CS431 : Assignment - 1

Devaishi Tiwari (170101021)

---

1. Socks Matching Robot

There is a heap of new labelled socks. Each sock is having one of four colors: **white, black, blue** and **grey**. There are several **robotic arms** which can pick up a single sock at a time and pass it to a **matching machine**. The matching machine is able to find two socks of the same color and pass the pair of socks to a **shelf manager robot**. The shelf manager robot then puts the pair of socks to the appropriate shelf.

1.1 Role of Concurrency and Synchronization

Concurrency

We have **multiple robotic arms** available in the system and they are executing concurrently to pick a sock from the heap and pass it to the matching machine. Moreover, the **sock matcher and the shelf manager** are also working concurrently with these robot arms.

Synchronization

We can access the heap of socks concurrently by the robotic arms but no two robotic arms should pick the same sock. And thus the **picking up of a sock** by robotic arms needs to be synchronized. Adding new socks in the matching machine and matching of the existing socks is also done synchronously. Similarly, the **arrangement of socks** by the shelf manager and **addition of new socks** to shelf manager is done synchronously.

1.2 Handling Concurrency and Synchronization

Concurrency

We are using **multithreading** to achieve concurrency. Each concurrent process is allotted a thread, and all threads run parallely.

Synchronization

We are using **semaphores** to achieve synchronization. A semaphore controls access to a shared resource through the use of a counter. We have used individual semaphores to lock each of the sock and then set its counter value **equal to 1**. This ensures that each sock can be picked up by only one robotic arm. We use **block synchronization** method to put a sock in the matching machine's buffer and similarly to put a sock in the shelf manager's buffer.

2. Data Modification in Distributed System

The evaluation process of the B. Tech final year students is going to be conducted for the Programming Languages Lab. The evaluation will be done by the **course coordinator (CC)** and two **teaching assistants (TA1, TA2)** of the department. A file (named **Stud_Info.txt**) is used to record the students' data with various fields, namely **Roll No, Name, Mail_id, Marks** and **Teacher**.

Assumptions

- The file can be updated by TA1, TA2 or CC.
- CC is having higher priority than TA1 and TA2.
- TA1 and TA2 are having equal priorities, i.e. if TA1 is modifying a field of a record, it can be modified by TA2 later and vice versa.
- If CC is modifying any data, that can only be modified by CC later and not by any TAs.
- The file can be assessed by TA1, TA2 or CC simultaneously.
- There can be negative marks for students.

2.1 Importance of Concurrency

The teaching assistants and course coordinator are independent of each other and can update the marks of the students simultaneously. If one of them is updating the marks of any one student, others can update the marks of some other student concurrently. So concurrency is required for allowing them to concurrently update the marks of different students simultaneously to speed up the complete process.

2.2 Shared Resources

The file containing the student marks and details is the shared resources here. The marks of the individual students is a shared resource in our case.

2.3 What if synchronization is not taken care of?

All the marks updates on the same student should be synchronized. If synchronization is not taken care of, some updation of marks may not happen because of independent threads for TAs and CC.

Example

Let some student ABC's initial marks be **50.**
Let us assume TA1 tries increasing ABC's marks by **20** while CC tries to increase it by **15**.

If both updates are executed simultaneously then both TA1 and CC will see the initial marks of ABC's to be 50. Now TA1 will update the value to 70 and the CC will update the value to 65. Both TA1 and CC write back their updated marks for the student. Depending on who writes back first the student's marks as 70 or 65, which it actually should be 95.

2.4 Handling Concurrency and Synchronization

Concurrency

Concurrency is achieved by **multithreading**. Individual threads are created for TA1,TA2 and CC and are being executed concurrently for updating the marks of the students. Note that only the marks of different students can be updated concurrently.

Synchronization

Block synchronization is used to maintain synchronization while accessing a particular student's record. This ensures that only one TA or CC can access the student's record at a time.
