Programming Languages Lab

JAVA

Annanya Pratap Singh Chauhan(170101008)

Sock Matching Robot:

1. **The role of concurrency and synchronization in the above system**
    a. **Concurrency**: Since the task was to simulate the sock sorting system, we need concurrent programming. The role of concurrency here is to simulate arms picking up socks from the heap of socks, simulate a matching robot finding the socks of the same colour and organizing them for shelf manager. All these tasks are happening simultaneously so the system needs to be concurrent.

    b. **Synchronization**: The heap of socks can be accessed simultaneously by all the available robotic arms but no two robotic arms can pick the same sock. Hence picking up of a sock by the robotic arms is being synchronized. Socks are being added continuously to the buffer of matching robot and at the same time matching robot is taking out socks from this buffer. This is also happening synchronously. Matched pairs are also added to the shelf manager synchronously.

2. **How did I handle it?**
    a. **Concurrency**: This is handled by multithreading and parallelising all the robot arm threads, matching robot thread and the shelf thread.

b. **Synchronization**: Synchronization is maintained by using the **Reentrant lock** for each sock and using the **concurrent linked queue** for maintaining our manager buffers and shelf buffers. Each sock has a separate Lock. The locks are used so that at one point only a single thread can pick up a particular sock. The concurrent linked queue helps us in synchronising the buffer of the matching robot. This concurrent linked queue is a thread-safe data structure in which all the insertions and pops are atomic and thus our code is synchronized.

Data Modification in Distributed System:

1. Why concurrency is important here?

As mentioned in the problem we have to allow TA1, TA2, and CC to update the student records simultaneously so concurrency is important. If concurrency is not present then while one evaluator is updating student A's marks other evaluators will not be allowed to update student B's marks. So concurrency is required for allowing them to concurrently update the marks of different students simultaneously and to speed up the complete process.

2. What are the shared resources?

The file "stud_info.txt" is a shared resource. The database in which this data is stored in a shared resource. The database is a hashmap containing information about each student and is used by all the evaluators.

3. What may happen if synchronization is not taken care of? Give examples.

If synchronization is not taken care of then some of the updates will not happen properly. For example, We have a student "X" whose current marks are 30.

Suppose we are going to execute two queries simultaneously:

1. TA1 wants to update this student's marks by +10.
2. TA2 wants to update this student's marks by -10.

If synchronization is not taken care of then both the TAs will see 30 as current marks and update them accordingly so the student might get 40 or 20 as final marks depending on which TA was last to update(race conditions). If synchronization is taken care of then whichever TA gets access to this student's data first will make the update first and then the other TA will update the marks thus final marks will be 30.

4. How you handled concurrency and synchronization?

a. **Concurrency:** Concurrency is achieved by multithreading. Individual threads are created for TA1, TA2 and CC. These threads can then make changes in parallel.

b. **Synchronization:** It is handled by using a semaphore(which is initialised to 1) for each student's record in the database. This allows only one evaluator to make modifications to a particular record at a given time.