# Chapter 6 – Cloud Resource Management and Scheduling



#### Resource management and scheduling

- Critical function of any man-made system.
- It affects the three basic criteria for the evaluation of a system:
  - **III** Functionality.
  - **Performance**.
  - **Cost.**
- Scheduling in a computing system >>> deciding how to allocate resources of a system, such as CPU cycles, memory, secondary storage space, I/O and network bandwidth, between users and tasks.
- Policies and mechanisms for resource allocation.
  - □ Policy → principles guiding decisions.

#### **Motivation**

- Cloud resource management.
  - □□ Requires complex policies and decisions for multi-objective optimization.
  - It is challenging the complexity of the system makes it impossible to have accurate global state information.
  - ☐ Affected by unpredictable interactions with the environment, e.g., system failures, attacks.
  - ☐ Cloud service providers are faced with large fluctuating loads which challenge the claim of cloud elasticity.
- The strategies for resource management for laaS, PaaS, and SaaS are different.

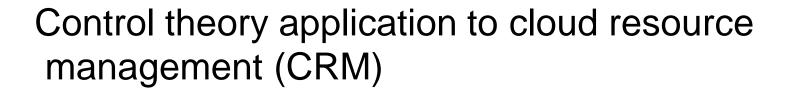
### Cloud resource management (CRM) policies

- Admission control >>>prevent the system from accepting workload in violation of high-level system policies.
- 2. Capacity allocation  $\rightarrow$ allocate resources for individual activations of a service.
- 3. Load balancing <del>>></del>distribute the workload evenly among the servers.
- 4. Energy optimization ->> minimization of energy consumption.
- 5. Quality of service (QoS) guarantees  $\rightarrow$ ability to satisfy timing or other conditions specified by a Service Level Agreement.



### Mechanisms for the implementation of resource management policies

- Control theory Duses the feedback to guarantee system stability and predict transient behavior.
- <u>Machine learning</u> → does not need a performance model of the system.
- <u>Utility-based</u> → require a performance model and a mechanism to correlate user-level performance with cost.
- <u>Market-oriented/economic</u> → do not require a model of the system, e.g., combinatorial auctions for bundles of resources.



- The main components of a control system:
  - The inputs → the offered workload and the policies for admission control, the capacity allocation, the load balancing, the energy optimization, and the QoS guarantees in the cloud.
  - □□ The control system components → sensors used to estimate relevant measures of performance and controllers which implement various policies.
  - $\square$  The outputs  $\rightarrow$ the resource allocations to the individual applications.



- Control granularity >>the level of detail of the information used to control the system.
  - □□ Fine control → very detailed information about the parameters controlling the system state is used.
  - □ Coarse control → the accuracy of these parameters is traded for the efficiency of implementation.
- The controllers use the feedback provided by sensors to stabilize the system. Stability is related to the change of the output.
- Sources of instability in any control system:
  - The delay in getting the system reaction after a control action.
  - The granularity of the control, the fact that a small change enacted by the controllers leads to very large changes of the output.
  - Oscillations, when the changes of the input are too large and the control is too weak, such that the changes of the input propagate directly to the output.



#### Resource bundling

- \* Resources in a cloud are allocated in bundles.
- ◆ Users get maximum benefit from a specific combination of resources: CPU cycles, main memory, disk space, network bandwidth, and so on.
- ★ Resource bundling complicates traditional resource allocation models and has generated an interest in economic models and, in particular, in auction algorithms.
- $\leftrightarrow$  The bidding process aims to optimize an objective function f(x,p).
- ♣ In the context of cloud computing, an auction is the allocation of resources to the highest bidder.

#### Cloud scheduling algorithms (1/2)

- Scheduling \(\rightarrow\righ
  - III A server can be shared among several virtual machines.
  - III A virtual machine could support several applications.
  - III An application may consist of multiple threads.
- A scheduling algorithm should be efficient, fair, and starvation-free.
- The objectives of a scheduler:
  - Batch system → maximize throughput and minimize turnaround time.
  - □□ Real-time system → meet the deadlines and be predictable.
- Best-effort: batch applications and analytics.
- Common algorithms for <u>best effort</u> applications:
  - **Round-robin.**
  - □ First-Come-First-Serve (FCFS).
  - □ Shortest-Job-First (SJF).
  - Priority algorithms.



#### Cloud scheduling algorithms (2/2)

- Multimedia applications (e.g., audio and video streaming)
  - III Have soft real-time constraints.
  - III Require statistically guaranteed maximum delay and throughput.
- Real-time applications have hard real-time constraints.
- Scheduling algorithms for real-time applications:
  - □□ Earliest Deadline First (EDF).
  - □□ Rate Monotonic Algorithms (RMA).
- Algorithms for integrated scheduling of several classes of applications (best-effort, multimedia, real-time):
  - □□ Resource Allocation/Dispatching (RAD) .
  - □□ Rate-Based Earliest Deadline (RBED).



#### Scheduling Policies

- First in, First out (FIFO) → The tasks are scheduled for execution in the order of their arrival.
- Earliest deadline first (EDF) → The task with the earliest deadline is scheduled first.
- Maximum workload derivative first (MWF) → The tasks are scheduled in the order of their derivatives, the one with the highest derivative first. The number n of nodes assigned to the application is kept to a minimum.

#### Workload Partitioning Rules

- Optimal Partitioning Rule (OPR) >>>the workload is partitioned to ensure the earliest possible completion time and all tasks are required to complete at the same time.
  - >> The head node distributes sequentially the data to individual worker nodes.
  - >> Worker nodes start processing the data as soon as the transfer is complete.
- Equal Partitioning Rule (EPR) → assigns an equal workload to individual worker nodes.
  - >> The head node distributes sequentially the data to individual worker nodes.
  - Worker nodes start processing the data as soon as the transfer is complete.
  - The workload is partitioned in equal segments.

#### м

## Scheduling MapReduce Applications subject to deadlines

#### Four commonly referenced scheduling are:

- The default FIFO schedule
- The Fair Scheduler
- The Capacity Scheduler
- The Dynamic Proportional Scheduler

#### They are all based upon the assumptions:

- 1. The system is homogeneous
- 2. Load equipartition