Course: COEN 383 Advanced Operating Systems
Professor: Ahmed Ezzat
Group 6: Ankita Rameshwar Sarda, Aditya Shrivastava, Cooper Smith, Meghana Shivaprasad, Shreeraj Suryvanshi Patil

Project 3 Report

Based on the project description, we designed and developed the project. It's a multithreaded project, where we simulate multiple child threads to handle the ticket sales of high-, medium- and low-priced tickets and main thread to synchronize clock ticks. We use mutex locks for accessing critical sections.

We use clock ticks to simulate passing of the minute. Below are the assumptions made to implement the simulation of ticket sales.

a. Clock ticks: the clock ticks are used to synchronize for the child threads to work in increments of time quanta, the lowest the quanta being one minute. The child seller threads serving the customer, waiting for the sale to finish and to call the sales closed are synchronized by clock ticks.

b. Seller Thread States: we simulate 1 high priced seller, 3 medium priced seller, 6 low priced seller threads. Each thread will be in any of the states during any point of time:

   1.If there are any customers in the customer queue, we put that in the seller queue, the thread will be in the waiting state anticipating the new client from the seller queue.

   2. when there is a customer in the seller queue, customer is serviced.

   3. Processing of sale of tickets.

   4. Execution and completion of the ticket sale.

c. for synchronization of time quanta across various sales, new time is created time a seller completes the transaction.

d. The concert hall seating is mimicked by 2D matrix. We use mutex locks while accessing the concert hall seats to prevent the corruption of accessing it at the same time.

Project Workflow

- **Initialization:** Thread creation and initialization for the sellers of each high, medium and low priced sellers. Initialization of mutex locks of critical sections, concert hall seat matrix and creating the customer queues of each sellers of high, medium and low priced sellers.
- **Waiting:** The threads are created and initialized and are waiting for the clock tick. Once the broadcast is called all the threads are signalled and starts executing.
- **Simulating clock tick:** The main thread, responsible for controlling the simulation loop and managing the clock ticks, needs to delay until all other threads have completed their tasks for the current time quanta. By checking the count of active threads, the main thread can determine when it's safe to proceed with the next clock tick.
- **Handling Concert Seat Allocation:** When the main thread signals a clock tick, all seller threads enter a competition to access the concert seat matrix. Their ability to obtain access depends on their current states.
- **Customer Handling:** Initially, each seller thread checks for new clients based on their arrival time. Once identified, they serve one client at a time.
- **Sales Simulation:** To mimic delays in completing sales, we introduce a time delay that decreases with each clock tick. When new customers arrive, a random time delay is generated. Sales for a thread conclude when this delay reaches zero.

Note:

The project's architecture ensures that seat allocation occurs sequentially, while processing occurs concurrently. Consequently, it's probable that all 10 sellers are concurrently serving clients. This means that up to 10 customers may be served simultaneously (assuming n = 10 customers per ticket seller), as evident from the output.