Hard vs. Soft Real-Time Systems

Pei-Hsuan Tsai

2.1 Jobs and Processors

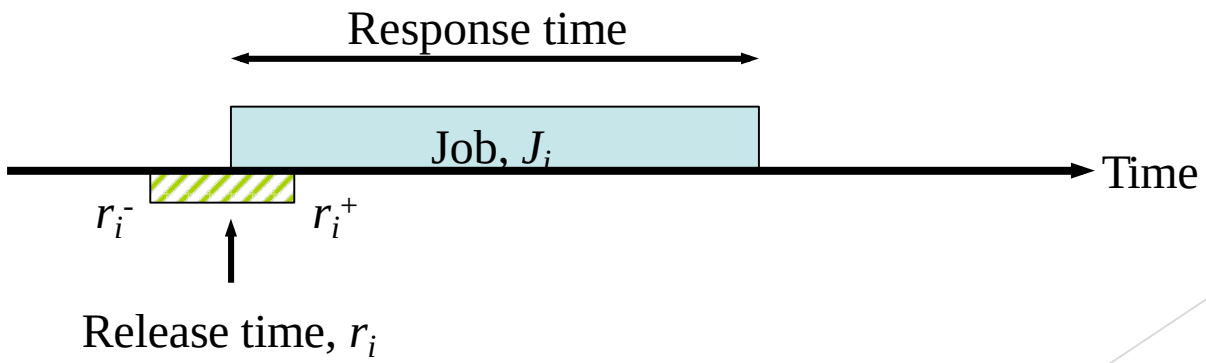
Jobs

- A job is a unit of work that is scheduled and executed by a system
  - e.g. computation of a control-law, computation of an FFT on sensor data, transmission of a data packet, retrieval of a file
- A task is a set of related jobs which jointly provide some function
  - e.g. the set of jobs that constitute the “maintain constant altitude” task, keeping an airplane flying at a constant altitude

2.2 Release times, deadlines and timing constraints

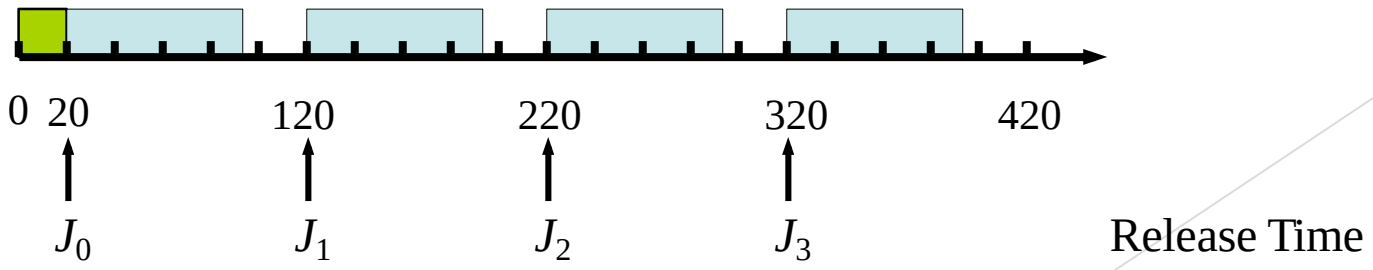
Release Times

- Release time – the instant of time at which the job becomes available for execution.
  - May not be exact: Release time jitter so r_i is in the interval $[r_i , r_i]$
  - A job can be scheduled and executed at any time at, or after, its release time, provided its resource dependency conditions are met
- Response time – the length of time from the release time of the job to the time instant when it completes
  - Not the same as execution time, since may not execute continually



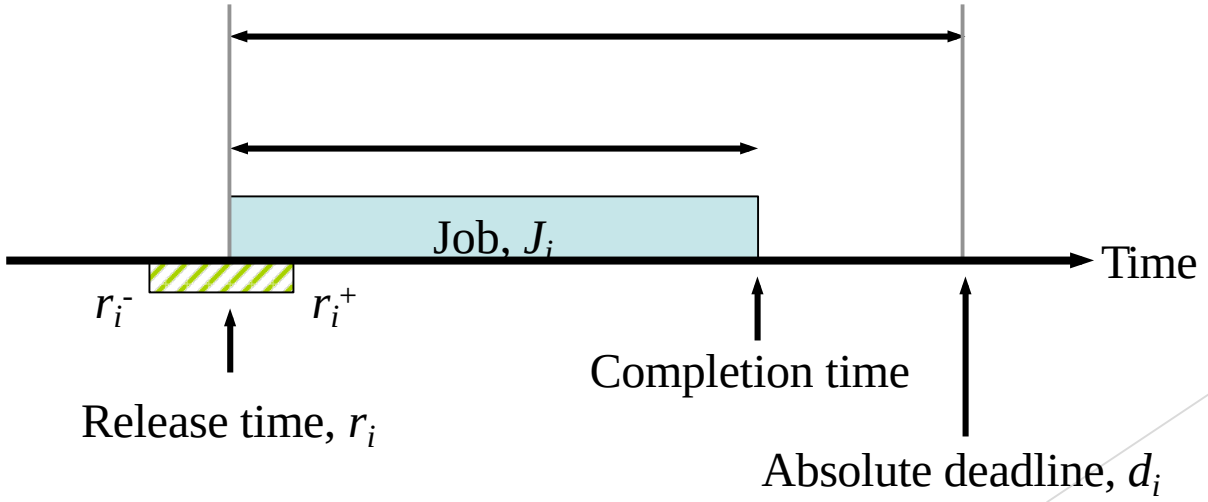
Example

- A system to monitor and control a heating furnace
- The system takes 20ms to initialize when turned on
- After initialization, every 100 ms, the system:
  - Samples and reads the temperature sensor
  - Computes the control-law for the furnace to process temperature readings, determine the correct flow rates of fuel, air and coolant
  - Adjusts flow rates to match computed values
- The periodic computations can be stated in terms of release times of the jobs computing the control-law: J_0, J_1, …, J_k, …
  - The release time of Jk is 20 + (k * 100) ms



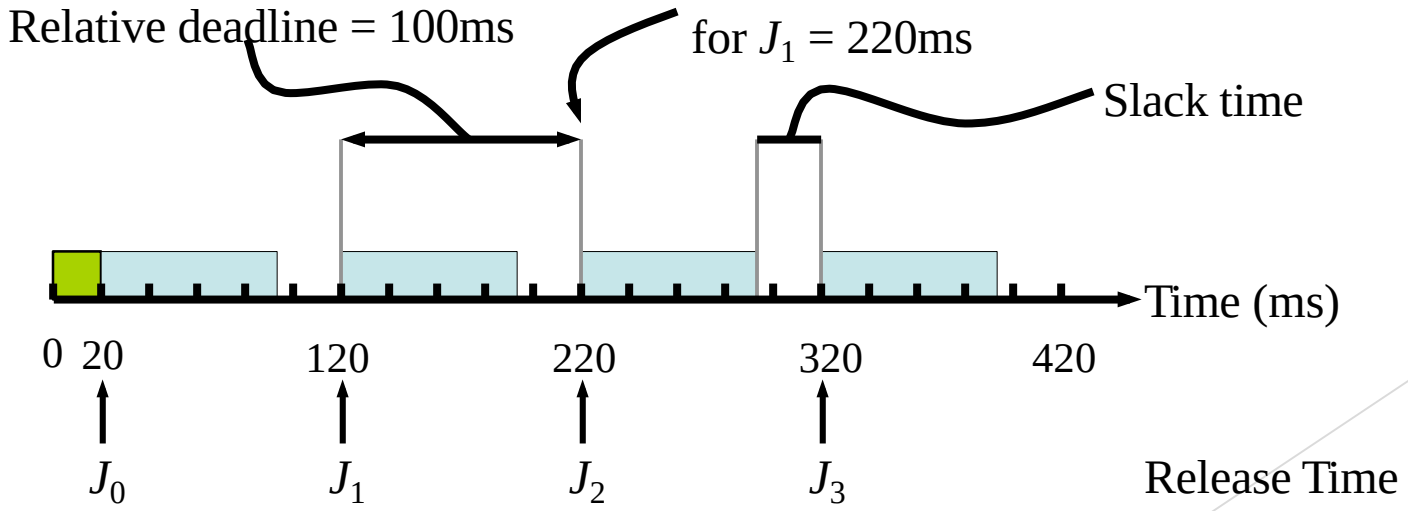
Deadlines

- Deadline – the instant of time by which its execution is required to be completed.
- Completion time – the instant at which a job completes execution
- Relative deadline – the maximum allowable job response time
- Absolute deadline – the instant of time by which a job is required to be completed (often called simply the deadline)
  - absolute deadline = release time + relative deadline
  - Feasible interval for a job Ji is the interval (ri, di]
- Deadlines are examples of timing constraints



Example

- Suppose each job must complete before the release of the next job:
  - Jk's relative deadline is 100 ms
  - Jk's absolute deadline is 20 + ((k + 1) * 100) ms
- Alternatively, each control-law computation may be required to finish sooner – i.e. the relative deadline is smaller than the time between jobs, allowing some slack time for other jobs



Timing constraints

- A timing constraint or deadline is *hard* if the failure to meet it is considered to be a fatal fault.
  - Ex: a late command to stop a train may cause a collision.
  - Ex: A bomb dropped too late may hit a civilian population instead of the intended military target.
- In contrast, the late completion of a job that has a *soft* deadline is undesirable.
  - A few misses of *soft deadlines* do not serious harm.
- The question of whether a timing constraint is hard or soft degenerates to that of how serious is serious.

Example

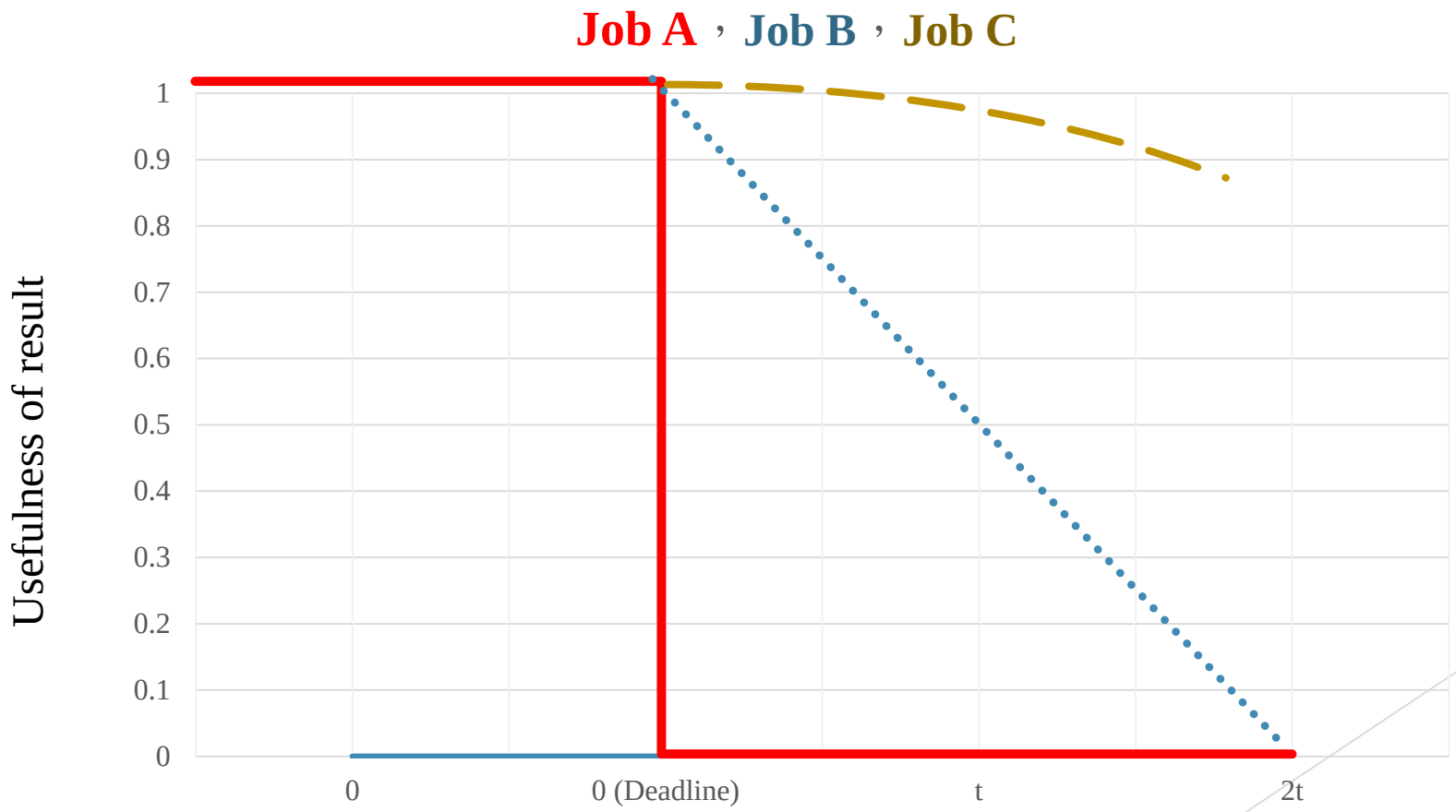
- Two deadlines for project submissions.
  - Projects can be submitted as they are announced
  - Soft deadline
    - Any submission AFTER soft deadline but BEFORE hard deadline, get 20% penalty.
    - Ex: your real score = 0.8 * your original score.
  - Hard deadline
    - Any submission AFTER hard deadline, get 0 points.
    - Ex: 0 points.
- ONLY ONE deadline for paper report!!
  - The deadline of paper report is HARD deadline.
  - Any submission AFTER hard deadline, get 0 points.

2.3 Hard and soft timing constraints

Common definitions

- Hard timing constraint: the failure to meet it is considered to be a fatal fault.
- Soft timing constraint: the late completion of a job is undesirable but a few misses of soft deadlines do not serious harm.
- The *tardiness* of a job measures how late it completes respective to its deadline.
  - Tardiness = 0: the job complete at or before its deadline.
  - Otherwise, tardiness = equal to the difference between its completion time and its deadline.
- Usefulness of a result:
  - Job with hard deadline: falls off abruptly, may even become negative when the tardiness of the job becomes larger than zero.
  - Job with soft deadline: decreases gradually as the tardiness of the tardiness of the job increases.

Usefulness of result



Hard timing constraints and temporal quality-of-service guarantees

- If the user requires the validation that the system always meet the timing constraint, the timing constraint of a job is hard.
- If no validation is required, or only a demonstration that the job meet some statistical constraint suffices, then the timing constraint of a job is soft.
- Guaranteed vs. best-effort
  - If the user wants the temporal quality (response time and jitter) of the service provided by a task guaranteed. The timing constraints are hard.
  - If the user demands the best quality of service the system can provide but the allows the system to deliver qualities below what is defined by the timing constraints. The timing constraints are soft.
- When an application creates a new task with hard timing constraints, it submit an admission request to the scheduler.

2.4 Hard real-time systems

Some reasons for requiring timing guarantees

- Deadlines of jobs in an embedded system are typically derived from the required responsiveness of sensors and actuators and controlled by it.
  - The controller compute the time for the automatically controlled train to travel the braking distance depending on the speed of the train and the safe value of deceleration.
  - Each control-law computation job of a flight controller must be completed in time.

- Jobs in non-embedded system may also have hard deadline.
  - Critical information system must never be down for more than 1 minute.
  - Reconfiguration and recovery of database servers and network connections in the system must complete within a few seconds or tens of seconds and this relative deadline is hard.

More on hard timing constraints

- No advantage in completing a job with a hard deadline early.
- Keep *jitters* in the response times of a stream of jobs small.
- Terms of specifying a hard timing constraints:
  - Deterministic constraints
    - The relative deadline of every control-law computation is 50 msec.
    - The response time of at most one out of five consecutive control-law computation exceeds 50 msec
  - Probabilistic constraints, that is, constraints defined in terms of tails of some probability distributions
    - The probability of the response time exceeding 50 milliseconds is less than 0.2
  - Constraints in terms of some usefulness function
    - The usefulness of every control law computation is 0.8 or more.

2.5 Soft real-time systems

- A system in which jobs have soft deadlines is a *soft real-time system.*
- Ex: on-line transaction systems, telephone switches and electronic games.
- The less rigorous validation required of the system and more relaxed timing constraints allow the developer to consider other performance metrics.
- Meeting all deadlines is not the only or primary consideration.
- Telephone network
  - Timing requirement of soft-real time system are often in probabilistic term.
  - The sequence must complete in no more than 10 seconds for 95 percent of the time and no more than 20 seconds for 99.95 percent of time.)
- Multimedia system
  - Provide the user with services of “guaranteed” quality.
  - Movie: 30 fps.
  - The difference in the times between video frame and its accompanied speech is no more than 80msec.
  - Quality of service guarantee is soft.
  - The validation requirement is soft.
  - The timing constraints defining the quality is soft.

Summary

- The scheduler works correctly if it never schedules any job before the release time of the job.
- A correctly scheduled job meets its timing constraint if it completes by its deadline, or it completes by its deadline with at least a certain probability, and so on.
- The timing constraint of a task can be hard or soft, depending on whether a rigorous validation of the timing constraint is required (hard) or not (soft).
- The process of validating that a system indeed meets its real-time performance objectives involves three major steps.
  - Consistency: verifies the timing constraints are specified correctly.
  - Feasibility: verifies the feasibility of each component with the underlying hardware and software resources.
  - Schedulability: verifies that the system as a whole behaves as specified by its timing constraints.

End