Sleeping Barbers Problem

EOPSY Laboratory 5 - IPC

Kacper Kamieniarz (293065)

Problem formulation

The analogy is based upon a hypothetical barber shop with many barbers serving women and man. In a barber shop there are N1 barbers serving only women, N2 barbers serving only men and N3 barbers serving both women and men. Each barber has one barber's chair in a cutting room. In a waiting room there are M chairs. When the barber finishes cutting a customer's hair, he dismisses the customer and goes to the waiting room to see if there are others waiting. If there are, he brings one of them (but only if he is able to cut hair, i.e. the barber brings a man into the cutting room only if he can cut his hair, etc.) back to the chair and cuts hair (it lasts a random time). If there are none, he returns to the chair and sleeps in it. Each customer can be either a male client or a female client. When a client arrives, he/she checks what barbers are doing. If there is any barber sleeping (who is able to serve the client), the customer wakes him up, and sits in the cutting room chair. If all barbers (able to serve the client) are cutting hair, the customer stays in the waiting room. If there is a free chair in the waiting room, the customer sits in it and waits their turn. If there is no free chair, the customer leaves.

Implementation

My solution to this problem is implemented in C language using semaphores <sys/sem.h> library. There are 3 different semaphore kinds used in total, one semaphore for the waiting room (structure with shared variables defining the state of the waiting room), set of semaphores for each type of client and a set of semaphores for each type of barber.

Shared variables

A structure with shared memory containing the variables with the waiting room state (free chairs and waiting customers of each gender) is declared globally and allocated in main.

Main function

Initialization
Within the main function, at first the shared memory segment is allocated and the variables are set to the initial state values - the waiting room is empty. Next thing is initialization of the semaphores using `semget` with `IPC_CREAT` flag and setting them to their initial values using semctl. All client and barber semaphores are initially set to 0 and the waiting room semaphore is set to 1 which means it's unlocked.

Barber loop
Barber loop is the loop within main function used to create barber processes using `fork()` function. Number of iterations is the sum of all the barbers defined and the whole interval is divided as follows:

- Female barbers are in the interval `0 <= i < FEMALE_BARBERS,`
- Male barbers are in the interval `FEMALE_BARBERS <= i < MALE_BARBERS,`
- Universal barbers are in the interval `MALE BARBERS <= i <FEMALE_BARBERS + MALE_BARBERS`

Client loop
Client loop is right after the Barber loop in the main function. The gender of each of the client processes is randomized within this loop while creating processes using `fork()` function. The life cycle of each of the clients is such that after its creation, a client enters the barber shop and after being served or leaving because of no chairs available, client waits for some randomized time (between 2 and 5 seconds) using `sleep()` to pay another visit in the barber shop.

Barber function

Barber function contains an infinite while loop. Because of the third kind of barber - the one that can serve both male and female clients, an integer variable `client_type` is introduced (next to int type passed in the function parameter which defines the barber type). In case the barber is `UNIVERSAL`, after the waiting room variable is locked, a check is made whether the amount of female clients is bigger than the amount of male clients in the waiting room. Based on that condition, `client_type` is set. After client_type is decided, the semaphore for this client is then decremented (down) to set this specific barber to respond to client of given gender. Next comes the critical region - accessing shared variables where the waiting room state is updated. At last the barber semaphore is incremented and waiting room semaphore is unlocked.

Client function

When a client enters the barber shop, the waiting room semaphore is locked so that the client process can access the shared variables to check the state of the waiting room. If there are free chairs, the client takes a seat and the waiting room state is updated. Client semaphore is then incremented and waiting room semaphore unlocked. Because of the `UNIVERSAL` kind of barber - a client who is inside the waiting room must first decide which kind of barber they want to wake up. A check of semaphore values is made using semctl function with flag `GETVAL` and based on that condition barbar_type is set to `UNIVERSAL` in case there are more `UNIVERSAL` barbers than the type corresponding specifically to the client's gender. At the end the barber semaphore for given `barber_type` is decremented. In case there are no free chairs in the waiting room at the start of Client function, the client leaves the barbershop.

Semaphores handling

The semaphores handling is achieved using `<sys/sem.h>` library. For operations on the semaphore (up and down) semop function is used. At initial stage for setting the semaphores, semctl function with flag `SETVAL` is used and for checking the value when deciding on barber choice, semctl with `GETVAL` is used.