Operating Systems-2

# Programming Assignment 4:
# The Jurassic Park Problem

## Design of the program:

- Synchronization using POSIX semaphores. Three semaphores are used to synchronize the passenger and car threads. They are smphC(semaphore for available cars) initialized to C, smphP(semaphore for waiting passengers) initialized to 0 and mutex(as a lock for shared variables).
- A variable p_left keeps track of the number of passengers who have not ridden k times yet. It is initialized to P. Two variables, t_c and t_p for exponential waiting time for cars and passengers respectively. A queue(p_thread_id) is created to store the id s of passenger threads that are waiting.
- Two arrays are created to store the state of cars and passengers(C_state and P_state). For a car, the state can be 0 (is available), 1 (is waiting for passengers) and 2 (is riding). For a passenger, the state can be 0 (is in the museum), 1 ( is waiting for a car) and 2 ( is riding).
- We take the values of P(no. of passengers), C(no. of cars), k(no. of rides for each passenger), lambda_P(for exponential wait time for passengers) and lambda_C(for exponential wait time for cars).
- P_state and C_state are allocated space of size of int, P and C times respectively. Semaphores are initialized.
- P passenger threads and C car threads are created. Passenger thread runs the function p_ride and car threads runs the function c_ride. Later the passenger threads and car threads are joined with the main thread.
- About the function p_ride: Takes passenger id as argument. Open output file to append the log into it. Set the state of the passenger to 0 indicating that it is in the museum using mutex. Calculate the exponential wait time using

exponential_distribution and default_random_engine. A for loop for riding k times. In the loop, make a ride request by adding the id to the queue(p_thread_id). Signal the smphP semaphore(Increments the semaphore) using mutex before and after. Check if any car is available by accessing the C_state array using mutex for mutual exclusion while accessing the array. If a car is available, set the car state to 1 ( waiting for passenger) and release the mutex. If there is no available car, release the mutex and wait for the car to signal the smphC semaphore. Wait for the car to start riding by acquiring the mutex and checking the state of the car. Set the passenger's state to 2 (riding) and release the mutex. Sleep for random exponential time before making another request after completing this ride. Outside the for loop, decrement the number of passengers still left to ride k times(p_left). Close the output file.

- About the function c_ride: Takes car id as argument. Open output file to append the log into it. Set the state of the carr to 0 indicating that it is available using mutex. Calculate the exponential wait time using exponential_distribution and default_random_engine. Car stays in a loop until there are no passengers that are left to ride. Car waits for passengers to arrive by calling sem_wait on smphP semaphore. Once there are waiting passenger threads, it acquires the mutex and takes the id of the waiting passenger from the queue(p_thread_id) and pops it and updates the state of the car and passenger. After the ride is finished, car state is set back to available state and smphC is signaled. Output file is closed.

## Analysis of the output:

### Plot1 - No. of passengers vs average time taken to complete:

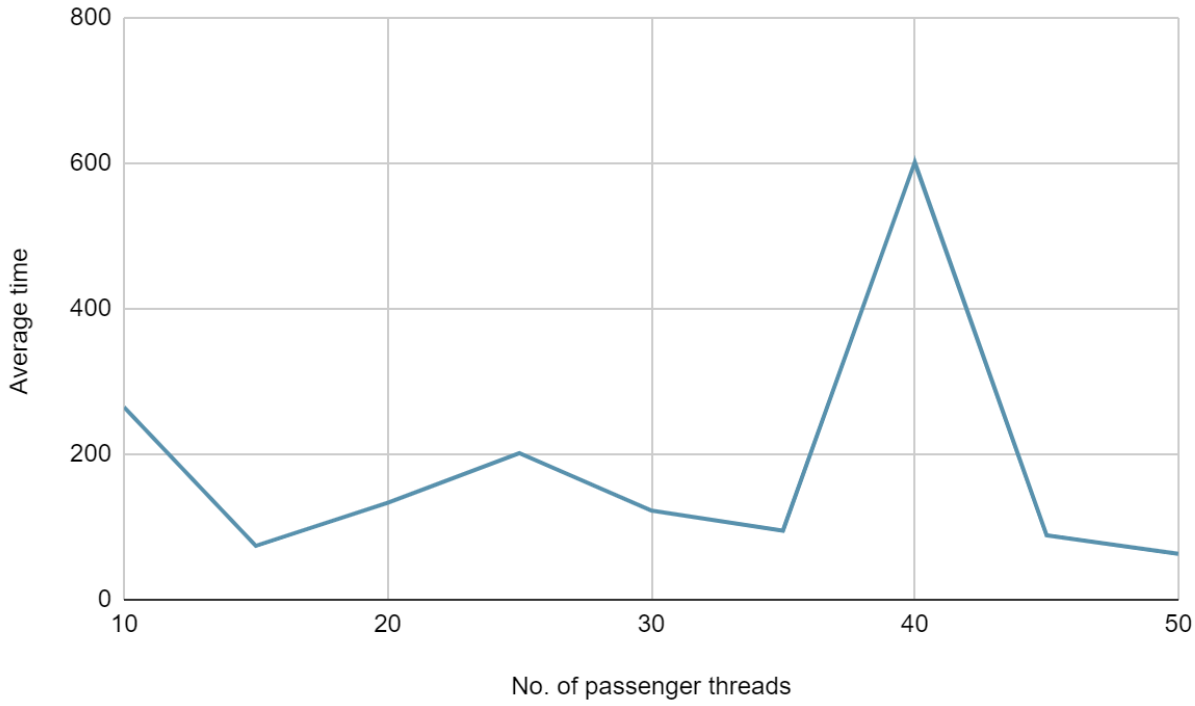
Total number of cars and k value are fixed to 25 and 5 respectively.

X-axis: No. of passengers

Y-axis: Average time to complete their tour

| No. of passengers | Average time |
|---|---|
| 10 | 264.9 |

| | |
|---|---|
| 15 | 74.066 |
| 20 | 133.15 |
| 25 | 201.56 |
| 30 | 122.533 |
| 35 | 95.028 |
| 40 | 601.4 |
| 45 | 88.622 |
| 50 | 63.18 |




### Plot2 - No. of car threads  vs average time taken to complete tour:

The no. of passenger threads and k value is fixed to 50 and 3 respectively.

X-axis: No. car threads

Y-axis: Average time taken to complete the tour

| No. of car threads | Average time taken |
|---|---|
| 5 | 17.8 |
| 10 | 23.7 |
| 15 | 33.66 |
| 20 | 22.5 |
| 25 | 21.64 |