OPERATING SYSTEM

Multi-Processor & Real-time Scheduling

Dr Rahul Nagpal
Computer Science

OPERATING SYSTEM

Multi-Processor & Real-time Scheduling

Dr. Rahul Nagpal
Computer Science

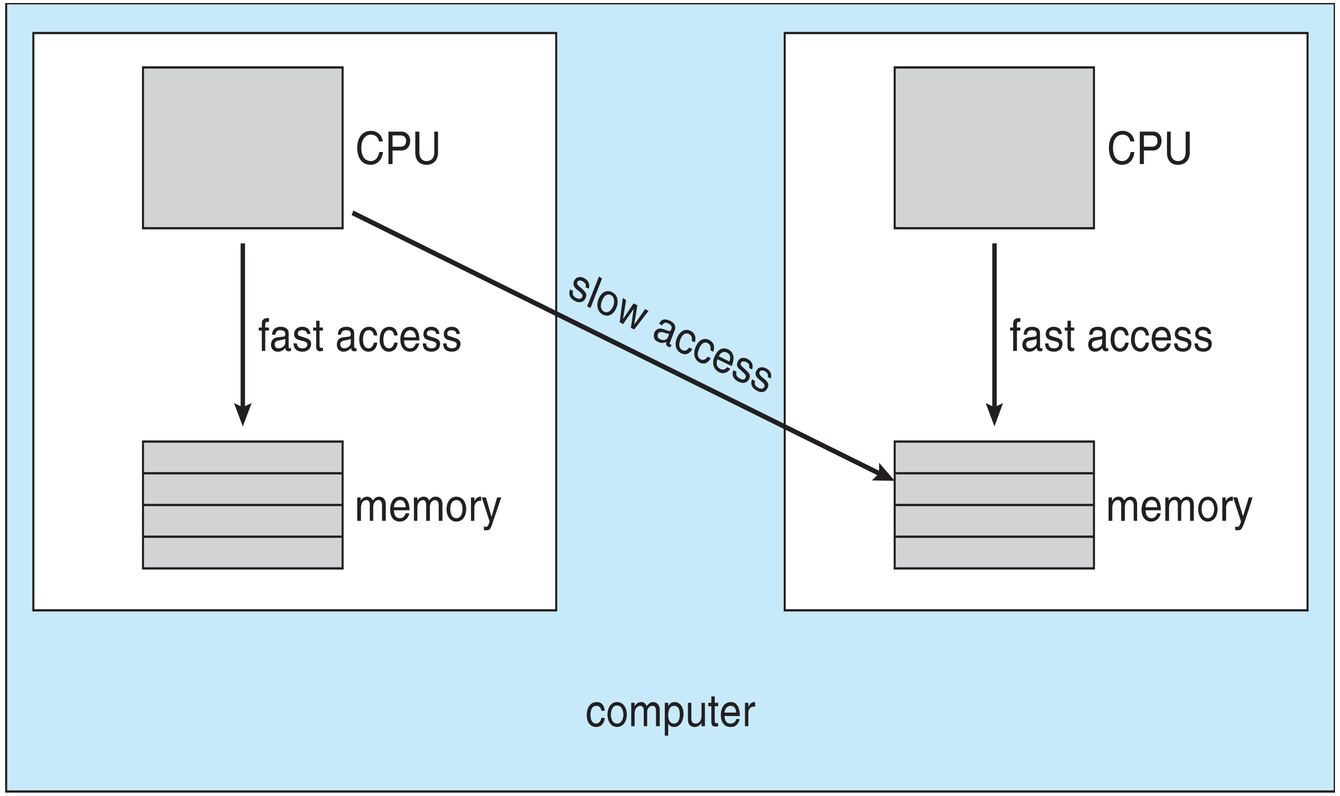
Outline

- Multiprocessor Scheduling
- Real-time Scheduling
  - Priority Based Scheduling
  - Rate Monotonic Scheduling
  - Earliest Deadline First Scheduling
  - Proportional Share Scheduling

Multiple-Processor Scheduling

- CPU scheduling more complex when multiple CPUs are available
- **Homogeneous processors** within a multiprocessor
- **Asymmetric multiprocessing** – only one processor accesses the system data structures, alleviating the need for data sharing
- **Symmetric multiprocessing (SMP)** – each processor is self-scheduling, all processes in common ready queue, or each has its own private queue of ready processes
  - Currently, most common
- **Processor affinity** – process has affinity for processor on which it is currently running
  - **soft affinity, hard affinity,** Variations including **processor sets**

NUMA and CPU Scheduling



Note that memory-placement algorithms can also consider affinity

Multiple-Processor Scheduling – Load Balancing

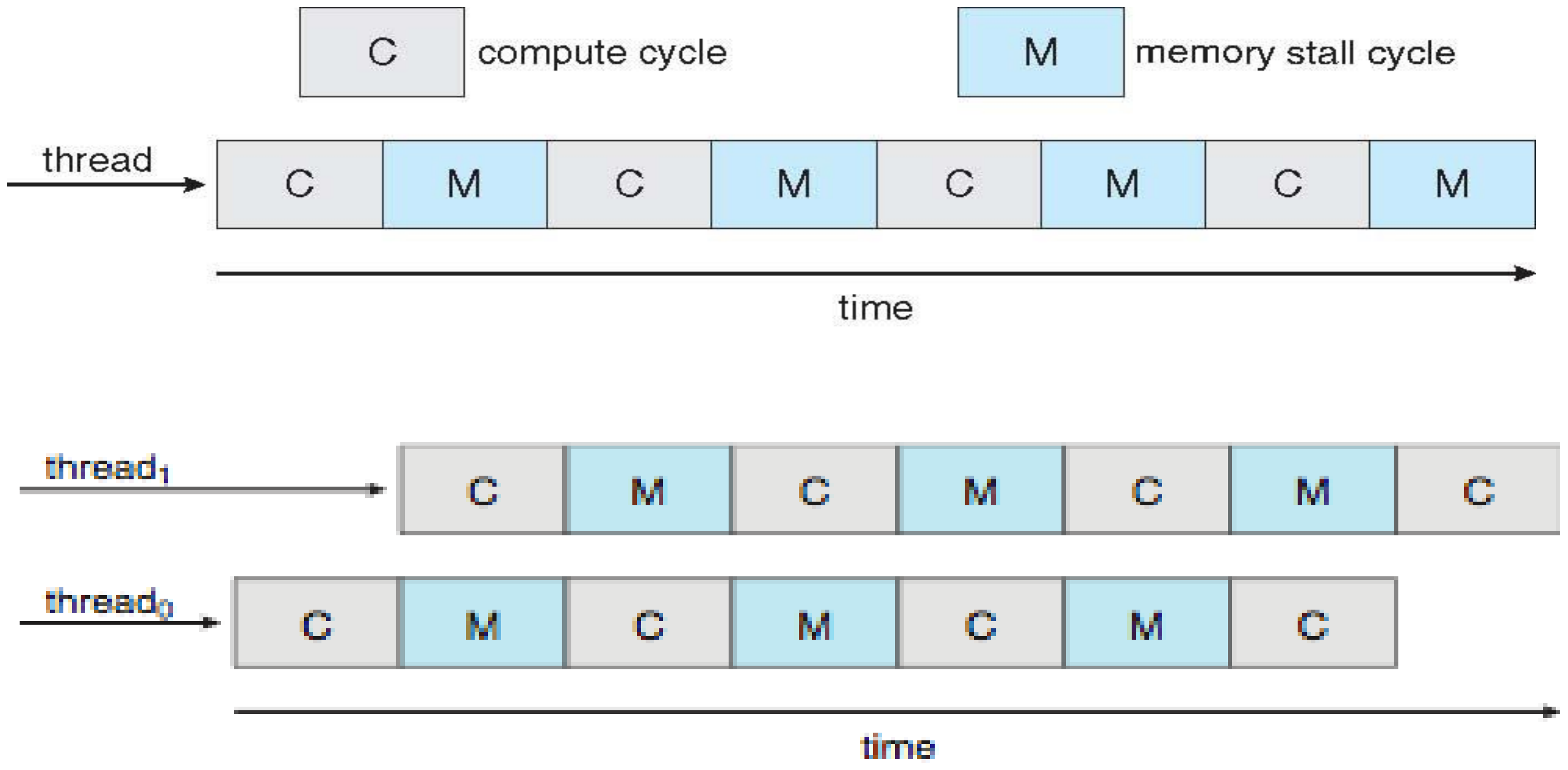
- If SMP, need to keep all CPUs loaded for efficiency
- **Load balancing** attempts to keep workload evenly distributed
- **Push migration** – periodic task checks load on each processor, and if found pushes task from overloaded CPU to other CPUs
- **Pull migration** – idle processors pulls waiting task from busy processor

Multicore Processors

- Recent trend to place multiple processor cores on same physical chip
- Faster and consumes less power
- Multiple threads per core also growing
  - Takes advantage of memory stall to make progress on another thread while memory retrieve happens

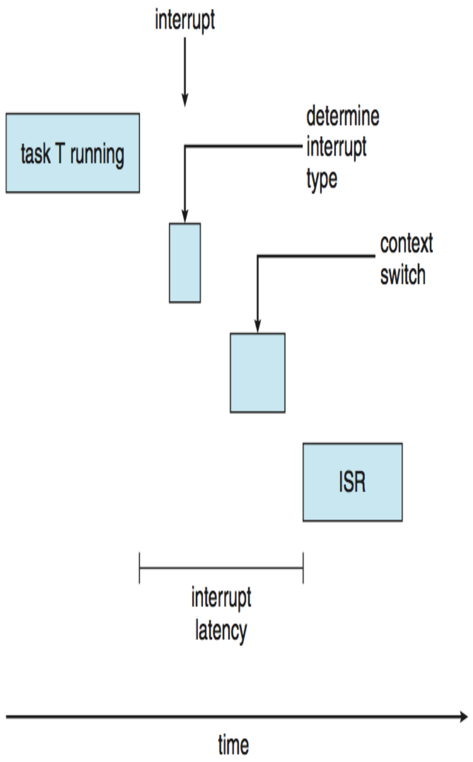
OPERATING SYSTEMS - Scheduling

Multithreaded Multicore System



Real-Time CPU Scheduling

- Can present obvious challenges
- **Soft real-time systems** – no guarantee as to when critical real-time process will be scheduled
- **Hard real-time systems** – task must be serviced by its deadline
- Two types of latencies affect performance
    1. Interrupt latency – time from arrival of interrupt to start of routine that services interrupt
    2. Dispatch latency – time for schedule to take current process off CPU and switch to another

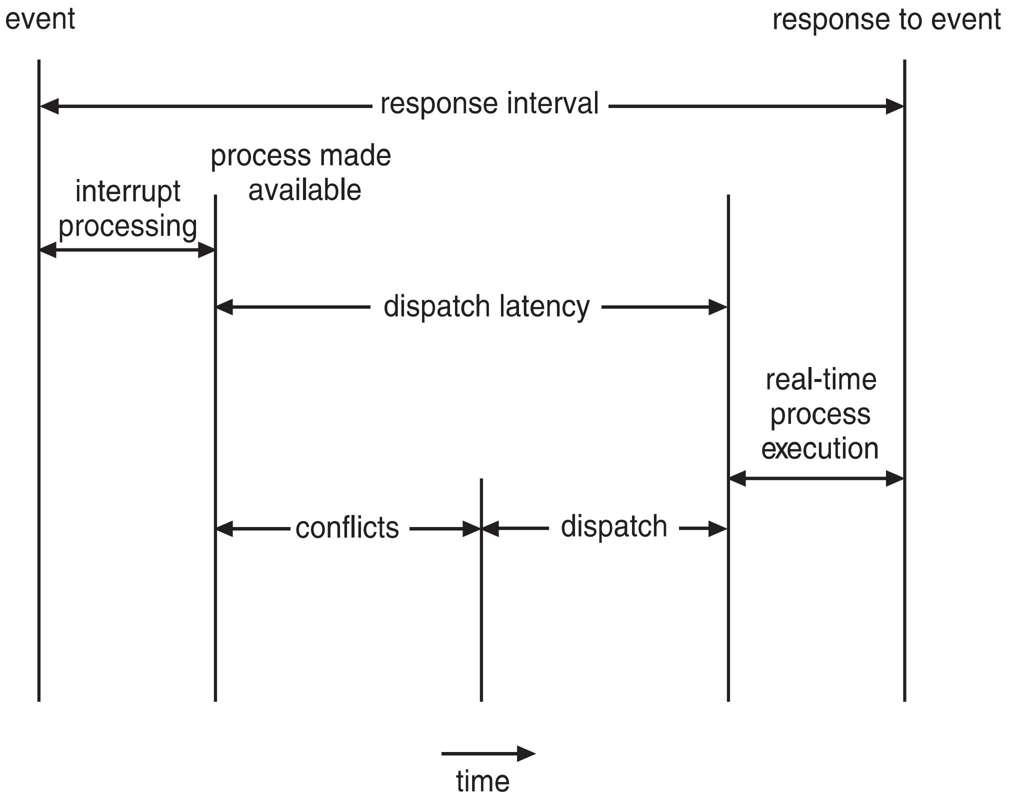


Real-Time CPU Scheduling (Cont.)

- Conflict phase of dispatch latency:

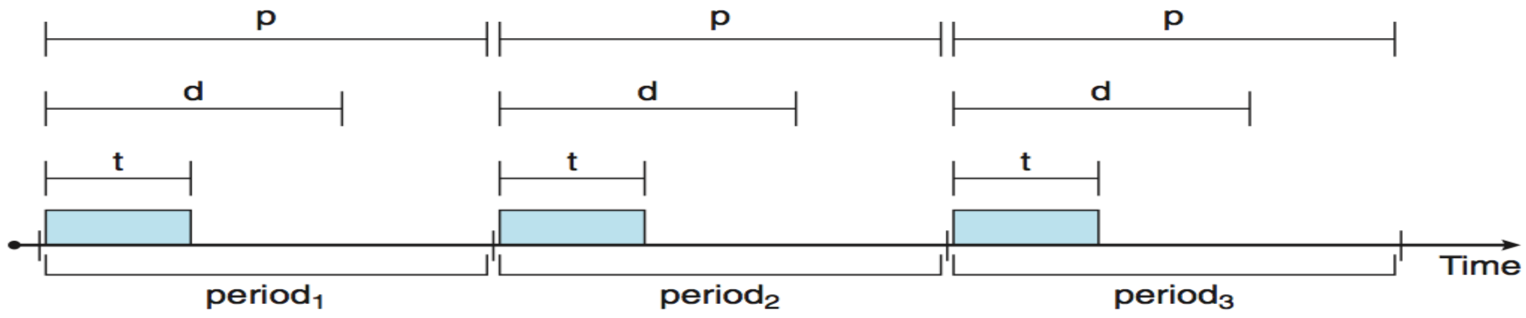
  1. Preemption of any process running in kernel mode

  2. Release by low-priority process of resources needed by high-priority processes



Priority-based Scheduling

- For real-time scheduling, scheduler must support preemptive, priority-based scheduling
  - But only guarantees soft real-time
- For hard real-time must also provide ability to meet deadlines
- Processes have new characteristics: **periodic** ones require CPU at constant intervals
  - Has processing time *t*, deadline *d*, period *p*
  - $0 \le t \le d \le p$
  - **Rate** of periodic task is $1/p$

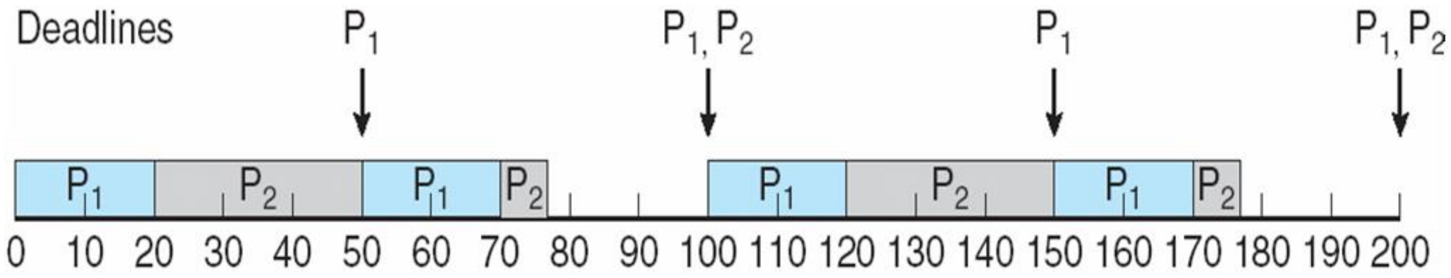


Virtualization and Scheduling

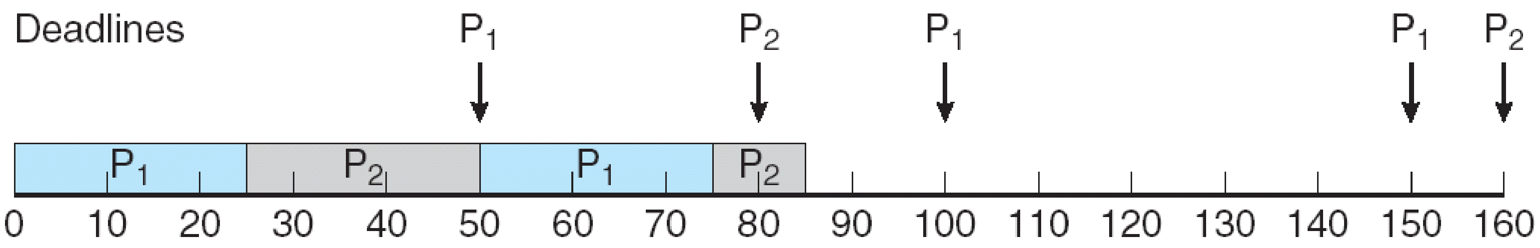
- Virtualization software schedules multiple guests onto CPU(s)
- Each guest doing its own scheduling
  - Not knowing it doesn't own the CPUs
  - Can result in poor response time
  - Can effect time-of-day clocks in guests
- Can undo good scheduling algorithm efforts of guests

Rate Monotonic Scheduling

- A priority is assigned based on the inverse of its period
- Shorter periods = higher priority;
- Longer periods = lower priority
- P_1 is assigned a higher priority than P_2.

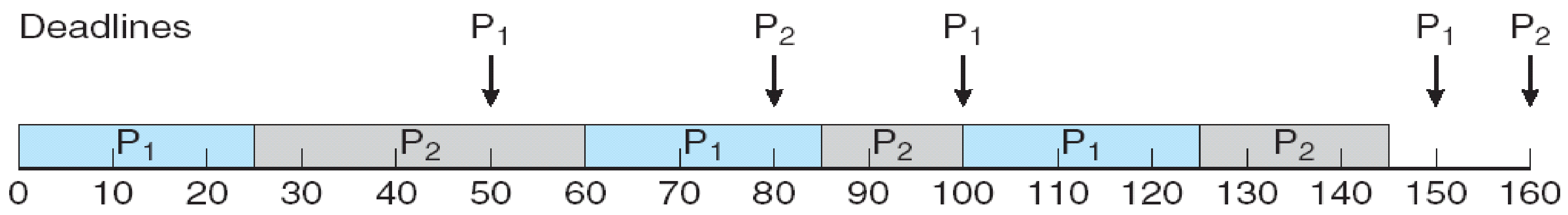


Missed Deadlines with Rate Monotonic Scheduling



Earliest Deadline First Scheduling (EDF)

- Priorities are assigned according to deadlines
  - the earlier the deadline, the higher the priority;
  - the later the deadline, the lower the priority



Proportional Share Scheduling

- T shares are allocated among all processes in the system
- An application receives N shares where $N < T$
- This ensures each application will receive $\boldsymbol{N} / T$ of the total processor time

THANK YOU

Dr Rahul Nagpal
Department of Computer Science
rahulnagpal@pes.edu