Written Assignment #2

CPEN 432: Real-time System Design

---

1. **[10 points]** (Knowledge is Power) In a periodic task system with relative deadlines equal to the periods, rate monotonic scheduling may do as well as the earliest deadline first policy if there is a specific relationship among the task periods. If there are n tasks to be scheduled and the periods of the tasks are $P, kP, k^2P, \ldots, k^{n-1}P$ respectively (where k is an integer greater than 1) then the utilization bound is 1 and not $n(2^{1/n} - 1)$. For example, if there are four tasks and their periods are $4, 8, 16, 32$ respectively then the utilization bound is 1. In fact, when a set of tasks have periods that follow this pattern (called a *harmonic sequence*) then that set of tasks can be replaced by one virtual task for the purpose of analysis. Using this idea, develop a procedure for testing schedulability using utilization bounds giving some attention to the possibility of a harmonic chain existing. Provide a general method and then use it to test the schedulability of the following task sets:

   (a) $(e_1 = 1.5, P_1 = 6), (2, 12), (2, 18), (1, 24), (4, 48), (6, 54), (4, 96)$.

   (b) $(e_1 = 1, P_1 = 5), (1, 10), (1, 15), (2, 20), (1, 25), (3, 30), (2, 40), (5, 75), (2, 80), (5, 225)$.

   Note: The description of the algorithm and the proof of its correctness is worth **6 points**, and the calculation in each part is worth **2 points**.

2. **[10 points]** Fix a static-priority scheduling policy. Let Γ denote an implicit-deadline periodic task set, and let $U(\Gamma)$ be its utilization factor. Let

$$b_n = \inf\{U(\Gamma) : \Gamma \text{ is a critically schedulable set of } n \text{ tasks}\}.$$

   That is, b_n is the utilization bound for the scheduling algorithm on n tasks (e.g., $b_n = n(2^{1/n} - 1)$ in the LL bound for RM). Let U_n denote the utilization factor of a set of n tasks. *Show that if b_n is a decreasing sequence, then for any set of n tasks with utilization factor U_n, if $U_n \leqslant b_n$, then the taskset is schedulable under the fixed-priority scheme.*

   We took this for granted when we derived the utilization bound for RM in class, but this—the sufficiency of the bound—is not so obvious and needs proof.

3. **[30 points + 10 Bonus]** (Empirical Evaluation of Scheduling Algorithms – Involves coding).

   (a) **[20 points]** To test the efficacy of different scheduling policies, you should compare the policies over a range of task sets. In particular, knowing that the ability to meet deadlines is (loosely) connected to the utilization of a task set, one might want to know whether a particular policy is suitable at a certain utilization level. To be able to compare across scheduling policies, you should use the same task sets for all policies and you should cover a wide range of possible task sets. It is sufficient to consider implicit-deadline tasks for this part of the assignment. It is possible to generate task sets "at random" but we really want to sample the space of task sets carefully. Given that scheduling policies are sensitive to the utilization of tasks, their periods and execution times, one can generate a task set with n tasks as follows:

i. Fix the task set utilization to be U ($1 \leqslant U \leqslant 1$);
ii. Generate a vector $\langle u_1, u_1, \ldots, u_n \rangle$ such that $\sum_{i=1}^{n} u_i = U$ and $0 \leqslant u_i \leqslant U$;
iii. Generate task periods $P_1, P_2, \ldots, P_n$;
iv. Then, compute $C_i = u_i P_i$.

For generating task periods, assume that task periods are random variables that are log-uniformly distributed in $[10, 1000000]$. In other words, if a task period is P then $\log P$ is uniformly distributed in $[1, 6]$. (There are particular reasons for this choice of distribution that are not detailed in this assignment specification.)

The central question then is the generation of the utilization vector. There are infinitely many utilization vectors that achieve utilization U. If v_1 and v_2 are two vectors that achieve utilization U, then a good sampling strategy would ensure that these two vectors are independent and are generated with equal probability.

A brute force scheme would generate a good sample of task sets but might be exceedingly time consuming. On the other hand, a simple strategy might result in utilization vectors that are concentrated in specific regions in the utilization space; e.g., a simple averaging strategy will result in the utilization vectors not evenly distributed but rather concentrated at the center of the utilization space (see Figure 1 below for bad and good utilization vector distributions).

For this part of the assignment, you should develop an **efficient** algorithm for generating task sets that performs **uniform** sampling over the space of all utilization vectors that achieve a utilization U. Once the utilization vector has been generated, the task set can be obtained as described above.

(b) **[10 points + 10 bonus]** In this problem we will consider three static-task priority schemes for scheduling recurrent real-time task systems on one processor: **Rate Monotonic (RM)**, **Shortest Processing Time First (SPTF)**, and **Maximum Utilization First (MUF)**. Both SPTF and MUF are fixed-priority if the priorities are assigned using the worst-case execution time (WCET) of a job.

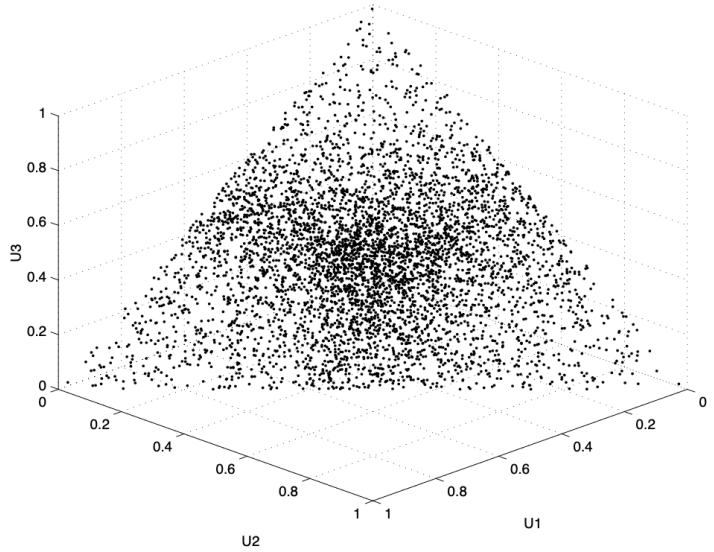
Using the schedulability analysis and task set generation tools you developed in the previous part, you must compare RM, SJF, and MUF. To compare their performance, we will use two metrics.

i. **[10 points]** The first metric to use is the fraction of feasible task sets. In other words, for a fixed utilization level U, generate $100,000$ task sets with n tasks and determine what percentage of these task sets were schedulable using the three policies. This is empirical performance analysis and you should start with $U = 0.05$ and increase U in steps of 0.05 until $U = 1$. Use response time analysis wherever it is necessary to do so. For this part of the assignment, you should plot a graph with U on the X-axis and the fraction of schedulable task sets on the Y-axis. Perform this assessment for four task set sizes: $n = 8, 16, 32, 64$ (you should plot four graphs).

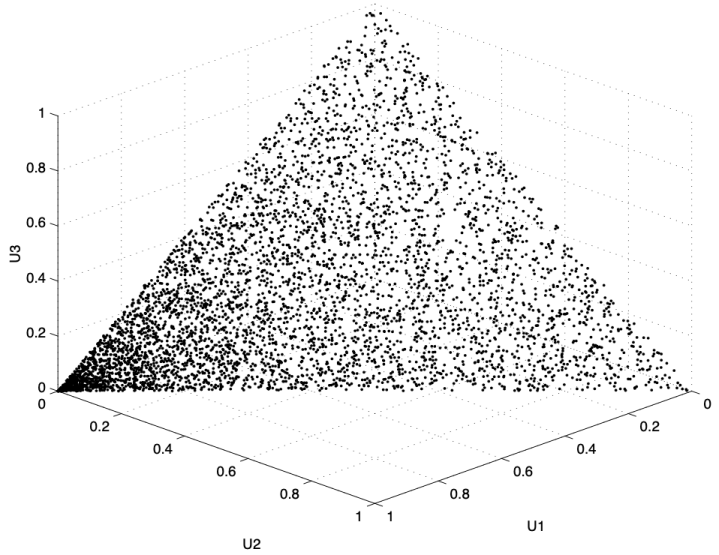
ii. **[Bonus: +10 points]** The second metric to use is the number of successful job completions. This is much harder to measure because we cannot use efficient analysis techniques. Therefore, generate 100 task sets of 16 tasks at each utilization level. **Simulate** the execution of these task sets according to each of the three policies under investigation for $100,000$ time units. For each simulation run, measure the percentage of successful job completions (i.e., jobs that meet their deadlines). Plot the mean percentage of successful job completions and indicate the standard deviation via error bars (or any other suitable method) at each utilization level. For this study, vary the total utilization of the task sets

from $U = 0.1$ to $U = 0.9$ in steps of 0.1 and generate task periods in the range $[10, 100000]$ using a log-uniform distribution.

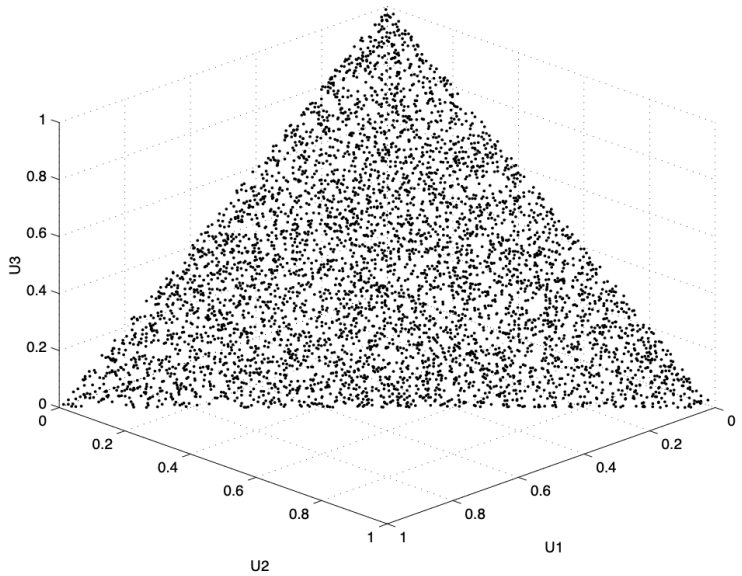
4. **[10 points]** (Theoretical analysis of SPTF and MUF). RM scheduling has two fast sufficient tests for schedulability analysis of implicit-deadline periodic task sets (the Liu and Layland bound and the Hyperbolic bound). Using some of the ideas that we explored in the proof of the Liu and Layland bound, derive tight utilization bounds for SPTF and MUF scheduling for recurrent implicit-deadline task systems. Recall that a utilization bound U_b is tight if all task sets with utilization $U \leqslant U_b$ are schedulable and for every $U > U_b$, there exists a task set with utilization U that is *not* schedulable.



(a) utilization points biased to the center of the U-space



(b) utilization points biased to a corner of the U-space



(c) utilization points uniformly distributed (desired)

Figure 1: Example utilization vector distributions in the regular 2-simplex (3 tasks)