Multi-Processor Real-Time Scheduling

Luca Abeni
luca.abeni@unitn.it

December 20, 2011

Multiprocessor Scheduling

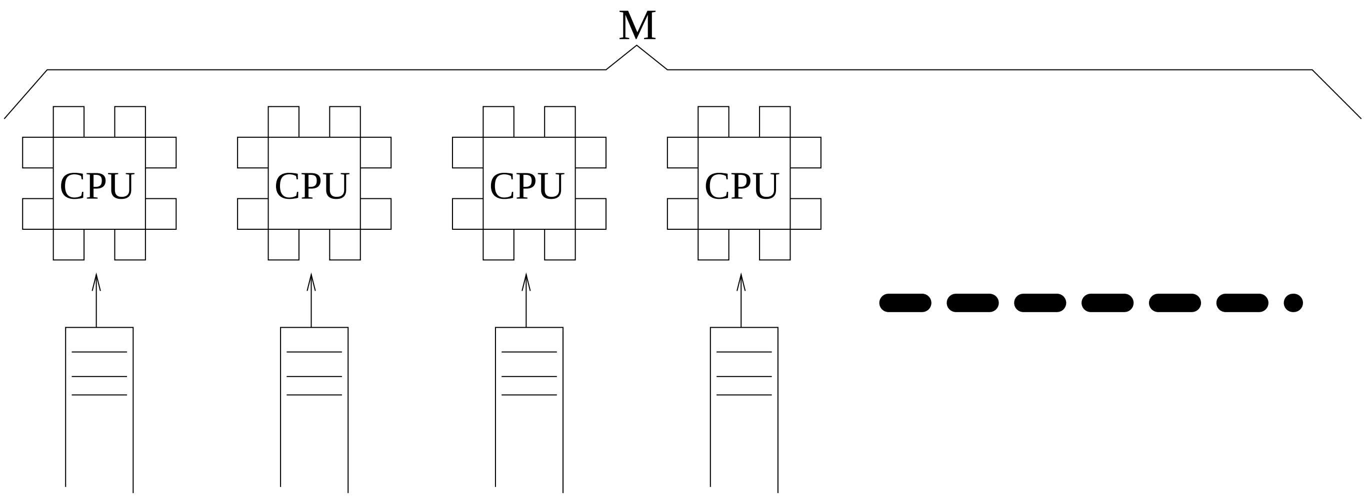
- UniProcessor Systems
  - A schedule $\sigma(t)$ is a function mapping time t into an executing task $\sigma : t \rightarrow \mathcal{T} \cup \tau_{idle}$ where $\mathcal{T}$ is the set of tasks running in the system
  - τ_{idle} is the *idle task*: when it is scheduled, the CPU becomes idle
- For a multiprocessor system with M CPUs, $\sigma(t)$ is extended to map t in vectors $\tau \in (\mathcal{T} \cup \tau_{idle})^m$
- How to implement a Real-Time scheduler for $M > 1$ processors?
  - Partitioned scheduling
  - Global scheduling

The Quest for Optimality

- UP Scheduling:
  - N periodic tasks with $D_i = T_i$: (C_i, T_i, T_i)
  - Optimal scheduler: if $\sum \frac{C_i}{T_i} \leq 1$, then the task set is schedulable
  - EDF is optimal
- Multiprocesso scheduling:
  - Goal: schedule periodic task sets with $\sum \frac{C_i}{T_i} \leq M$
  - Is this possible?
  - Optimal algorithms

Partitioned Scheduling - 1

- Reduce $\sigma : t \rightarrow (\mathcal{T} \cup \tau_{idle})^M$ to M uniprocessor schedules $\sigma_p : t \rightarrow \mathcal{T} \cup \tau_{idle}$, $0 \leq p < M$
  - Statically assign tasks to CPUs
  - Reduce the problem of scheduling on M CPUs to M instances of uniprocessor scheduling
  - Problem: system underutilisation

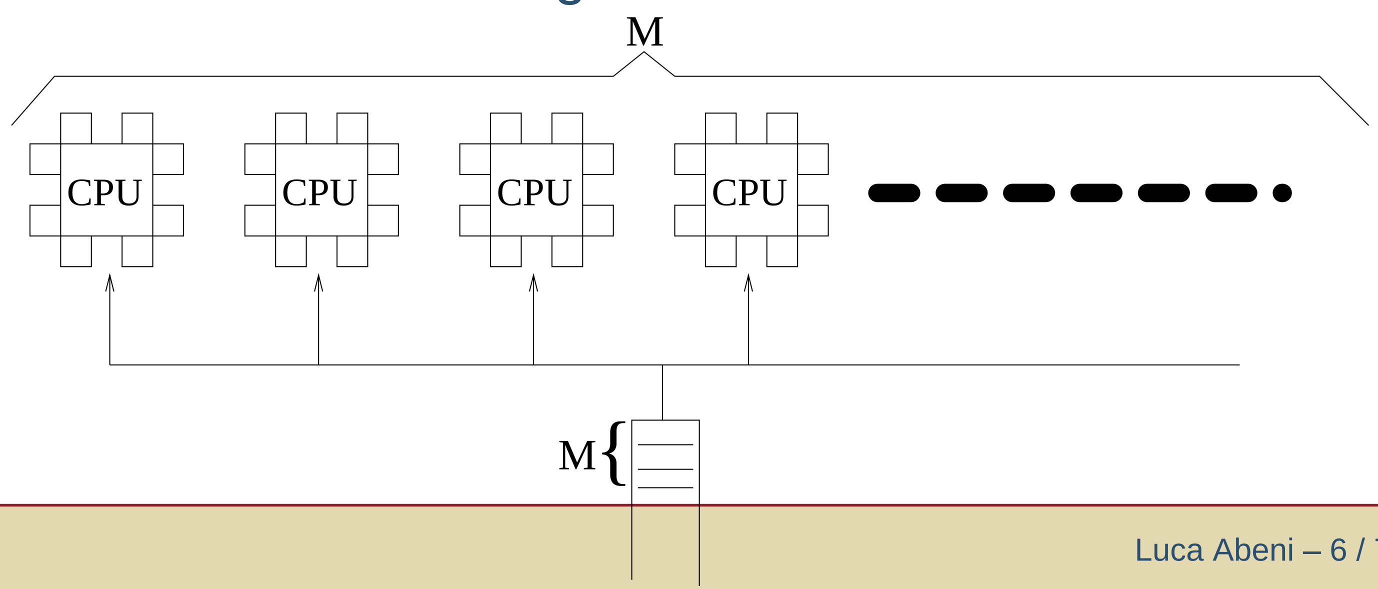


Partitioned Scheduling - 2

- Reduce an M CPUs scheduling problem to M single CPU scheduling problems and a bin-packing problem
- CPU schedulers: uni-processor, EDF can be used
- Bin-packing: assign tasks to CPUs so that every CPU has load ≤ 1
  - Is this possible?
- Think about 2 CPUs with $\{(6, 10, 10), (6, 10, 10), (6, 10, 10)\}$

Global Scheduling

- One single task queue, shared by M CPUs
  - The first M ready tasks from the queue are selected
  - What happens using fixed priorities (or EDF)?
  - Tasks are not bound to specific CPUs
  - Tasks can often migrate between different CPUs
- Problem: schedulers designed for UP do not work well



Global Scheduling - Problems

- Dhall's effect: U^{lub} for global multiprocessor scheduling can be quite low (for RM, converges to 1)
  - Pathological case: M CPUs, $M+1$ tasks. M tasks $(\epsilon, T-1, T-1)$, a task (T, T, T).
  - $U = M\frac{\epsilon}{T-1} + 1$. $\epsilon \to 0 \Rightarrow U \to 1$
- Global scheduling can cause a lot of useless migrations
  - Migrations are overhead!
  - Decrease in the throughput
  - Migrations are not accounted in admission tests...