data.
  - Having the data on the same processor chip would have saved this communication & data transformation delay.



Simple view of a dual-core blade

Cores/Processor

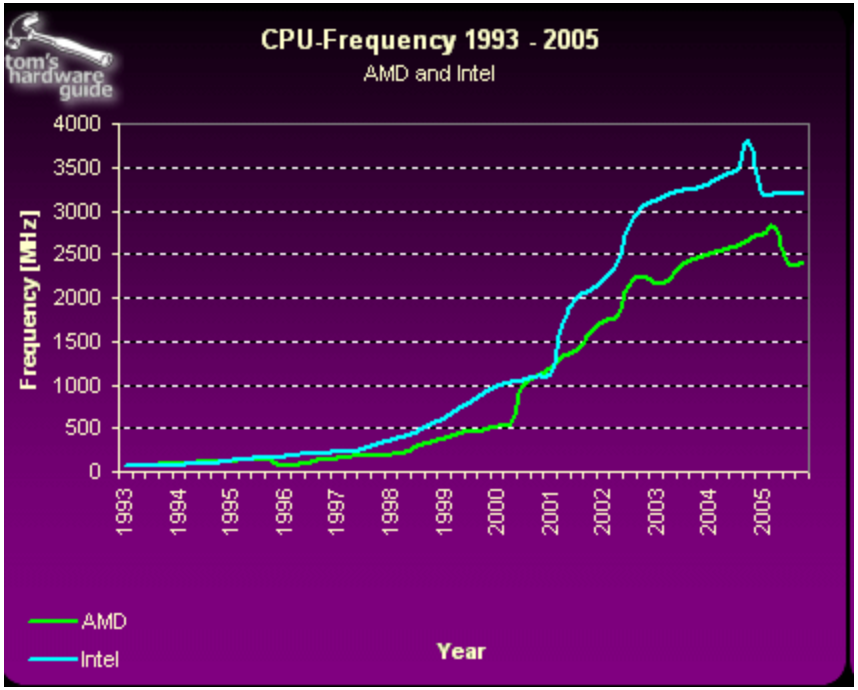
- **Each processor has cores that share L1 cache & L2 cache.**
  - They are shared resources between the cores.
  - Again bandwidth & latency of the network pipes connecting the cores to the caches affects the overall functionality.
- **There are single-core, dual-core, quad-core , …. processors**
- **A blade with multiple processors is called** *Symmetric Multi-processor Machine (SMP)*



Simple view of a dual-core blade

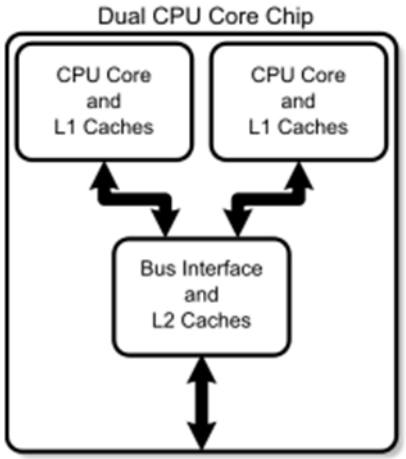
Why a Multi-core Processor?

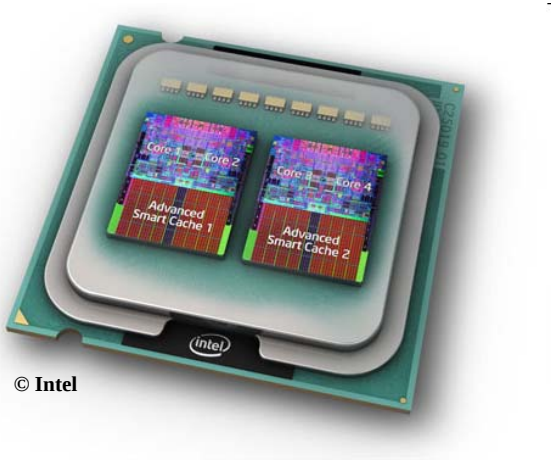
- **Problem: How much faster can a processor get?**
  - Towards the early 2000's we hit a "frequency wall" for processors.
  - Processor frequencies topped out at 4 GHz due to the thermal limit of Silicon.



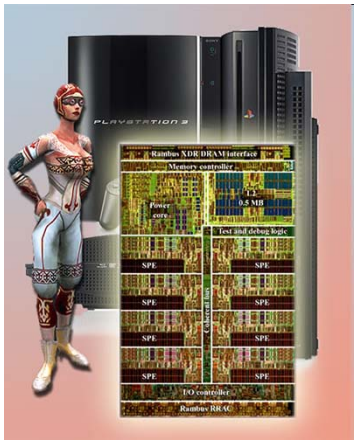
Why a Multi-core Processor?

- **Solution : More chips in Parallel!**
  - Enter Multicore.
  - Cram more processing units into the chip rather than increase clock frequency.
  - Traditional processors are available with up to Quad and Six Cores.
  - The Cell Broadband Engine (Playstation 3) has 9 cores.
  - Intel recently announced a 48-core processor.



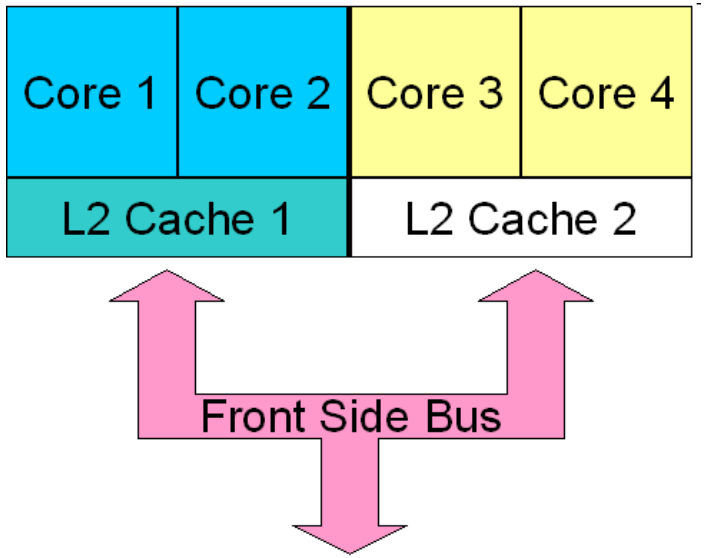


© Intel



Multicore processors

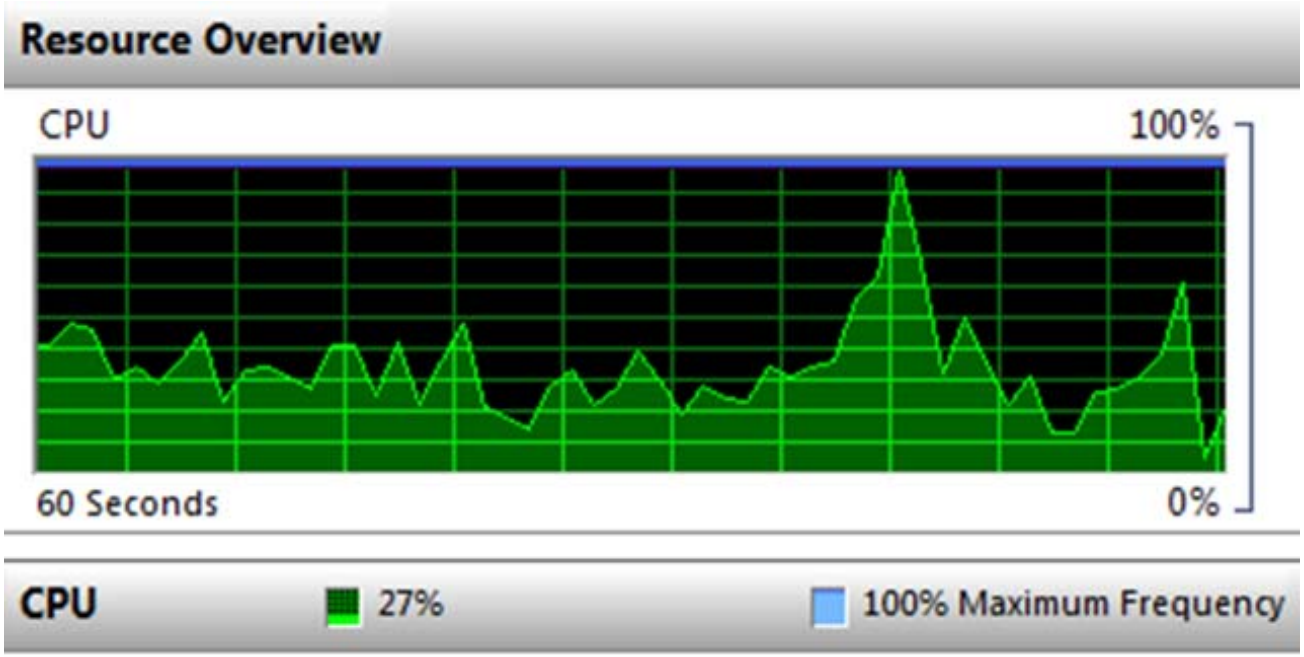
- Have fully functioning processor “cores” in a processor.
- Have individual L1 and shared L2 caches.
- OS and applications see each core as an individual processor.
- Applications have to be specifically rewritten for optimized performance.



Source: hardwaresecrets.com

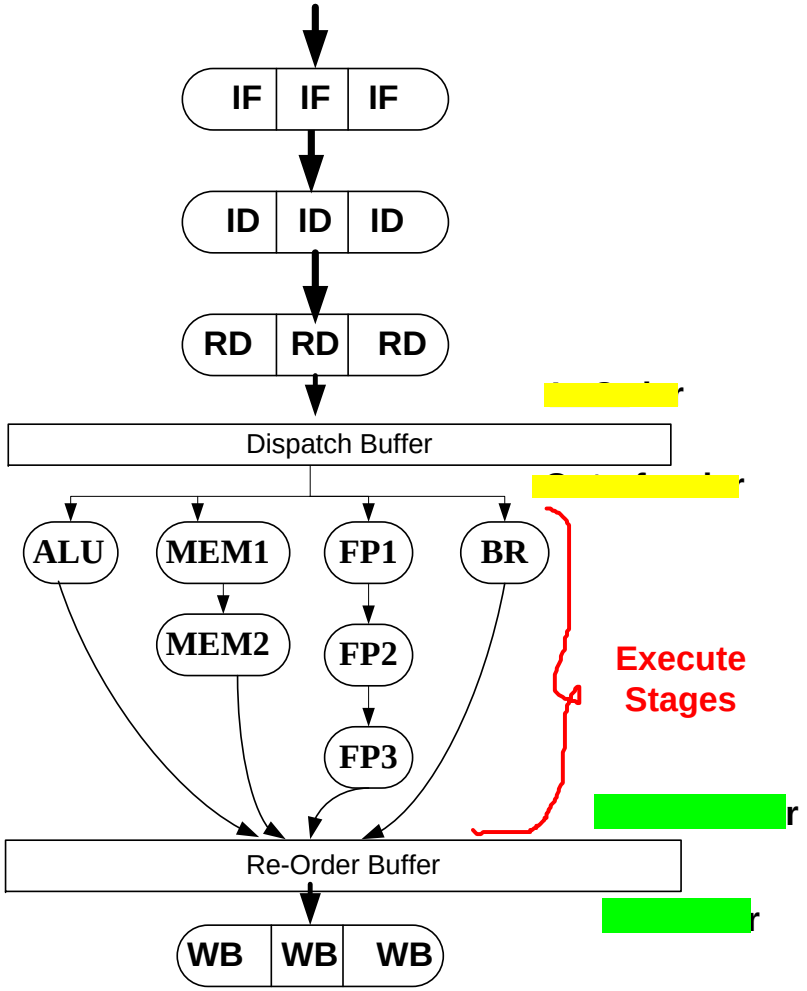
Design Concerns: processor frequency

- **Frequency**
  - It is the clock speed of the processor
  - The faster the clock speed, the more instruction the processor can execute at a given time



Core Design (1/4)

- Order of superscalars
  - Superscalar of a core is divided at levels:
  - A level-x core is the one with x functional units.
  - The more functional units:
    - the more parallelism the system can do &
    - the less is the time spent on execution.



Core Design (2/4)

- **Number of Registers**
  - **Registers hold temporary values that are needed by the processor to finish the current instruction it is executing.**
  - **The more the number of registers:**
    - **the less the (register spill) &**
    - **the less instructions there are &**
    - **the less the memory accesses.**

Core Design (3/4)

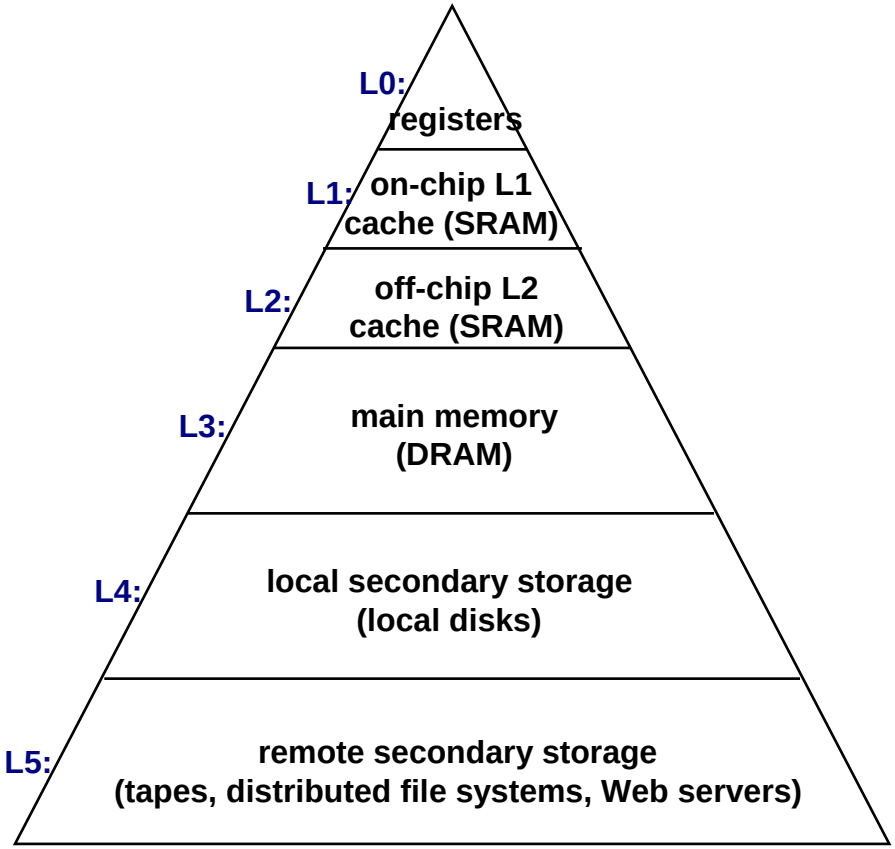
- **Number of Registers**
  - **Example:**
    - **Suppose we have 2 registers R1 and R2 and 2 instructions: (add b,a,c) and (mult d, a, c)**
    - **Execution:**
      - Load a in R1 → Load c in R2
      - 1) Now: need to save result (b) into a register. We have a ***Register Spill***.
        - » Solution: free one of the registers by storing its value to memory;
        - » say we free R1: Store R1 → Put b in R1
      - 2) Now: to execute the second instruction: Load (a) from memory again in R1, for example.
  - **Problem:** there is fewer # of registers than # of instructions to execute and we face **Register Spill** (i.e. there is not enough registers for the values)
    - we end up having more instructions that load and store from/to memory.
  - **More instructions = delay in execution time & more memory load/store (EXPENSIVE!!!!).**

Core Design (4/4)

- **Bandwidth between register file and the Functional Unit**
  - This Bandwidth of the network, the number of buses, between the register file and the functional unit.
  - Needs to be big enough to utilize communication between the functional units and the register file.
  - Some ALUs may be assigned no jobs and they become a wasted resource.

Design Concerns

- **Why do we need to worry about the following details when we design a core?**
  - **L1 Cache**
    - Size
    - Bandwidth between (processor & L1 Cache)
    - Latency
  - **L2 Cache**
    - Size
    - Bandwidth between (L1 & L2 Caches)
    - Latency
  - **L3 Cache**
    - Size
    - Bandwidth between (L2 & L3 Caches)
    - Latency
  - **RAM memory**
    - Size
    - Bandwidth between (L3 Cache & RAM)
    - Latency
  - **Disk**
    - Size
    - Bandwidth between (RAM & Disk)
    - Latency

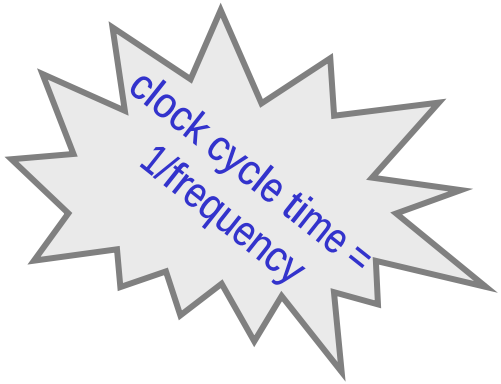


Design Concerns

- **Answer: depending on the application, the properties of the different memory components need to be specified.**

- **A transaction application's needs of resources is very different from those of scientific application.**
  - Transaction applications are usually **Disk Bound**: most of the data is stored on Disk.
    - Requirements: fast disk. High bandwidth and low latency from/to disk and memory.
- **Scientific applications are usually** processor Bound**: dependent on speed of the central processor.**
    - Requirements: fast processor & fast L1 cache access.

Example: Given This Data.....

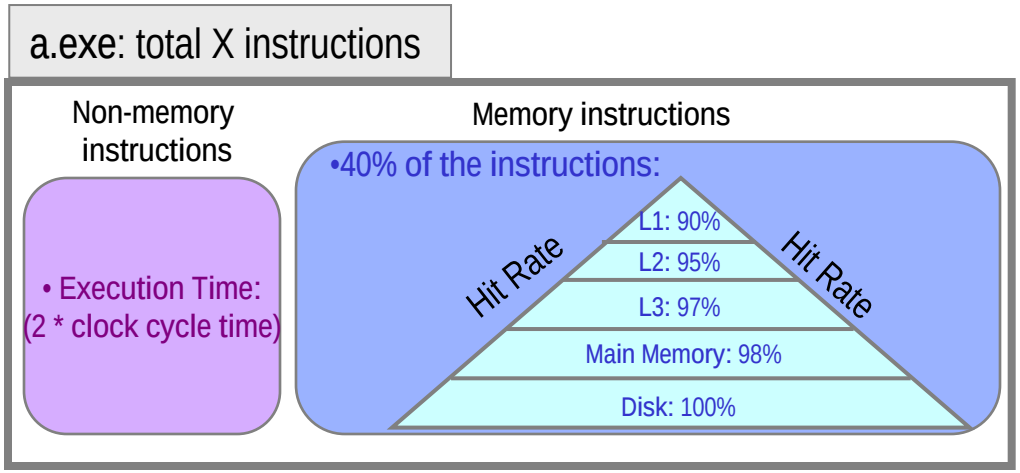


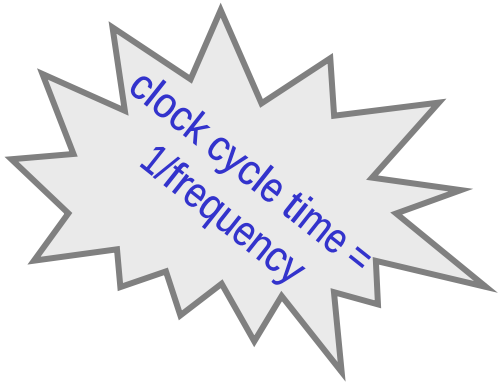
Time = (Instructions * Ave Clocks Per Instruction) * Clock Cycle Time

If 40% of all instructions are memory instructions, then:

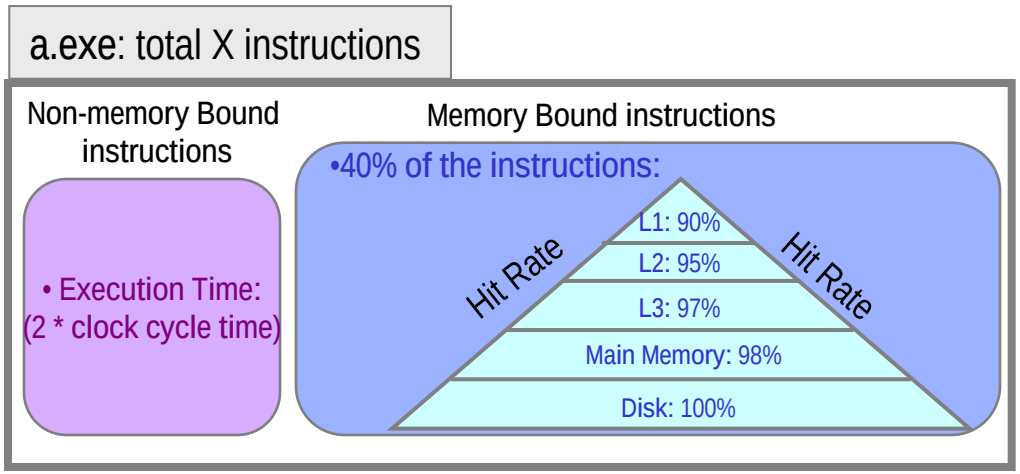
Time = (0.6(Instructions) * Ave Clocks Per Other Instruction +
0.4(Instructions) * Ave Clocks Per Memory Instruction) *
Clock Cycle Time

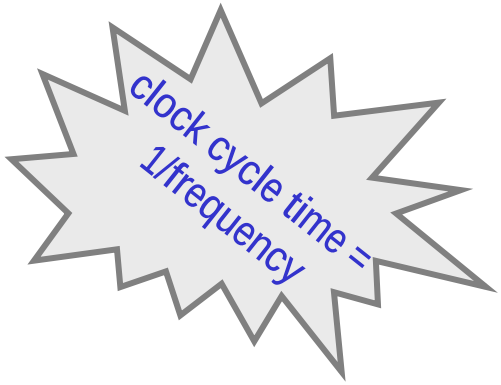
Example: Given This Data…..

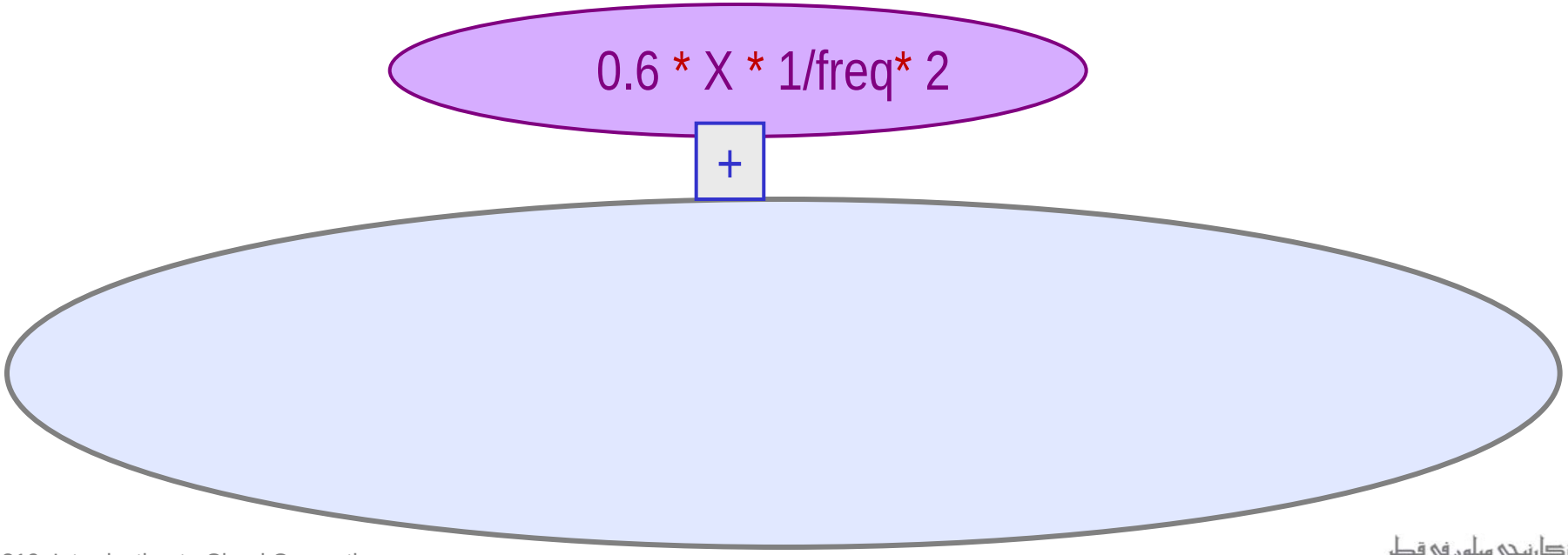




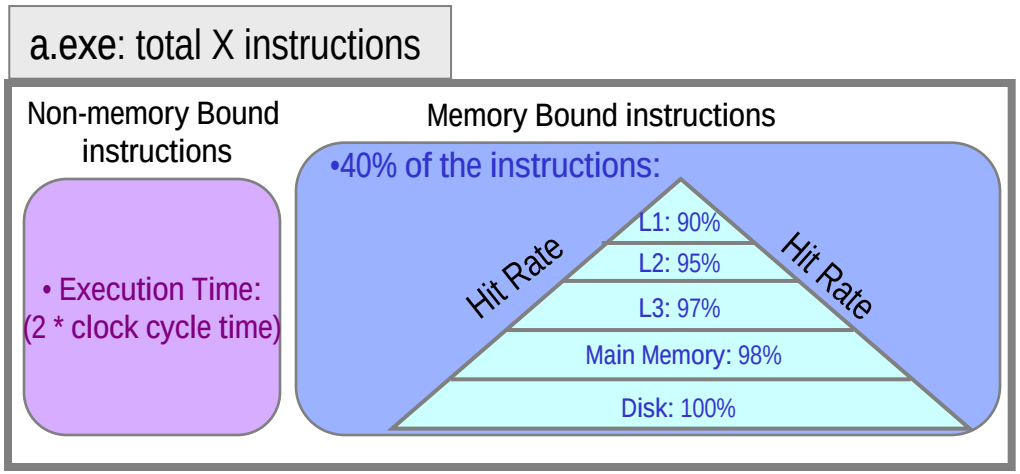
Example: Calculate Execution Time for a.exe

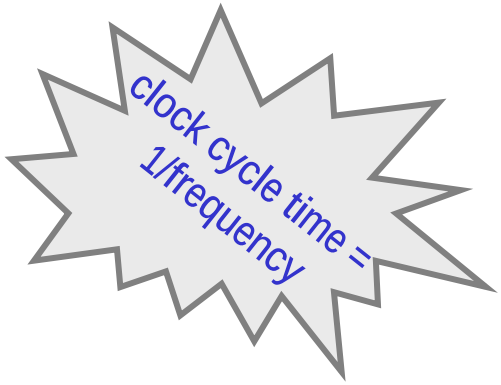


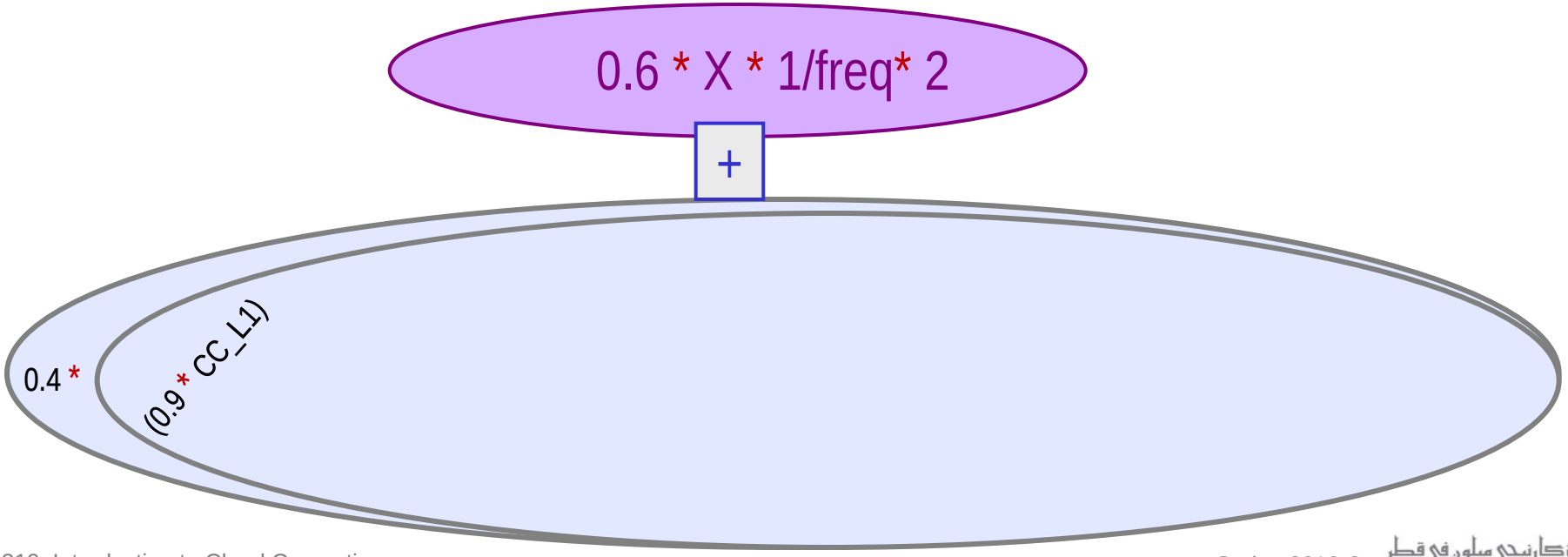




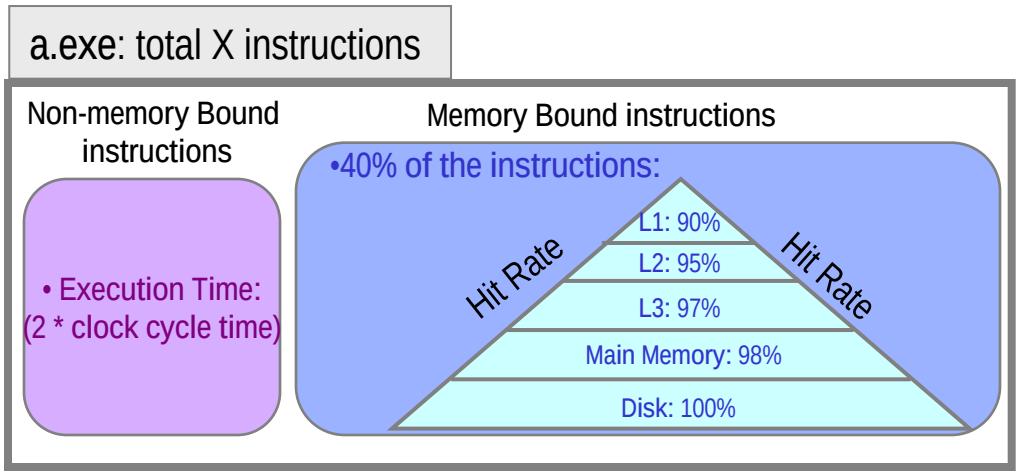
Example: Calculate Execution Time for a.exe

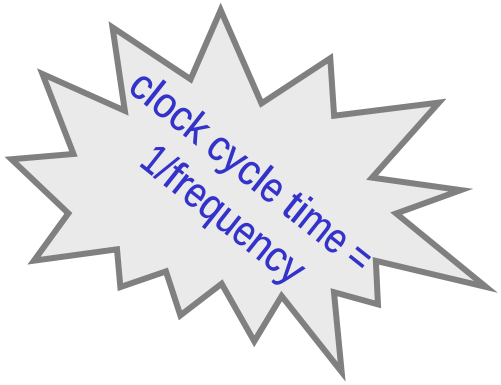


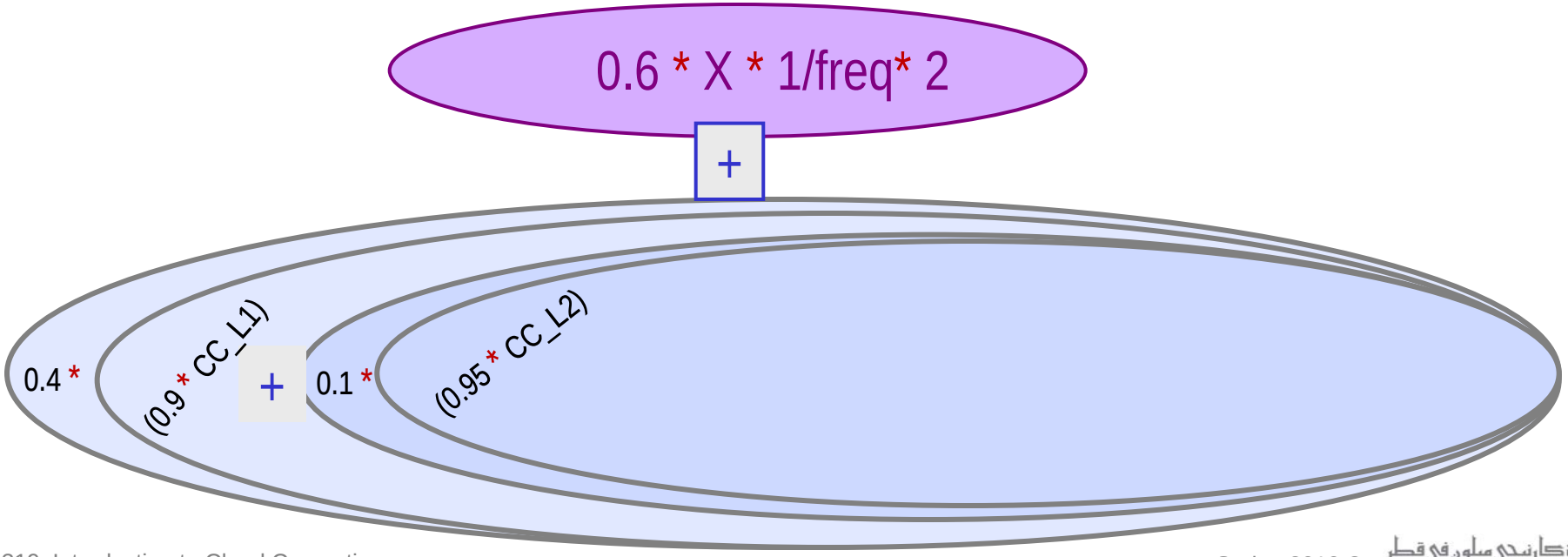




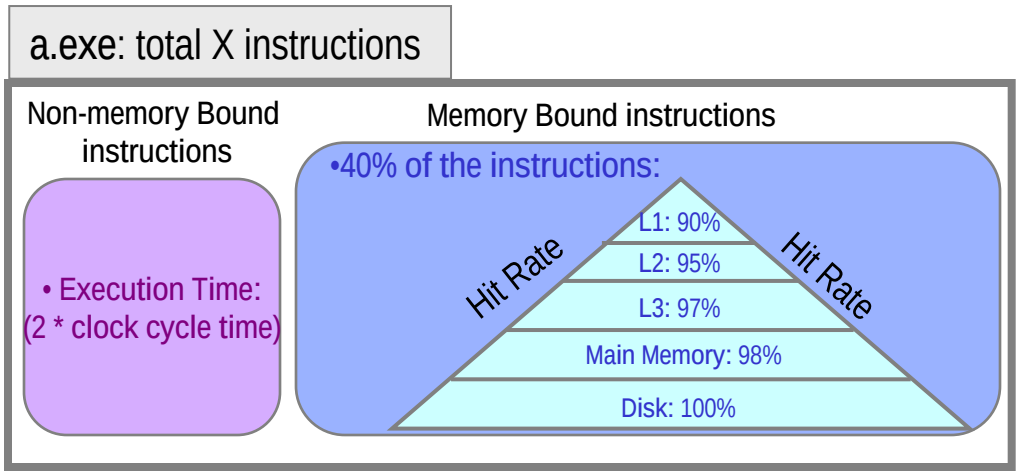
Example: Calculate Execution Time for a.exe

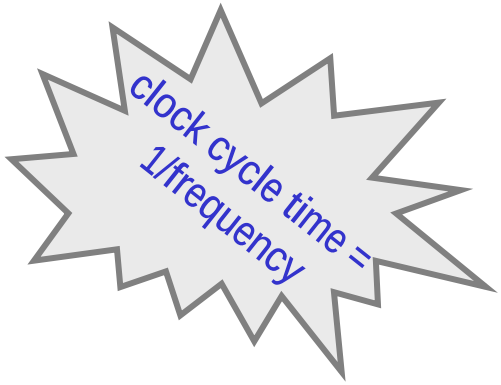


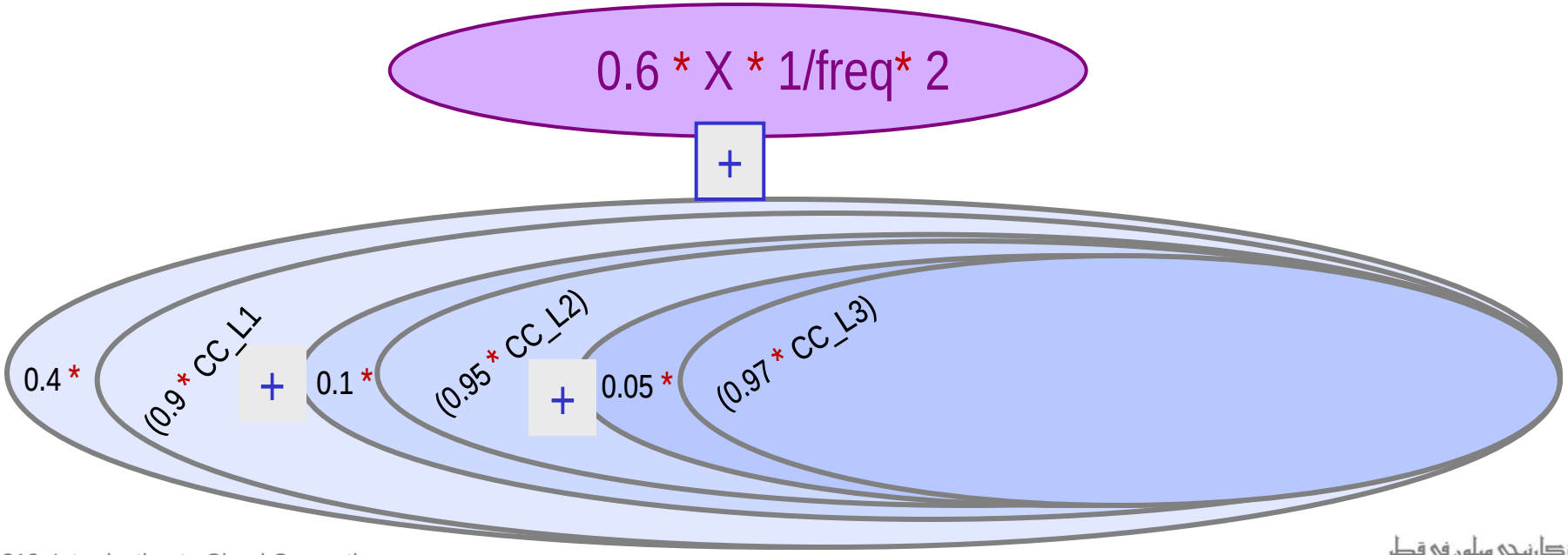




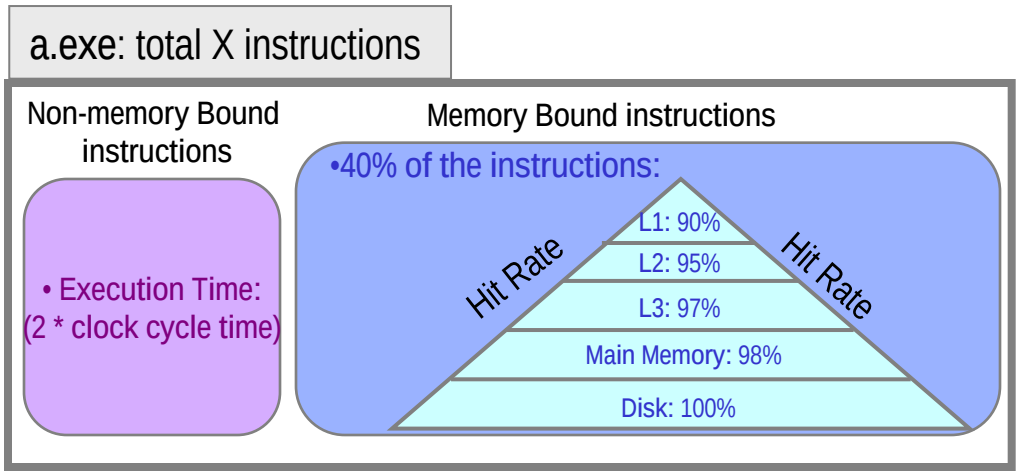
Example: Calculate Execution Time for a.exe

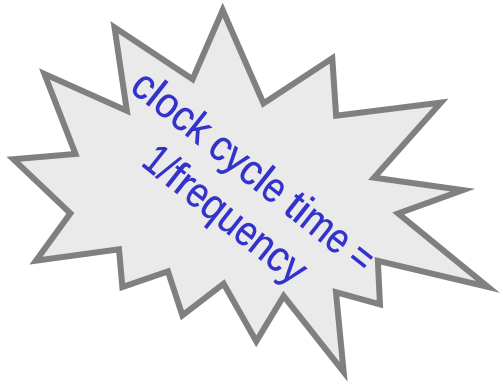


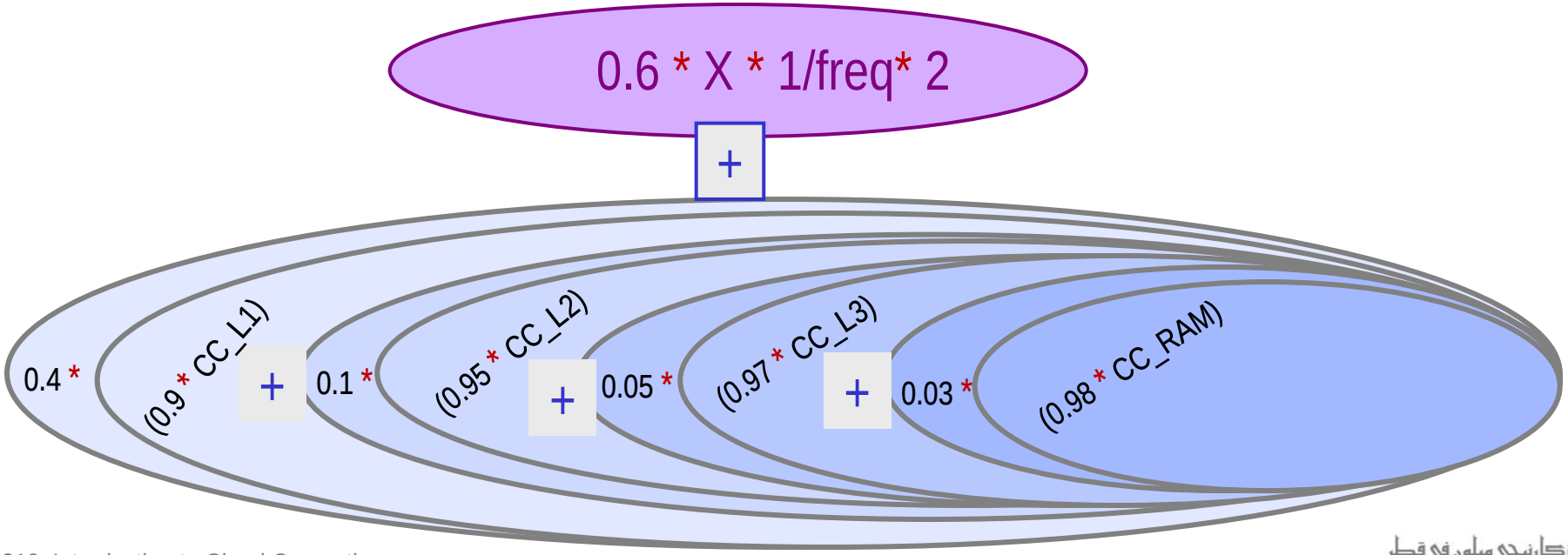




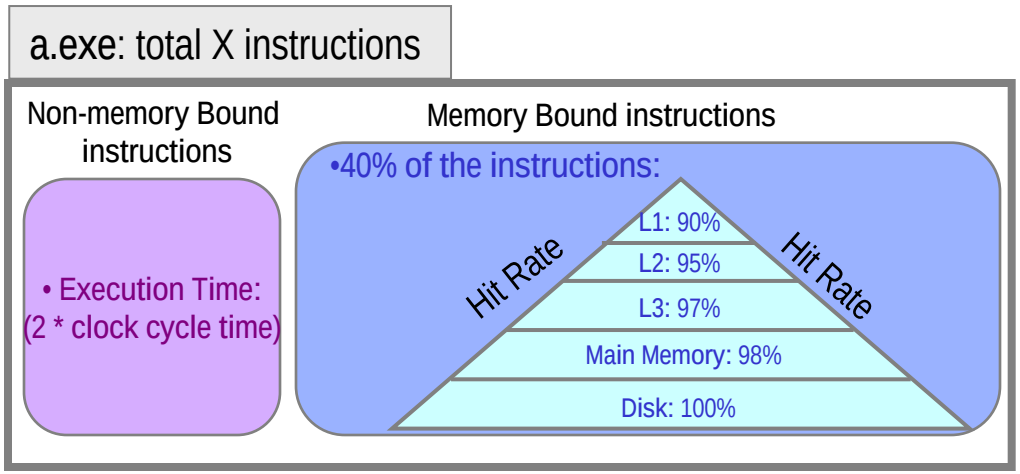
Example: Calculate Execution Time for a.exe

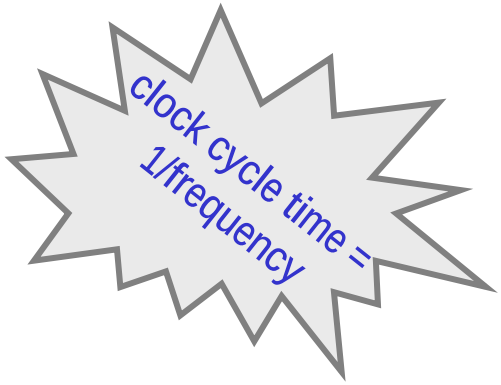


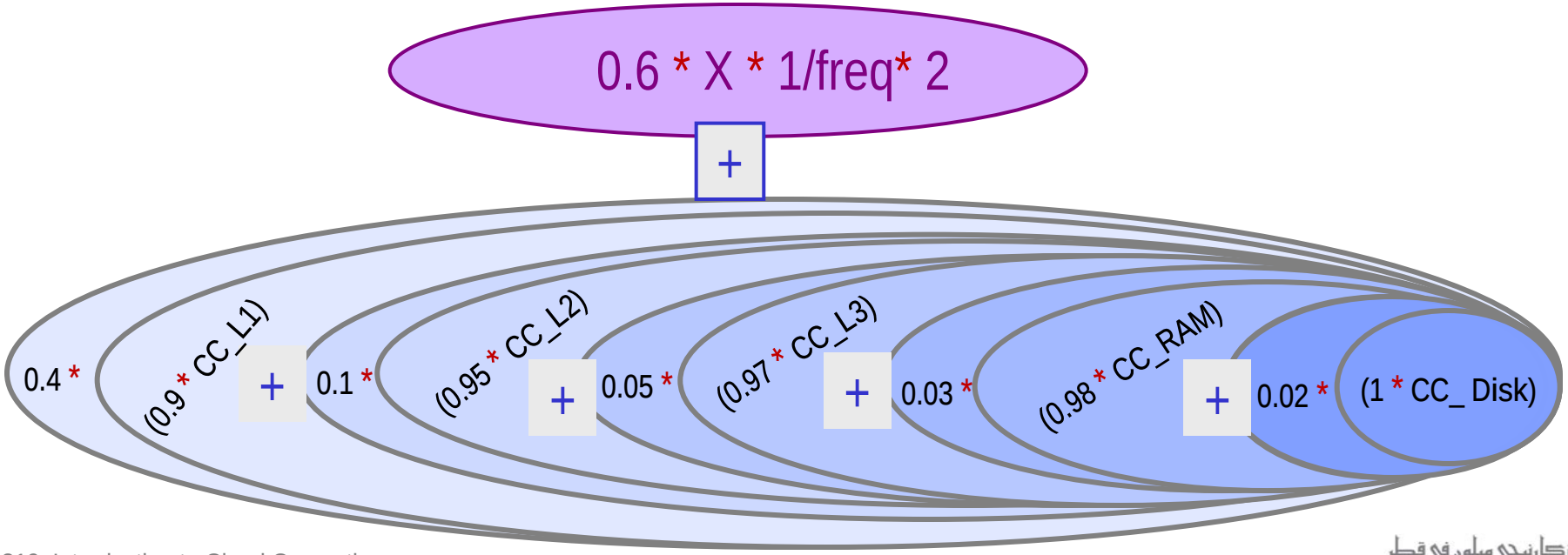




Example: Calculate Execution Time for a.exe







Design Concerns

- **As the example demonstrates a process execution time is dependent on the speed of the different computer components**

- **Different applications could be:**
  - Disk bound
  - Memory bound
  - processor bound

- **The system design should serve the application accordingly to reduce the computation time and utilize resources usage**

Towards Faster Networks

- Remember the memory hierarchy?
- Networks are THE slowest form of data transfer.
- Researchers sent 4 GB of data faster using a pigeon and a thumb drive than using broadband

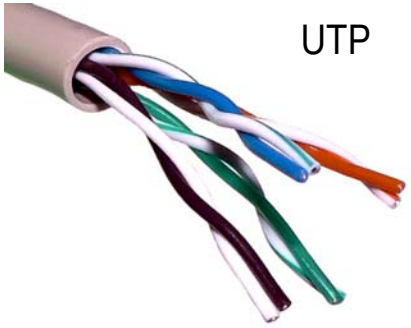
BBC News, SA pigeon 'faster than broadband'
http://news.bbc.co.uk/2/hi/8248056.stm

High BW & Speed Networks

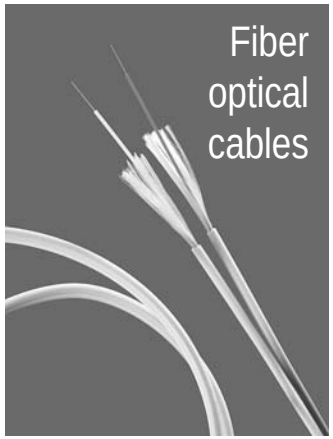
- **Server and cluster backbones typically need fast interconnects**
- **Gigabit Ethernet**
  - 10 Gigabit
  - 100 Gigabit
- **Myrinet**
- **Infiniband**

Gigabit Ethernet

- Known as “IEEE Standard 802.3z”
- Offers 1 Gbps raw bandwidth
- Speed: (10 x speed of fast Ethernet)
  (100 x speed of regular Ethernet)
- 1 Gig Ethernet uses UTP cables
- 10 Gig Ethernet and 100 Gig Ethernet are emerging technologies, typically require fiber optical cables



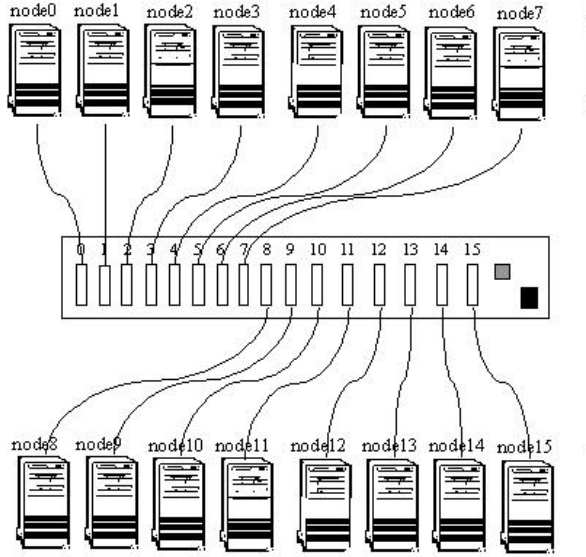
commons.wikimedia.org/wiki/File:UTP_cable.jpg



http://www.directindustry.com/prod/lapp-group/fiber-optic-cable-17287-404578.html

Myrinet

- High-speed Local Area Network Interconnect
- Typically requires two fiber optic cables per node (upstream and downstream)
- Offers low-latency networking with low protocol overhead @ 1.9 Gbps
- Next Generation (Myri-10G) is 10 Gbps.



Infiniband

- High-bandwidth interconnect primarily for processors to high performance I/O devices
- InfiniBand offers point-to-point bidirectional serial links which forms a switched fabric
- Upto 120 Gbps theoretical bandwidth