Lecture 22: Dynamic scheduling

dynamic priority scheduling – at any given time during a task schedule, the scheduler can re-arrange task priorities. This approach is much more conducive to aperiodic scheduling or interrupt handling. Priority inheritance or priority ceiling protocols can be utilized. However, the potential exists for unpredictable scheduling behavior and death spiraling of missed deadlines.

Examples:
Earliest-Deadline-First scheduling (EDF) – the task with the shortest deadline at any point in time is assigned the highest priority. This is the optimal dynamic scheduling algorithm for uniprocessor systems by achieving the most efficient utilization of the CPU. Processor idling will never occur if a deadline is missed, but all n tasks are still schedulable if $Sum_{i=1,n} (e_i/p_i) <= 1$.
Note: when using multiprocessor systems, it may still be best to stick with fixed R-M scheduling on each processor separately because more CPU resources are available and hence have more predictable behavior and less task corruption.

Shortest-Completion-Time scheduling (SCT) – task priorities are set according to which task has the least execution time **remaining**, independent of all deadlines.

Least-Slack-Time scheduling (LST) – task priorities are set according to which task has the least amount of time left before it misses its deadline. Drawbacks are that the scheduling algorithm becomes unstable if a deadline is missed, and more pre-emption (hence more context switching) is possible.

Summary:
EDF highest priority task = task with $(d - t)_{minimum}$, where t is any instance in time.
SCT highest priority task = task with $(e_{total} - e_{done})_{minimum}$, where t is any instance in time.
LST highest priority task = task with $[(d - t) - (e_{total} - e_{done})]_{minimum}$, where t is any instance in time.

Exercise:	task	e	p	d	Draw Gantt charts for 2 major cycle periods of
	T1	2	4	4	T1 and T2 using EDF, SCT, and LST scheduling.
	T2	4	10	8	Hint: do minimum amount of context switching.

Exercise:	task	e	p	d	Draw Gantt charts for 1 major cycle period of
	T1	2	5	5	T1 and T2 using EDF, SCT, and LST scheduling.
	T2	4	6	6	Do any of these algorithms miss their deadlines?
					What could you notice right away about T1 and T2?

In summary about task scheduling:

- The most optimal fixed pre-emptive priority scheduling algorithm is Rate-Monotonic scheduling.
- The most optimal dynamic pre-emptive priority scheduling algorithm is Earliest-Deadline-First scheduling.
- There are two kinds of fixed scheduling techniques: Round-Robin scheduling and pre-emptive fixed scheduling.
- There are two kinds of pre-emptive scheduling: fixed and dynamic scheduling.
- Theorem (C.L. Liu, L.W. Layland, Journal of ACM, v20, p46, 1973): any set of n periodic tasks is Rate-Monotonic schedulable if total processor utilization $Sum_{i=1,n} (e_i/p_i) <= n(2^{(1/n)} – 1)$. Note that as n goes to infinity, this sum approximates $ln_e(2) \sim .69$. Also note that R-M scheduling may still be possible if this sum <= 1, but is not possible otherwise.
- Another theorem: any set of n periodic tasks is Deadline-Monotonic schedulable if the total deadline utilization $Sum_{i=1,n} (e_i/d_i) <= 1$; if this sum is > 1, D-M scheduling **may** still be possible.