University of Glasgow | School of Computing Science

Real-time Scheduling of Periodic Tasks

Advanced Operating Systems
Tutorial 1

Tutorial Outline

- Review of lectured material
- Worked examples

Review of Lectured Material

- Introduction and system model
  - Tasks, jobs, processors, resources
  - Timing constraints and scheduling algorithms
  - Periodic, aperiodic, and sporadic tasks
  - Hard and soft real-time systems
- Scheduling periodic tasks
  - Types of scheduler: clock-driven vs. priority-driven
  - Scheduling algorithms; approaches to proving correctness
  - Rate monotonic: non-optimality, time-demand analysis & critical instants, simply periodic systems, maximum utilisation tests
  - Earliest deadline first: optimality, maximum utilisation test, density test
  - Choice of rate monotonic vs. earliest deadline first
  - Other algorithms: deadline monotonic and least slack time

Worked Examples

- Identifying tasks
- Clock-driven periodic scheduling
- Rate monotonic scheduling
- Earliest deadline scheduling

Identifying Tasks

- A hypothetical helicopter flight control system, with 1/180th second cycle:
    - Validate sensor data and select data source; on failure reconfigure system
    - Do 30Hz avionics tasks, each once every 6 cycles:
        - Flight control input; data normalisation & coordinate transformation; tracking reference update
    - Do 30Hz calculations, each once every 6 cycles:
        - Control laws of outer pitch-control loop; control laws of outer roll-control loop; control laws of outer yaw- and collective-control loop
    - Every 2 cycles, do 90Hz calculations, using outputs of 30Hz calculation
        - Control laws of inner pitch-control loop; control laws of inner roll- and collective-control loop
    - Calculate inner yaw-control loop, using outputs of the 90Hz calculations
    - Output commands to control surfaces
    - Carry out built-in-test
- What are the jobs and tasks in this example?

Clock-driven Periodic Scheduling

- Example – building a cyclic schedule:
  - Consider a system of independent preemptable periodic tasks, with no precedence or resource constraints, running on a single processor: $T_1 = (6, 2)$, $T_2 = (12, 3)$, and $T_3 = (4, 1)$
  - All jobs have phase equal to zero, and relative deadline equal to their period
  - Construct a cyclic schedule for the tasks, and show that the system meets all its deadlines

Rate Monotonic Scheduling (1)

- Can the system of five independent, preemptable, tasks T_1=(1.0, 0.25), T_2=(1.25, 0.1), T_3=(1.5, 0.3), T_4=(1.75, 0.07) and T_5=(2.0, 0.1) be scheduled using the rate monotonic algorithm?

Rate Monotonic Scheduling (2)

- Can the system of three independent preemptable periodic tasks $T_1=(8, 3)$, $T_2=(9, 3)$ and $T_3=(15, 3)$ be scheduled using the rate monotonic algorithm?

Earliest Deadline Scheduling

- Can the system of three independent preemptable periodic tasks T_1=(8, 4), T_2=(12, 4) and T_3=(20, 4) be scheduled using earliest deadline first?

Discussion

- Maximum utilisation tests are powerful
    - Don’t perform exhaustive simulation or time-demand analysis unless you have to

- Exercise 1 now available – due in tutorial 2