Real-Time in the Real World

Luca Abeni
luca.abeni@unitn.it

October 24, 2011
Real-time system: \( \{ \tau_i \} \)
- \( \tau_i : (C_i, T_i) \)
- Independent tasks
- Periodic tasks, \( D_i = T_i \)
- WCET???

Theoretical schedule: function \( t \rightarrow \tau_i \)

1 CPU
Real-time system: \( \{\tau_i\}, \{S_k\} \)

\[ \tau_i : (C_i, D_i, T_i) \]

Sporadic Tasks

- Minimum Inter-Arrival Time???

Still do be solved:

- Do something about WCET and MIT knowledge
- Scheduling for more than 1 CPU (example: SMP or multicore)
- Take OS overhead (and practical issues) into account
Schedulability analysis is based on the WCET

But... How can I know it?

Today, my crystal ball is broken...

Problem: a task $\tau_i$ executing for more than $C_i$ can cause deadline misses in a different task $\tau_j$

Two possible solutions:

- Analyse the effects of variations in the WCETs: Sensitivity Analysis
- Limit the execution time in some way (enforcing a WCET): Resource Reservations
Sensitivity Analysis

- WCETs are estimations. What happens if my WCET estimation is wrong?
  - A job $J_{i,j}$ can execute for a time $c_{i,j} > C_i$!

- What’s the acceptable error in WCETs estimations?

- Formulate TDA or RTA as a sensitivity analysis problem
  - How sensible is the demanded time (or response time) to variations of the WCETs?
  - Example: What happens to $R_i$ if $C_h$ (with $p_h > p_i$) is increased by a small amount $\delta$?
    - $R_i = f(C_1, \ldots C_i, T_1, \ldots T_{i-1})$; $f()$ is not linear...

- Complex analysis, not explained here (see old slides if you are curious)
Reservation-Based Scheduling

- Force the task not to demand more time than a periodic (or sporadic!) \((Q, T)\) task

- How to enforce this?
  - Measure the demanded time, and deschedule the task when it’s too much
  - Similar to “traffic shaping used in networks”

- Temporal Protection!!!
  - If task \(\tau_i\) executes for more than \(Q_i = C_i\), it will be blocked...
  - \(\tau_i\) will miss a deadline (not other tasks!!!)
  - Similar to memory protection...
Implementing Temporal Protection

- Budget $q$, consumed when the task executes
  - When the budget is 0 the task cannot be scheduled

- Budget
  - Accounting (Enforcement)
  - Replenishment
How to cope with the MIT?

- Aperiodic tasks: no particular structure (no knowledge about the MIT)

Traditional solution: use a periodic (or sporadic) task to serve aperiodic requests...

Aperiodic Servers

- Polling Server, Deferrable Server, Sporadic Server, ...

Implementation: use a budget...

- We end up with resource reservations, again!!!
Multiprocessor Scheduling

- Real-Time scheduling with more than 1 processor?
- Trivial solution: partitioned scheduling
  - Statically assign tasks to CPUs
  - Reduce the problem of scheduling on \( M \) CPUs to \( M \) instances of uniprocessor scheduling
  - Problem: system underutilisation

- Global scheduling
  - One single ready task queue
  - Select the first \( M \) tasks from the queue
  - Problem: migrations...