Programming Assignment: Dining UPC Philosophers

CS 377: Parallel Programming
Fall 2020

October 2, 2020

1 Administrative Details

Due: Mon, Oct. 12, 2020

To be handed in: via the `submit377` script (as `assign3`), your write-up, and your two (or three!) versions of your Dining UPC Philosophers solutions.

Write-up: Your write-up will answer the questions posed below (look for the question marks...), in addition to the standard elements required in all write-ups: your overall experience solving problems, problems encountered, how solved, lessons learned, etc.

Starter Code: You should have already copied the starter code when you copied my `/home/mlsmith/cs377-examples/upc` directory. The file containing the starter code is *creatively* named: `philosophers.upc`

2 Description

The given implementation of a solution to the dining philosophers problem "suffers" from several problems. First, the shared resources are forks, instead of chopsticks! But in the interest of shorter variable names, we'll let this go. Second, more seriously, it *cheats*: it doesn't block while waiting to acquire the locks (forks)! Instead, it uses the UPC function: `upc_lock_attempt()`, which introduces the tradeoff between deadlock and livelock. Third, even though this solution doesn't suffer from deadlock, it does suffer from something else. What?

3 Assignment

Your mission is to replace the `upc_lock_attempt()` statements with `upc_lock()` statements. Unfortunately, this will introduce the possibility of deadlock into the program. Why? (Note: if your program deadlocks, you can break out of it by pressing the CTRL+C keys.)

We discussed several strategies for avoiding the possibility of deadlock in the Dining Philosophers problem:

1. One of the philosophers picks up her chopsticks in a different order from the other four.
2. Even numbered philosophers pick up chopsticks left-right; odd numbered philosophers pick up chopsticks right-left.
3. A ”waiter” seats at most four philosophers at a time at the table. (Optional)

UPC supports implementing two of these strategies directly by using the MYTHREAD and THREADS values. Implement the first two solutions. Remember, you must use `upc_lock()`, and not `upc_lock_attempt()`.

The third strategy, which uses a waiter to seat the philosophers, will use these two values, as well, but is a little trickier to pull off. It may be impossible in UPC without busy waiting. (*Hint: you may want to consider adding two additional states:* `STANDING` and `SITTING`.