CS 333: Operating Systems Lab
Autumn 2017

Lab #9 Thoughts in Dining

Goal

In this lab you will solve another important synchronization problem.

The Dining Philosophers Problem!

The problem: N philosophers are sitting around a table with N forks between them. Each philosopher must pick up both forks on his/her left and right before he/she can start eating. The goal is to come up with an algorithm that lets all philosophers eat, without deadlock or starvation.

0. Getting Started

Implement a program `dining0.c` which takes in the number of philosophers as a command-line argument.

Usage : `./dining0 N`

1. Each philosopher spawns a new thread using `pthreads` which executes the following logic:
   - Waits till he/she can pick up both his/her forks
   - Prints `Philosopher <i> eating` when he/she is able to pick both forks
   - Then sleeps for some random time between 1-2 seconds before putting down both forks
   - Prints `Philosopher <i> finished`.
2. A variable state is associated with each philosopher, and can be one of EATING (holding both forks) or THINKING (when not eating).
3. Maintain a conditional variable for each fork. So for eating, a philosopher has to sleep on both the left and right fork's conditional variables.

NOTE: This is not a correct solution and might lead to a deadlock. Write the reason as to why this approach is flawed, clearly and briefly, in `report.pdf`

1. The Fix

Design and implement a deadlock free solution, by making a philosopher pick up forks and eat only when both forks are available.

Implement a program `dining1.c` which takes in the number of philosophers as a command-line argument and which executes without deadlocks

Usage : `./dining1 N`

Submission Guidelines

- All submissions via moodle. Name your submission as: **<rollno_lab9>.tar.gz**
- The tar should contain the following files in the following directory structure: `<rollno_lab9>/`

  ```
  |__task0/
  |____dining0.c
  |__task1/
  |____dining1.c
  |__report.pdf
  |__outputs/
  |____<Outputs with various N on dining0.c and dining1.c>
  ```

- **Deadline: Friday, 20th October 2017 - 05:15 PM**.