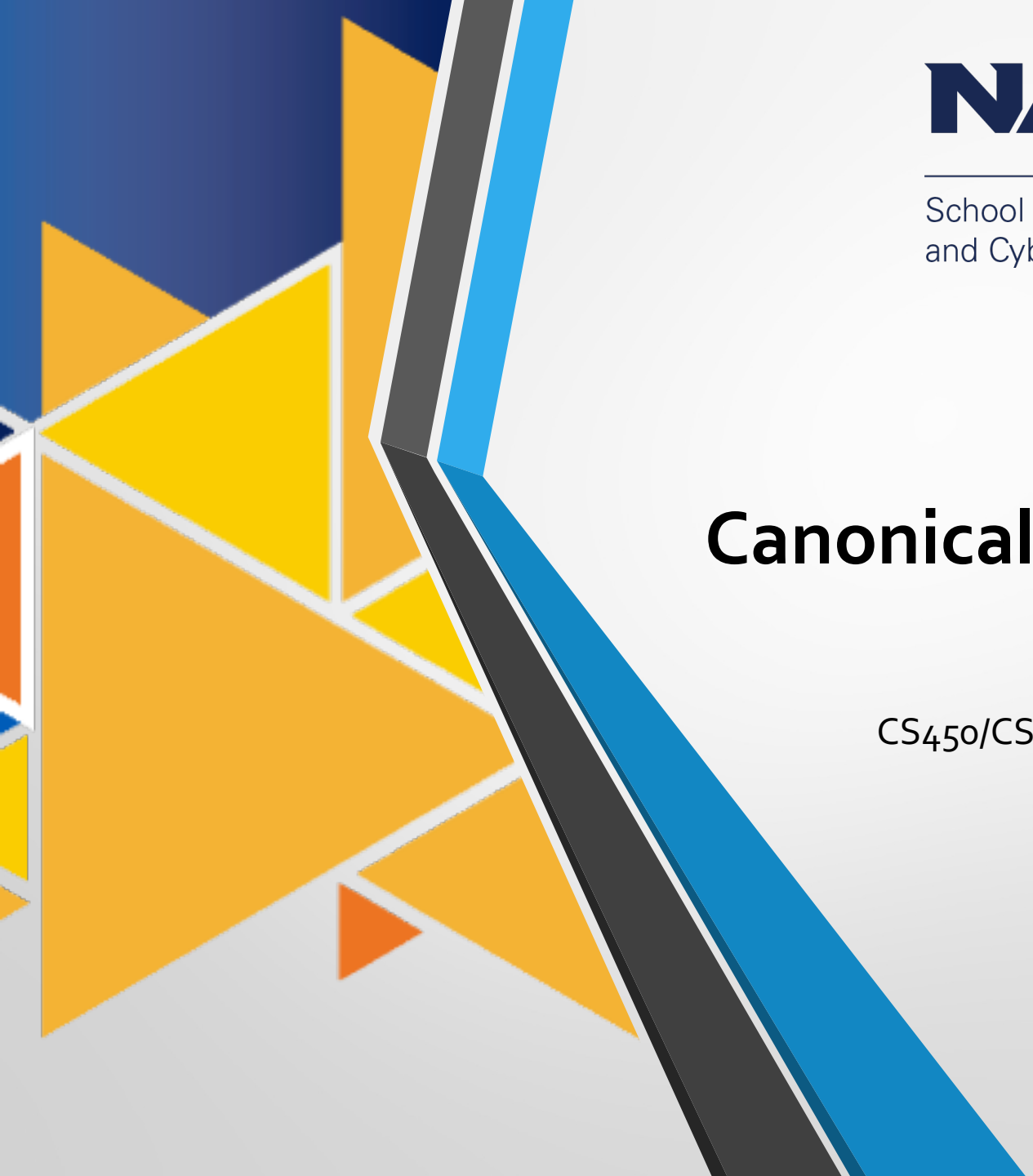


NAU NORTHERN ARIZONA UNIVERSITY

School of Informatics, Computing, and Cyber Systems

Canonical concurrency problems

CS450/CS550: Parallel Programming

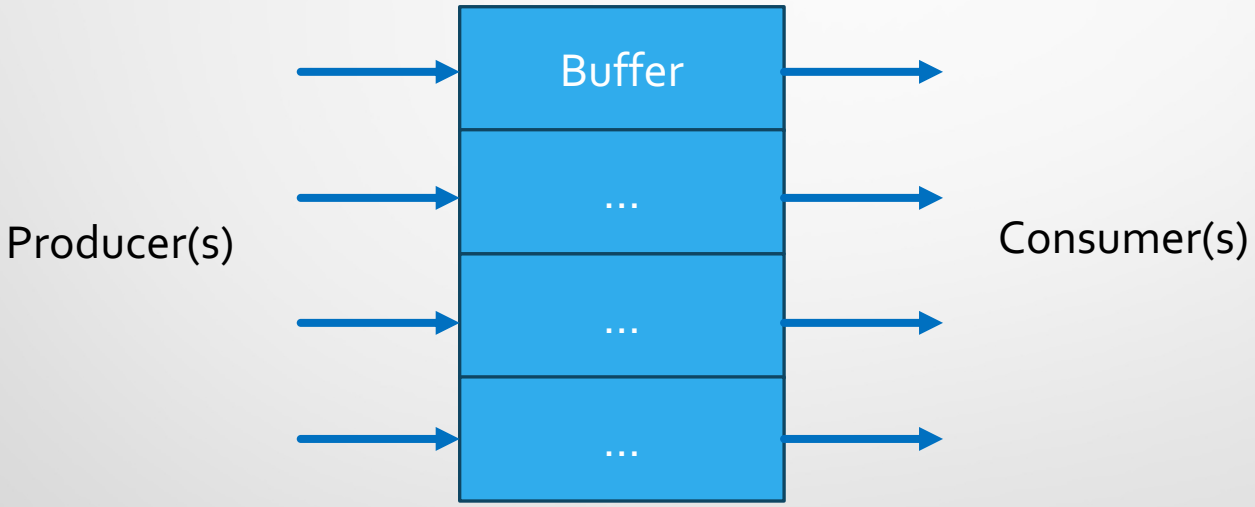
Week 5

General information

- Most canonical synchronization (concurrency) problems were described by Edsger Dijkstra in 1960's
- Canonical synchronization problems describe the general situations that appear when dealing with concurrency in parallel computing
- The most popular concurrency problems include:
  - Producer-Consumer problem
  - Sleeping barber problem
  - Dining philosophers' problem
  - Readers–Writers problem
  - Cigarette smokers' problem
  - ABA problem, etc.

Producer-Consumer Problem

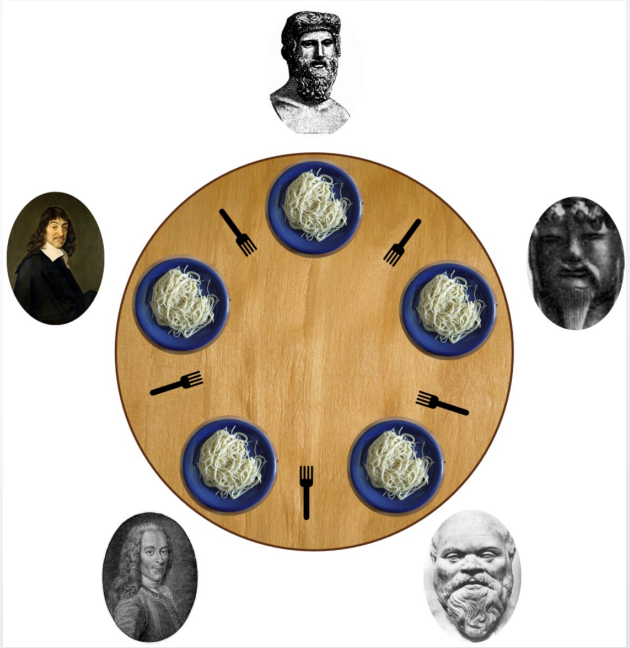
- The producer-consumer problem (bounded-buffer problem)



Example 5-1: Producer-Consumer Problem

Dining philosophers' problem

- Five philosophers dine together at the same table
- Each philosopher can only alternately think and eat
- Each philosopher has their own plate at the table
- There is a fork between each plate
- The food has to be eaten with two forks
- A philosopher can only eat when they have both a left and right fork: forks will only be available when their two nearest neighbors are thinking, not eating
- After a philosopher finishes eating, he puts down both forks
- The problem is how to design a concurrent algorithm such that no philosopher will starve

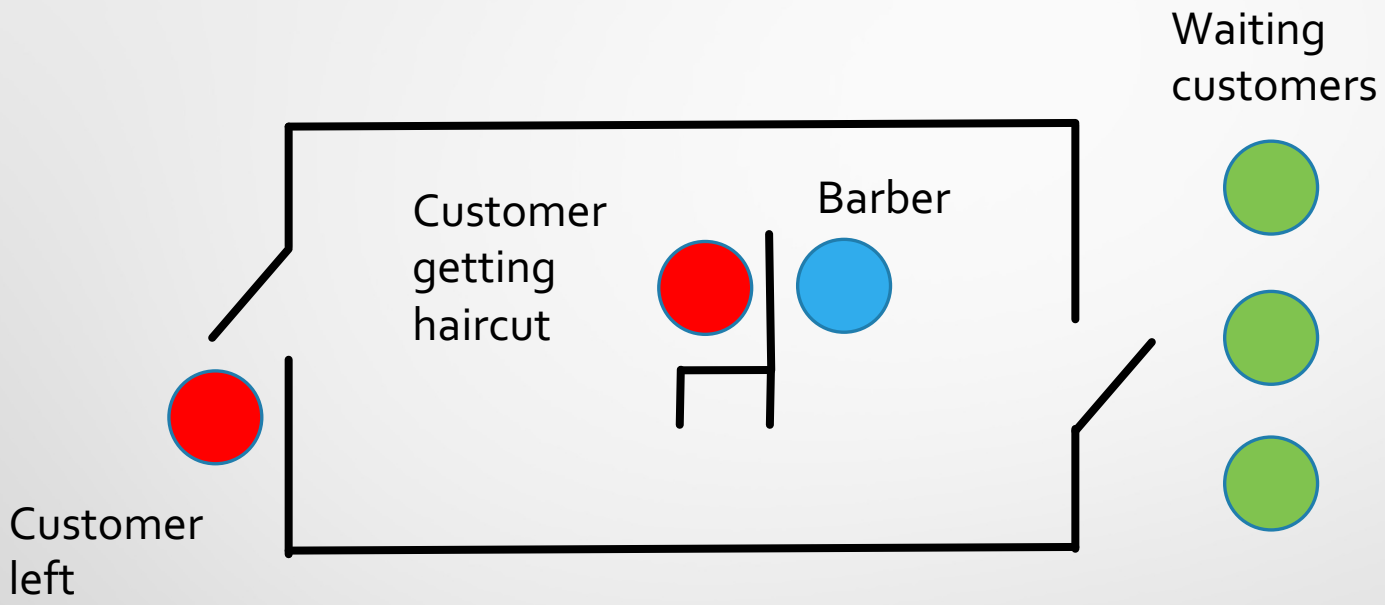


Example 5-2: Dining Philosophers Problem

Barbershop Problem

- A hypothetical barbershop with one barber, one barber chair, and a queue of waiting customers
- The following rules apply:
  - If there are no customers, the barber falls asleep in the chair
  - A customer must wake the barber if he is asleep
  - If a customer arrives while the barber is working, the customer waits to be serviced
  - When the barber finishes a haircut, he checks if there are any waiting customers and falls asleep if there are none

Barbershop Problem



Barbershop Problem: Barber routine

- Opens the door to the shop
- Waits for a customer
- Gives a haircut
- Tells the customer to leave
- Waits until the customer has left through the back of the store (not the front, where the customers wait)

Barbershop Problem: Customer routine

- Wait for the door to open
- Enters the barber shop
- Waits until the barber is available
- Waits until the haircut is finished
- Leaves the shop through the back door

Example 5-3: Barbershop Problem

Assignment #2

- Consider the process of servicing the requests for deliveries by a freight forwarding company:
  - There are *N* dispatchers involved in the servicing process
  - There are *K* requests in a flow of requests representing the demand
  - Each dispatcher can serve any request
  - Requests appear at random time
  - The duration of servicing a single request is random
  - If the dispatchers are busy at the moment when a request appears, the request is declined (there's no queue)

Assignment #2

- Dispatcher's routine:
  - Check if there is a request to serve
  - If there is a request, serve it (servicing time is random), otherwise – wait for a request to appear
- Request's routine
  - Check if there's an available dispatcher
  - If none of the dispatchers is available, leave
- The result of simulations is a servicing level – the number of serviced requests divided by the demand size