Clint Moulds
cmould1@gl.umbc.edu

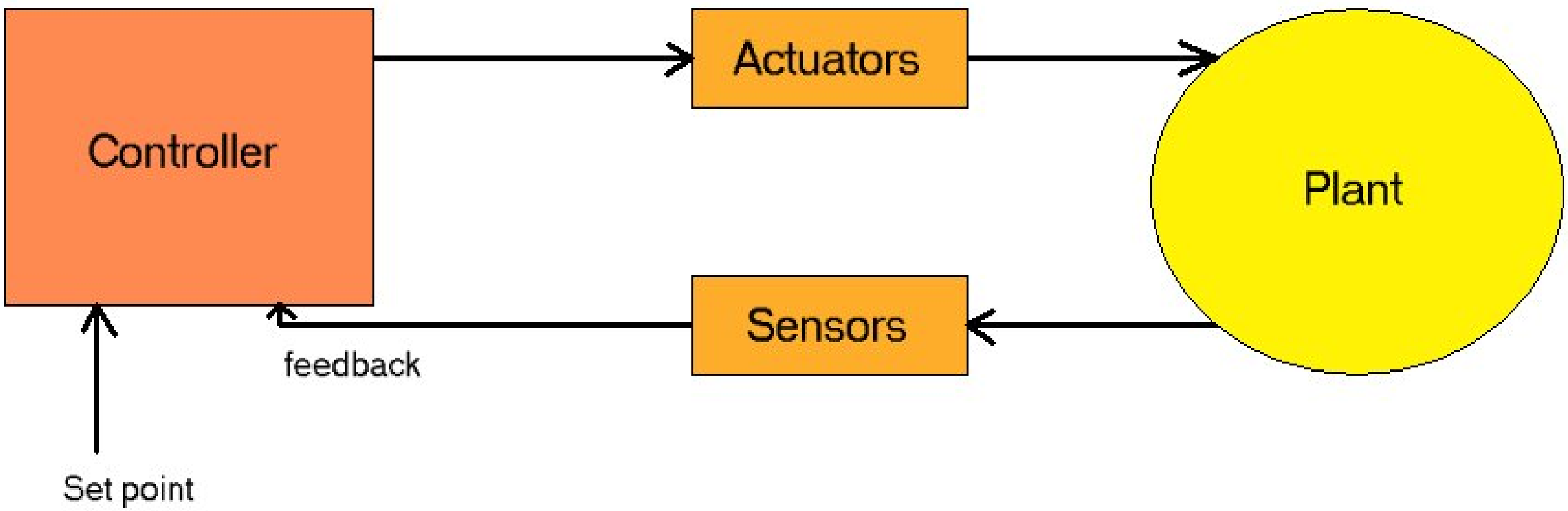
"Design and Evaluation of a Feedback Control EDF Scheduling Algorithm"

by Lu, Stankovic, Tao, and Son

A summary presentation

Table of Section Headings

What is Feedback Control?



- Example: Thermostat (controller), set point is desired temperature, actuator is the H/C system in your house, plant is the air in your house, sensor is thermometer that takes readings and gives feedback to the controller.

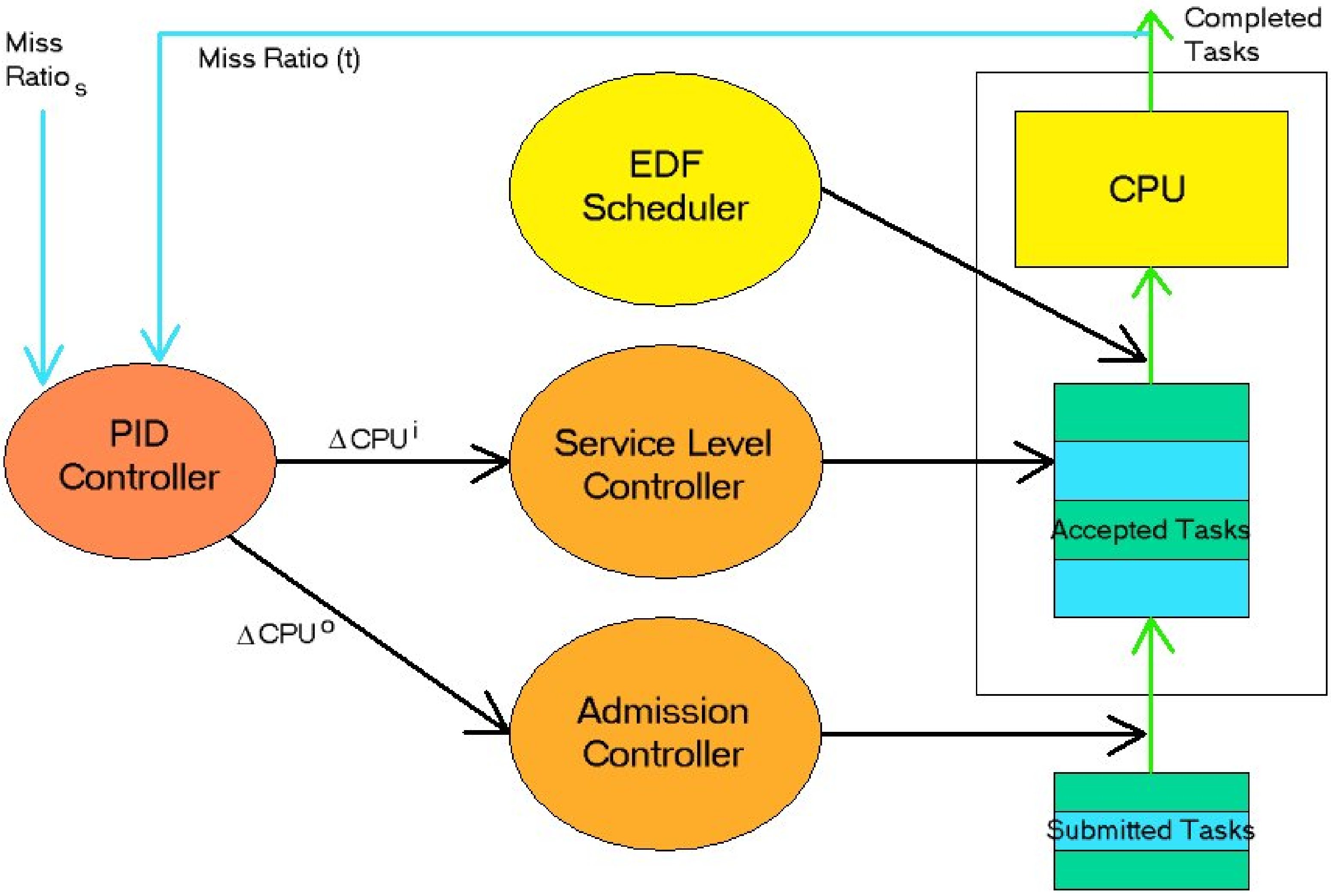
Feedback Control Terminology

- Controlled variable
  - What the system measures and tries to control.
- Set point
  - The desired (correct) value of the controlled variable.
- Error (function)
  - f (time, (value of the controlled variable) – (Set point)).
- Manipulated variable
  - What the system acts on to affect the controlled variable.
- Control function
  - Computes how much to act on the system based on the error.
- Actuator mechanisms
  - The method used to apply the control function output to the manipulated variable.

Feedback Control Scheduling

- Controlled variable
  - Deadline miss ratio : The percentage of admitted tasks that miss their deadlines.
- Set point
  - A value of 1% is deemed acceptable for a soft real-time scheduler.
- Error (function)
  - The difference between the actual miss ratio and the set point of 1%.
- Manipulated variable
  - Requested CPU Utilization : Since it has a clear and direct impact on miss ratio.
- Control function
  - Proportional-Integral-Derivative (PID) control formula was chosen.
- Actuator mechanisms
  - Admission controller (AC) and Service Level controller (SLC).

FC-EDF Architecture



Model Specifics : Tasks

- Task := { I, ET, VAL, S, D }

- [I]
  Set of logical versions (Instances) available for this task. For example one instance may be more precise but require more execution time.

- [ET]
  Set of Execution Times, in the form of requested CPU utilization, one for each instance. A percentage value (0 < ET <= 1).

- [VAL]
  Set of relative Values reflecting the value (worth) of the output from each instance. Assumption: Longer time version yields higher value, (high Service Level).

- [S]
  Set of relative Start Times, one for each instance.

- [D]
  Set of soft Deadlines, one for each instance. Could be expressed as a range to give the SLC finer control.

Model Specifics : PID Controller

$$\Delta CPU(t) = C_P error(t) + C_I \sum_{IW} error(t) + C_D \frac{error(t) - error(t - DW)}{DW}$$

- error(t)
  $MissRatio_S$ – MissRatio(t)

- [SP]
  Sampling Period, how often this controller should be called (applied). SP is the time unit.

- [IW]
  Integral Window, include the error terms from each of the previous IW time units in the calculation of the integral term.

- [DW]
  Derivative Window, include the change in the error from DW time units ago (to now) in the derivative term.

- [C_P, C_I, C_D]
  Coefficients used to tune the PID controller.

Model Specifics : Actuators

- Service Level Controller (SLC)
  - The fine-grained control mechanism of the system. By adjusting the service levels of the admitted tasks the SLC can adjust the utilization. For example changing a task T_i from using instance j to using instance k will accommodate the following change in utilization (may be + or -):

$$\Delta CPU_{\text{accommodated}} = ET_{ij} - ET_{ik}$$

- Admission Controller (AC)
  - The "broad brush" control mechanism of the system. By adjusting when and how many tasks are admitted the AC adjusts the utilization. Applied after the SLC to change the utilization by ΔCPU – ΔCPU_{SLC}.
  - When negative the AC will stop admitting tasks until the utilization has decreased by that amount. When positive it will continue admitting tasks at the highest service level k for task T_i such that:

$$CPU\,(t) + ET_{ik} < 1$$

Applying Control Theory

- The only element of the model left to define is the value of the coefficients. These must be chosen carefully so that the system satisfies the BIBO stability constraint. That is, for a bounded input the system output is also bounded.

- Stability is an important condition for any real-time system to avoid starvation conditions and/or cascading failures.

- Control theory defines conditions for the coefficients that guarantee stability (even for an imprecise model in the case of PID control). These hold as long as the product of the gain functions is close to 1.

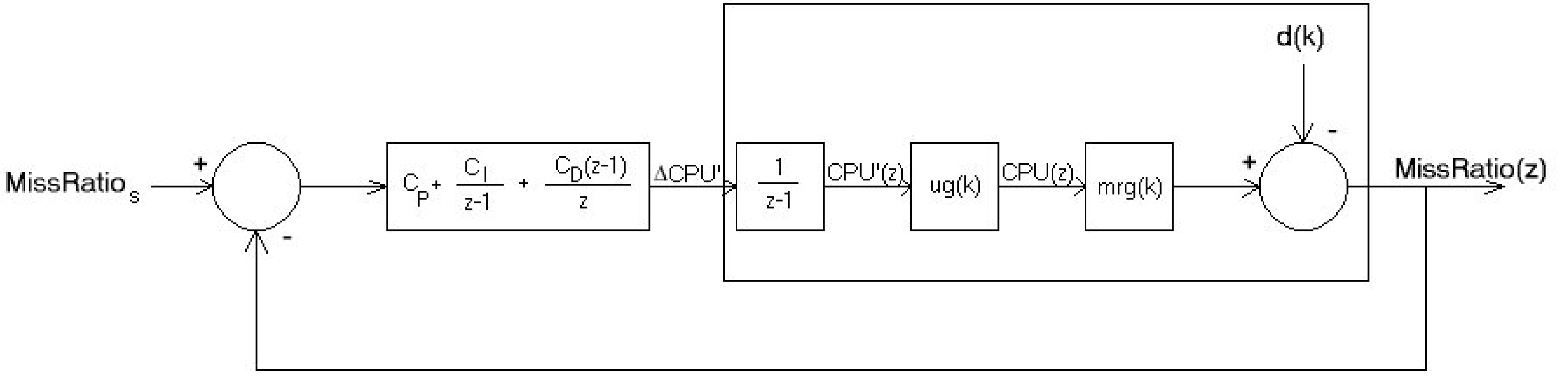
- The gain functions in our example are [1.] ug(k), the mapping of estimated CPU utilization to actual CPU utilization, and [2.] mrg(k), the mapping of CPU utilization to deadline miss ratio. In our imprecise model these functions are purely conceptual, but are important to understanding the system.

- Either rule may be used to define values for the coefficients:

1. $C_I > 0$, $|C_D| < 1$, $2C_P – C_I + 4C_D < 4$, and $2 – 2C_D2 > C_DC_P + C_P - C_I > 0$

2. $C_I = 0$, $|C_D| < 1$, and $0 < C_P + 2C_D < 2$

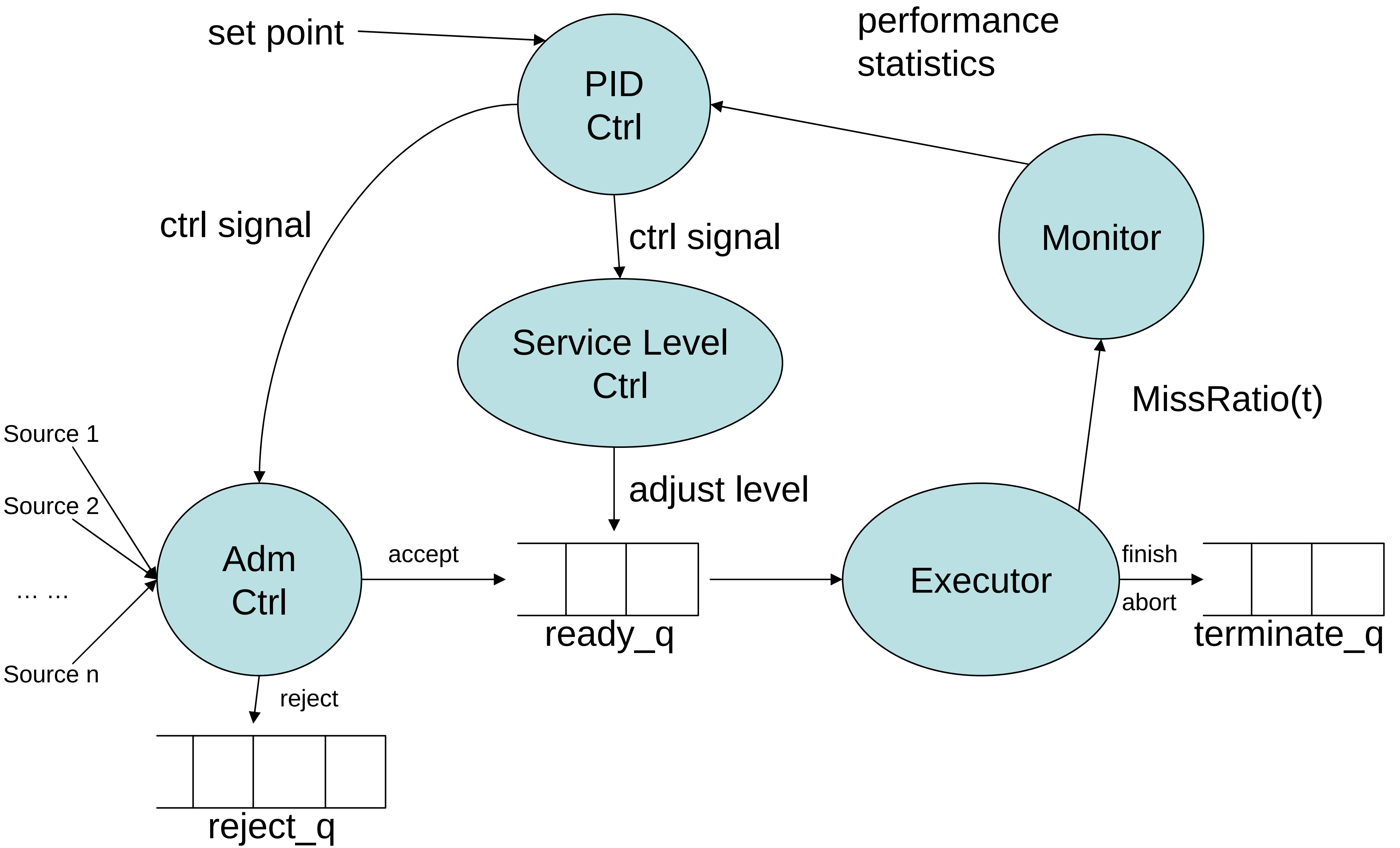
FC-EDF Block Diagram



- This represents the feedback control system as a whole. The elements inside the large box represent the “plant” (the real-time operating system) and are essentially conceptual only.

- ΔCPU’ : The change in estimated CPU utilization (system input).
- CPU’(z) : The estimated CPU utilization.
- CPU(z) : The CPU utilization.
- ug(k) : Utilization gain function. Maps estimate to actual.
- mrg(k) : Miss Ratio gain function. Maps utilization to miss ratio.
- d(k) : The disturbance. Reflects the difference between calculated (approximated) effect on the output and the actual effect. If this were known it could be added to the calculation to arrive at the exact (actual) miss ratio of the system.

Simulation Model



Simulation Inputs

- Workload
  - Task := {P, WCET, BCET, EET, AET, VAL}
  - [P] : Task period, also its deadline. Same for both service levels.
  - [WCET] : 2 Worst-case ETs. $WCET_0 = 2*WCET_1$
  - [BCET] : 2 Best-case ETs. $WCET_i = 4*BCET_i$
  - [EET] : 2 Estimated ETs. $EET_i = (WCET_i + BCET_i) * 0.5$
  - [AET] : 2 Average ETs. $AET_i = EET_i * etf$. (etf = <u>execution time factor</u>)
  - [VAL] : 2 Values, one for each service level. $VAL_0 = 1.0$, $VAL_1 = 0.5$
- Values
  - Set Point = 1% (0.01), SP = 2400 time units, IW = 100 (SP), DW = 1 (SP), $C_P = 0.5$, $C_I = 0.05$, $C_D = 0.1$

Simulation Metrics

- Compared 4 variant EDF algorithms:
  - EDF : Basic algorithm, no admission controller, service level controller, or PID controller.
  - EDF + AC : Static admission control. No service level controller or PID controller.
  - EDF + P : Proportional control. $C_P = C_D = 0$.
  - FC-EDF : Feedback control EDF, the full model.
- Used 4 metrics to compare performance:
  - MRA : Miss Ratio among Admitted tasks. Primary function.
  - UTIL : CPU Utilization. Cost consideration.
  - HRS : Hit Ratio among Submitted tasks. Measure of throughput.
  - VCR : Value Completion Ratio. Measure of quality.

Simulation Results 1: Steady Execution Time

- Set-up
  - Describes situation where the average load doesn’t change during the experiment. The estimation is the only variant. Each run chooses a fixed value for etf, optimistic or pessimistic, in the range of [0.4, 1.6].
  - Note that in hard real-time systems etf << 1.0.
- Results
  - FC-EDF and EDF+P dominate in MRA when estimate is optimistic, since the others do not incorporate any feedback.
  - EDF is greedy and so has the highest CPU utilization, but FC-EDF maintains a high value here as well.
  - For throughput (HRS) and quality (VCR), FC-EDF also comes out on top most consistently, as the PID control adjusts the pessimistic estimate upwards and the optimistic estimate downwards.

Simulation Results 2: Dynamic Execution Time

- Set-up
  - Describes situation where the average load changes dramatically over time. The estimation is adjusted periodically to represent changes in workload. Each run chooses 4 sequential values for etf in the range of [0.4, 1.6].
  - The 4 intervals will each have their own value of etf, (e.g. 0.8, 1.3, 0.8, 1.2). The response time at the change points is the primary item of interest here.
- Results
  - FC-EDF yields the fastest response time, indeed the only response time since EDF+P was dropped from the experiment and the others don't use feedback.
  - EDF has 100% CPU utilization as before but at the cost of missing over half of the scheduled tasks. EDF + AC can handle miss ratio when etf is pessimistic or CPU utilization when etf is optimistic, but not both at once.
  - FC-EDF clearly demonstrates the ability to adapt to changes in workload during a run, despite a small adjustment "spike" where the miss ratio climbs sharply for a brief period or the utilization drops sharply for a brief period.

Conclusion

- Overhead
  - Very minimal overhead. The simulation was running with SP = 2400 which meant more than 100 task instances were executed between each PID control calculation.
  - The calculation itself is also very minimal, and the actuators can operate in O(n) where n is the number of admitted tasks.
- Overall Performance
  - FC-EDF performs very well for little cost and should be considered for any soft real-time system as it can (1) provide soft guarantees for miss-ratio, (2) maintain high CPU utilization, and (3) maintain high throughput.