Recitation 6: Deadlocks and TCP

Department of Computer Science and Engineering
University of Minnesota

July 13, 2025

Overview

- Deadlocks.
- Brief intro on TCP/IP.
- Exercise: Dining Philosophers.

Deadlocks

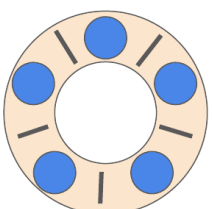
- All threads are blocked and cannot make progress
- **Resource**
  - A hardware device or piece of information required by a thread to proceed
  - Sometimes exclusive access is required
  - Using a resource: **Request** - **Use** - **Release**
- **Four conditions for a deadlock**
  - Mutual exclusion
  - Hold-and-wait
  - No-preemption
  - Circular wait

Strategies to Deal with Deadlocks

- Ignore and continue to next task (Ostrich algorithm)
- Detect deadlock and take action to recover
- Careful allocation of resources
- Negate one of the four conditions for deadlock to occur

Dining Philosophers Problem

- **1965** - Dijkstra posed the Dining Philosophers Problem
- Given 5 philosophers who are seated around a table. Each philosopher has a plate of spaghetti. Two forks are required to eat the spaghetti. Between each plate there is a single fork. A philosopher alternates between eating and thinking. When a philosopher is hungry, they try to acquire left and right forks one at a time. If successful, they eat and then put down the forks and continue to think.
- **What is the issue here?**
  - 5 philosophers, 5 forks
  - How to acquire both the forks without causing a deadlock?



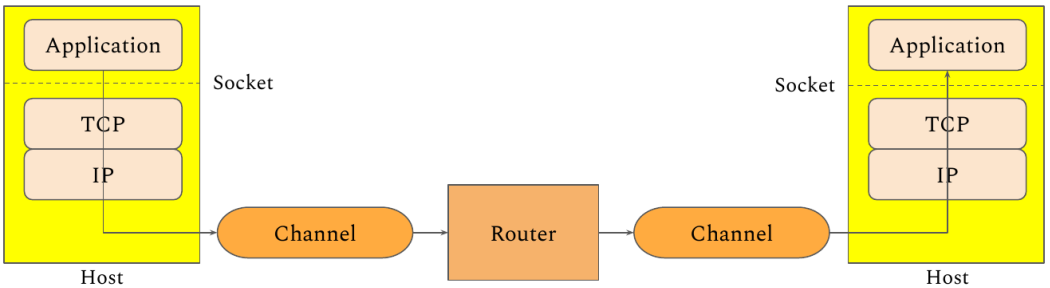
Solution

- **Acquire left fork**
  - Successful → Acquire right fork
    - Successful → Eat and release left followed by right fork
    - Unsuccessful → Release left fork
  - Unsuccessful → Retry

```
# philosopher(i):
#   while(true)
#     think()              // philosopher thinking
#     acquireForks(i)      // acquire left fork and right fork
#     eat()                // eat spaghetti
#     releaseForks(i)      // release left fork and right fork
```

TCP

- **Protocol suite for data transfer in network**
  - TCP - Transmission Control Protocol
  - IP - Internet Protocol
  - UDP - User Datagram Protocol
- **TCP/IP**



TCP

- **Reliable byte-stream channel**
  - Detect and recover from the losses, duplications and other errors experience by IP
- **Connection-oriented protocol**
  - Programs must establish connection using handshake mechanism

Exercises and Deliverables: Dining Philosophers

- You are given the code for dining philosophers (**philo.c**). Complete the code and ensure there is no deadlock.
- Each philosopher has 3 states: **HUNGRY/EATING/THINKING**
- Use mutex locks and condition variables to solve the problem. (**NO SEMAPHORES**)
- Check the philo target. We are using redirection (>) to store the output of the program. Ensure to call **clean** target before re-running the code.
- The expected is given: **out_expected.txt**. The order of output line may vary.
- Submit the exercise zip (philo.c, out.txt, Makefile) to Gradescope by **July 18th 11:59 pm**.
- There are no sample codes this time as we are re-visiting similar ideas as Lab 5.