Real-Time Programming

Lecture 8

Farhang Nemati

Spring 2016

Repetition

- RTOS Facilities
  - Timing Facilities, Task Management, Memory Management, Error Handling, I/O Services, Interrupt Handling, Task Synchronization and Communication, Scheduling
- Task Models
- Scheduling
  - Schedulability, Feasible Schedule, Schedulability Analysis
  - Preemptive vs. Non-preemptive, Static vs. Dynamic, Offline vs. Online, Optimal vs. Heuristic, Time driven vs. Event driven

Aperiodic Task Scheduling with Synchronous Arrival Times

- A set of aperiodic tasks that arrive at the same time
- Independent (No resource sharing)
- No precedence constraints
- Tasks are specified only by execution time and deadline:

$$\tau_i(e_i, d_i)$$

Aperiodic Task Scheduling; Earliest Due Date (EDD) – Jackson's Algorithm

- Algorithm: Execute tasks according to their non-decreasing deadlines. E.g., the task with the earliest deadline is executed first.
- Since all tasks arrive at the same time, preemption is not an issue

EDD; Example1

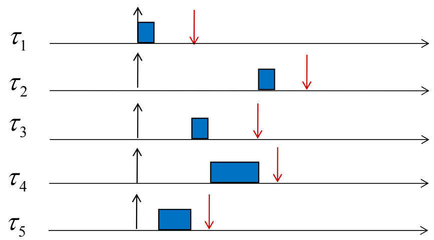
- Schedule the tasks according to EDD algorithm

	e_i	d_i
τ_1	1	3
τ_2	1	10
τ_3	1	7
τ_4	3	8
τ_5	2	5

EDD; Example

- The feasible schedule found by EDD

$\tau_1(1,3)\ \tau_5(2,5)\ \tau_3(1,7)\ \tau_4(3,8)\ \tau_2(1,10)$



EDD; Example2

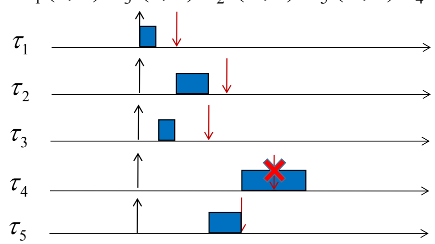
- Schedule the tasks according to EDD algorithm

	e_i	d_i
τ_1	1	2
τ_2	2	5
τ_3	1	4
τ_4	4	8
τ_5	2	6

EDD; Example

- Schedule according to EDD

$\tau_1(1,2)\ \tau_3(1,4)\ \tau_2(2,5)\ \tau_5(2,6)\ \tau_4(4,8)$



EDD

- EDD is optimal
- EDD minimizes the maximum lateness of a task set even if it cannot schedule it
  - Lateness of task τ_i denoted by L_i : $L_i = f_i - d_i$
  - Maximum lateness of a task set:

$$L_{\max} = \max(L_i)$$

EDD
Schedulability

- A task set is schedulable if:

$$\forall \tau_i = \tau_1 \ldots \tau_n \quad f_i \le d_i$$

- Suppose H_i denotes all tasks with their priority higher than or equal to τ_i

$$f_i = \sum_{\tau_k \in H_i} e_k$$

- Thus:

$$\forall \tau_i = \tau_1 \ldots \tau_n \quad \sum_{\tau_k \in H_i} e_k \le d_i$$

Aperiodic Task Scheduling with Arbitrary Arrival Times

- Preemptive Earliest Deadline First (EDF) - Horn's algorithm
- Tasks are preemptive
- Tasks can arrive at **arbitrary** time
- Tasks are specified as follows:

$$\tau_i(a_i, e_i, d_i)$$

Aperiodic Task Scheduling; Preemptive EDF

- Similar to EDD
- Independent (No resource sharing)
- No precedence constraint
- Algorithm: Any time a task arrives, sort tasks according to their non-decreasing deadlines and execute the task with the earliest deadline

Preemptive EDF

- EDF is optimal
- EDF minimizes the maximum lateness of a task set even if it cannot schedule it

$$L_{\max} = \max(L_i)$$

Preemptive EDF Schedulability

- At any time instant t:

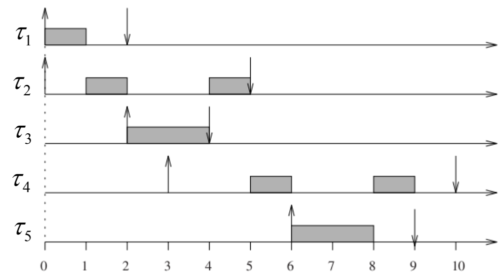
$$\forall \tau_i = \tau_1 \ldots \tau_n \quad \sum_{\tau_k \in H_i} e_k \le d_i$$

Preemptive EDF; Example

- Schedule the tasks according to Preemptive EDF algorithm

	a_i	e_i	d_i
τ_1	0	1	2
τ_2	0	2	5
τ_3	2	2	4
τ_4	3	2	10
τ_5	6	2	9

Preemptive EDF; Example



[G. Buttazzo's book]

Aperiodic Scheduling; Least Slack Time First (LST)

- Slack time of a task: $s_i = d_i - e_i$
- Algorithm: At any time a task arrives, order the tasks according to their non-decreasing slack times and execute the task with minimum slack time
- Objectives
  - There is no gain in finishing a task much earlier than its deadline. As long as it does not miss its deadline its execution can be postponed
  - Soft tasks can have more chance to execute.

Non-preemptive EDF

- Tasks are non-preemptive
- Independent (No resource sharing)
- No precedence constraint
- Algorithm: Whenever a task is finished order tasks according to their non-decreasing deadlines and execute the task with the earliest deadline until it is finished

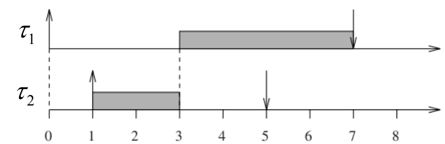
Non-preemptive EDF

- Non-preemptive EDF is **Not** optimal
- Example:

	a_i	e_i	d_i
τ_1	0	4	7
τ_2	1	2	5

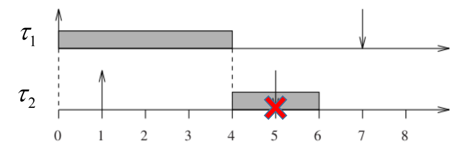
Non-preemptive EDF; Example

- A feasible schedule:



[G. Buttazzo's book]

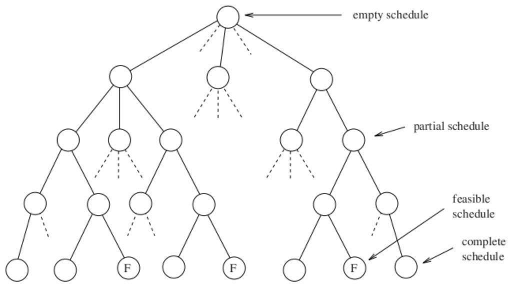
- Schedule according to Non-preemptive EDF:



[G. Buttazzo's book]

Aperiodic Non-preemptive Scheduling

- If the arrival times are known in advance for a non-preemptive scheduling, a branch-and-bound algorithm can be used:



[G. Buttazzo's book]

Aperiodic Non-preemptive Scheduling

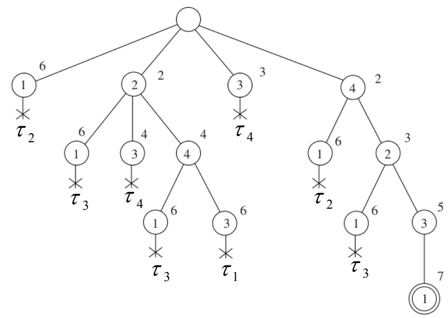
- To find a feasible schedule in the worst case an exhaustive search might be performed
  - Assuming n tasks, n.n! combinations will be checked
  - Too expensive; not practical if the number of tasks is high.

Aperiodic Non-preemptive Scheduling; Bratley's Algorithm

- Algorithm: When searching in the tree a branch skipped whenever:
  - Adding a new task (node) causes an infeasible schedule
  - A feasible schedule is found
- The algorithm is simple and easy and in average case it's effective
- The worst case can still take n.n! checks
- Not practical for online scheduling if the number of tasks are high

Bratley's Algorithm; Example

	a_i	e_i	d_i
τ_1	4	2	7
τ_2	1	1	5
τ_3	1	2	6
τ_4	0	2	4



Number in the node : the task scheduled
Number next to node: finishing time

[G. Buttazzo's book]

Aperiodic Non-preemptive Scheduling; The Spring Algorithm

- A Heuristic algorithm
  - Adding a task to a branch is guided by a heuristic function
- The objective of the algorithm is to find a feasible schedule for a task set with different constraints, e.g., non-preemptive, precedence constraints, arbitrary arrival times, resource sharing, etc.
- Algorithm: When searching in the search tree and adding a new task (node) to the tree: Add a task whose heuristic function has the minimum value among other tasks

The Spring Algorithm

- Heuristic function examples

$$H(\tau_i) = a_i$$

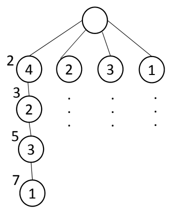
$$H(\tau_i) = d_i$$

$$H(\tau_i) = e_i$$

The Spring Algorithm; Example

	a_i	e_i	d_i
τ_1	4	2	7
τ_2	1	1	5
τ_3	1	2	6
τ_4	0	2	4

$$H(\tau_i) = d_i$$

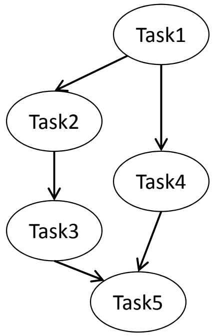


Aperiodic Scheduling; Summary of Presented Algorithm

Same Arrival Times	Different Arrival Times	
	Preemptive	Non-preemptive
•EDD	•EDF •LST	•Bratley •Spring

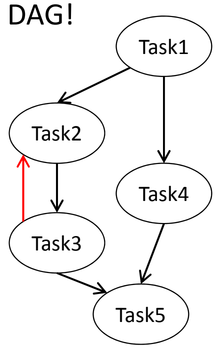
Scheduling with Precedence Constraints

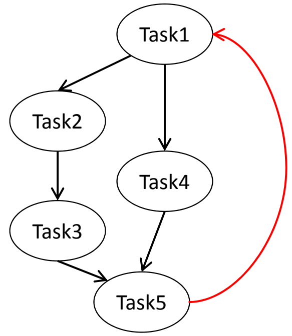
- The precedence constraints are represented with Directed Acyclic Graph (DAG)



Scheduling with Precedence Constraints

- Not a DAG!





Scheduling with Precedence Constraints

- With same arrival times
- With arbitrary arrival times

Scheduling with Precedence Constrain; Latest Deadline First (LDF)

- With same arrival times
- Algorithm: Adding a task to a schedule queue:
  - Select from tail to head
  - Tasks without successors or those whose successors are selected. Select the task with the latest deadline