A set of n periodic real-time tasks, where a task is specified by (C_i, T_i), where C_i is the computation time and T_i is the period of the task, is schedulable via EDF if $U = \sum_{i=1}^{n} U_i = C_i/T_i \leq 1$.

- Assume that all tasks require exactly C_i time units to complete, and that all tasks have the same power consumption. What is the optimal static speed that you should run the following task sets, such that the energy savings is maximal? Justify your answer
  - $\{(3, 10), (3, 10), (2, 20)\}$
  - $\{(1, 5), (3, 10), (4, 20)\}$
- If each task only needs A_i time units to complete, where $A_i < C_i$, the new task characterization can be seen as by (C_i, T_i, A_i). The extra time generated can be used by the system to further slow down the subsequent tasks in the system. Assuming a greedy slow down procedure, that is, the entire slack goes to the next task in the ready queue, give the expressions for the speed settings of the tasks in the following task set (assume tasks will be in the ready queue in the order they appear): $\{(3, 10, 2), (5, 10, 3), (4, 20, 4)\}$.

Dependable systems must include redundant capacity in either time or space (or both). Time redundancy can be exploited for re-executing jobs that failed, or to reduce power consumption if no jobs fail. Typically, checkpoints are added to reduce the recovery time.

- If a task and a checkpoint take C_i and R_i time units to execute, respectively, what is the expression (equation) that shows what is the time a task takes to complete if k checkpoints are introduced in the tasks at even intervals with no faults and with a single fault? Justify your answer.
- For the case of checkpoints, what is the expression that ensures schedulability of a single task (assume the deadline is d).
- How can one use the slack in the system to slow down tasks? Give an example and show a formula for the speed computation of task segments (segments between checkpoints).
- Give one idea and the rationale behind it for combining fault tolerance and power management.