**Name :** Dudekula Dheeraj
**Roll No :** CS22BTECH11019

# Readers-Writers Problem using semaphores

**Report :**

The objective of the code is to solve the readers-writers problem using semaphores. There are two code files, one solution gives preference to the writers while the other one is fair and works on FCFS principle.

The solutions are developed for the following problems:
1. Writers Preference Readers-Writers Problem
2. Fair Readers-Writers Problem

## Low-Level Design

### Input

- Read from a specified file in the command line arguments.
- Includes six numbers: nw, nr, kw, kr, μCS, and μRem.

### Main Function

- Reads and parses input values from the input file.

### Solve Function

- Creates nr reader threads and nw writer threads.

### Writer Function

- Enters a loop to access the critical section.
- Requests access, acquires the write lock, and enters the critical section to perform writing operations.
- Records request, entry, and exit times.
- Simulates writing operations and executes in the remainder section.

### Reader Function

- Enters a loop to access the critical section.
- Requests access, acquires the read lock, and enters the critical section to perform reading operations.
- Records request, entry, and exit times.
- Simulates reading operations and executes in the remainder section.

## Writers Preference Solution

- Writer threads gain priority over readers.

- Increment writecount when a writer attempts to enter the critical section.
- Lock the readTry semaphore if it's the first writer, preventing new readers from entering.
- Unlock the readTry semaphore if it's the last writer, allowing readers to enter again.

## Fair Solution

- Implements fairness using the queue semaphore.
- Acts as a first-come-first-served (FCFS) queue, ensuring threads are serviced in the order they arrive.
- Threads must wait in line to be serviced before accessing the critical section.
- After completing their operation, threads release their spot in the service queue for the next thread.

By regulating access to the critical section using the queue semaphore, the solution ensures fairness in resource access, preventing any thread from being unfairly delayed or starved.

**Graph Observations :**

Table 1 : Average Waiting Time with Constant Writers

| nr | F Reader | F Writer | W Reader | W Writer |
|---|---|---|---|---|
| 1 | 0 | 63 | 0 | 57 |
| 5 | 43 | 75 | 62 | 58 |
| 10 | 114 | 94 | 47 | 50 |
| 15 | 146 | 118 | 67 | 39 |
| 20 | 156 | 122 | 94 | 54 |

Table 2 : Average Waiting Times with Constant Readers

| nw | F Reader | F Writer | W Reader | W Writer |
|---|---|---|---|---|
| 1 | 5 | 24 | 13 | 24 |
| 5 | 46 | 65 | 18 | 21 |
| 10 | 138 | 88 | 58 | 39 |
| 15 | 194 | 142 | 103 | 67 |
| 20 | 113 | 137 | 170 | 117 |

Table 3: Worst – case Waiting Times with Constant Writers

| nr | F Reader | F Writer | W Reader | W Writer |
|---|---|---|---|---|
| 1 | 0 | 254 | 0 | 331 |
| 5 | 202 | 310 | 969 | 135 |
| 10 | 371 | 347 | 924 | 230 |
| 15 | 446 | 393 | 1012 | 221 |
| 20 | 646 | 434 | 1156 | 303 |

Table 4: Worst – case Waiting Times with Constant Readers

| nw | F Reader | F Writer | W Reader | W Writer |
|---|---|---|---|---|
| 1 | 98 | 0 | 69 | 0 |
| 5 | 521 | 87 | 204 | 165 |
| 10 | 1207 | 213 | 291 | 296 |
| 15 | 1462 | 360 | 521 | 540 |
| 20 | 1781 | 586 | 821 | 933 |

**Graphs :**

**Experiment 1 :**

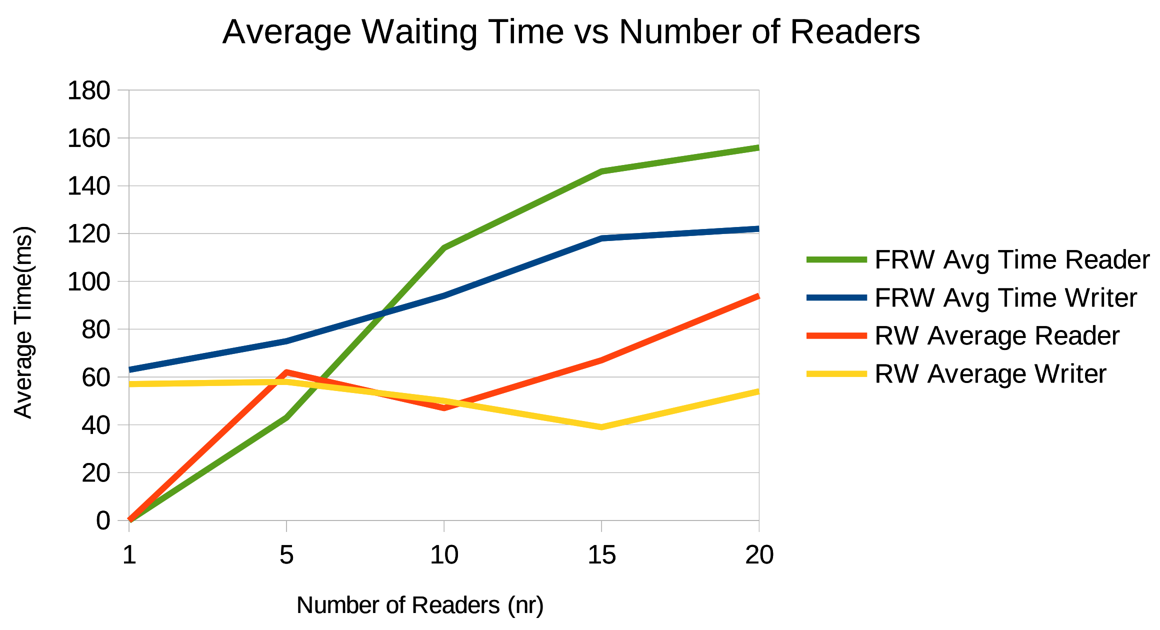


Figure 1 : Time (vs) No of Readers (nr)

**Observations:**

1. Waiting times of writers are least in writer preferred solution, which is expected.
2. Being writer preferred, reader threads take much more time than writer threads in writer preferred solution.
3. The average time is increasing in fair solution with increase in number of readers due to increased contention.

**Experiment 2 :**

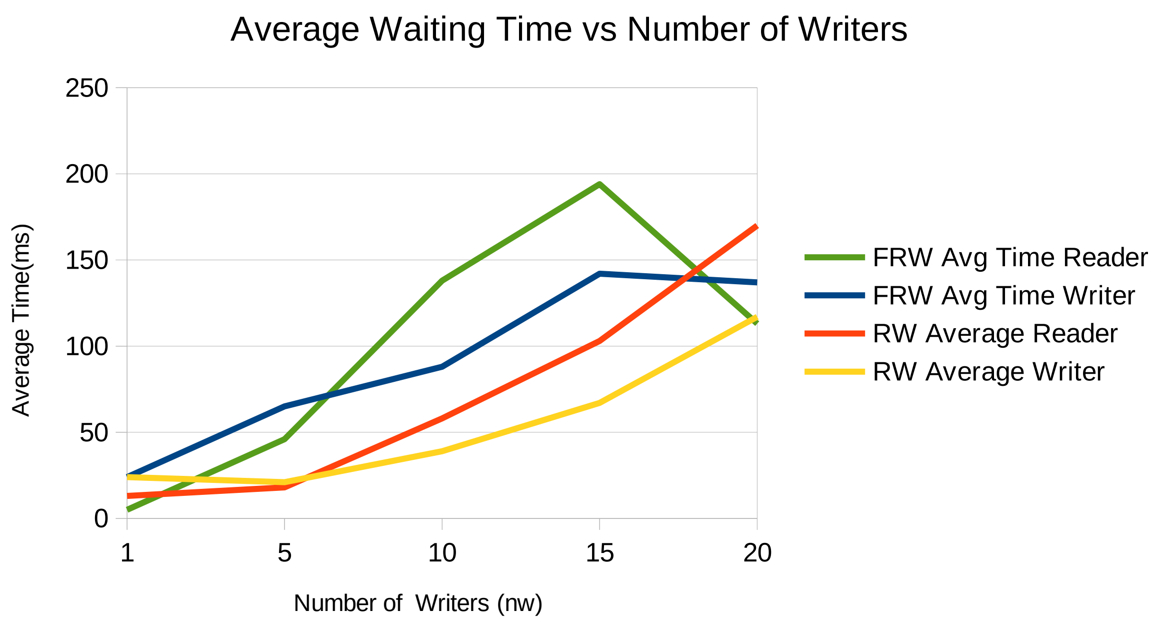


Figure 2 : Time (vs) No of Writers (nw)

**Observations:**

1. The waiting time of threads is increasing with increase in number of writer threads.
2. This behaviour is expected as more threads means more contention and more than one writer threads are not allowed in CS.
3. The waiting time of both types of threads is almost same for the fair solution. While writer has to wait less in writer preferred solution than reader.
4. Threads in fair solution have greater waiting times than those in writer preferred solution due to overhead of queue semaphore.

**Experiment 3 :**

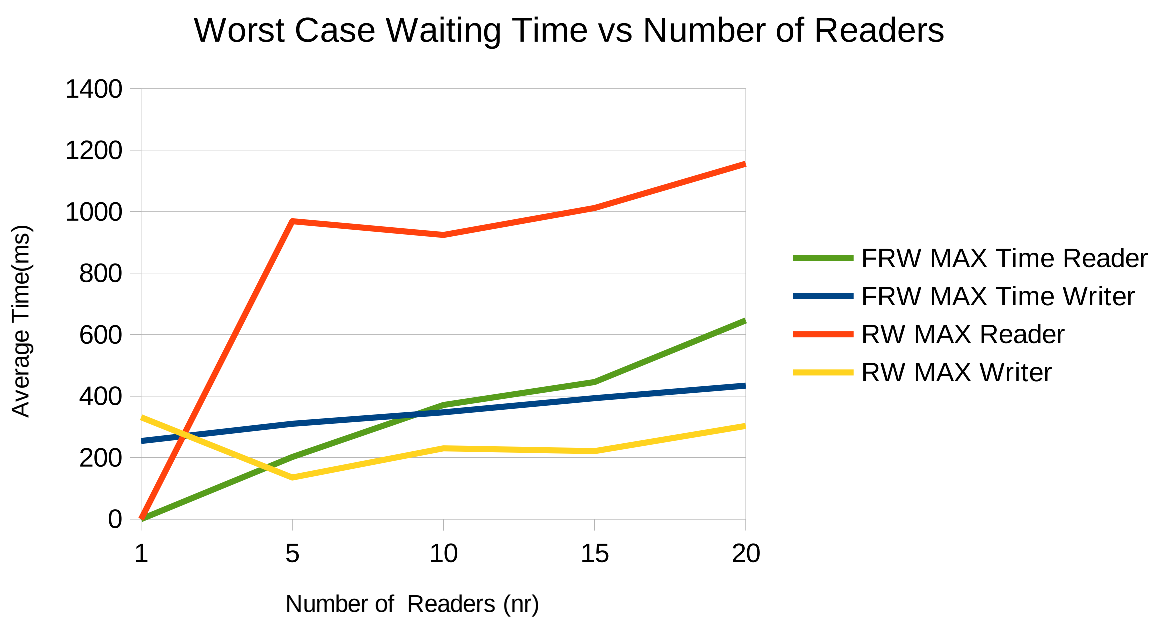
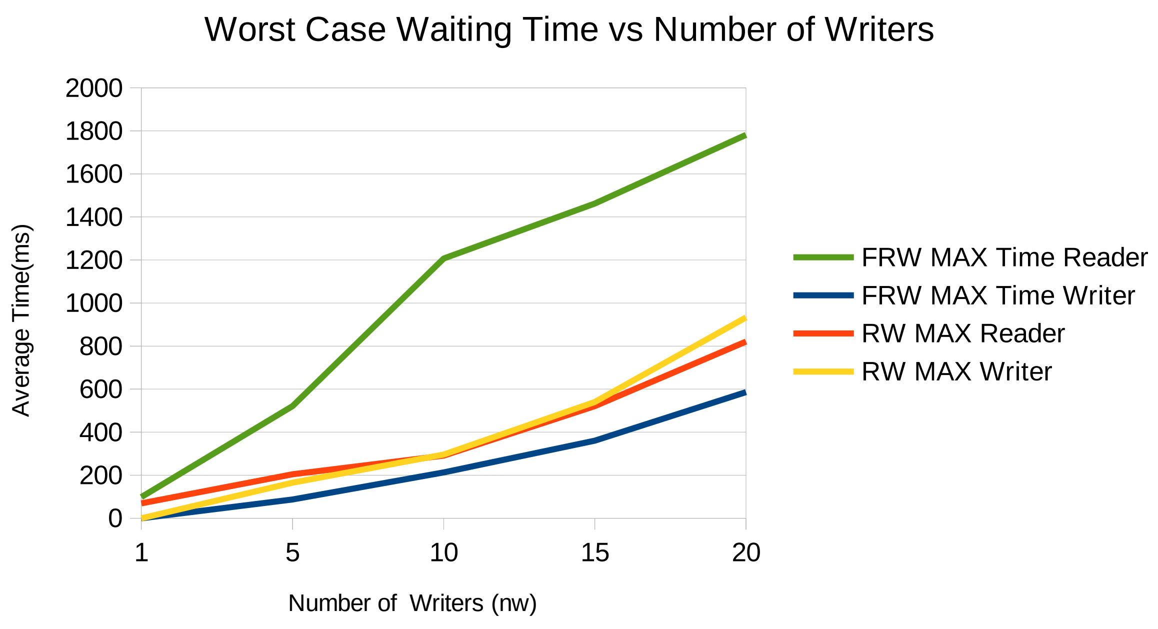


Figure 3: Max. Time vs Number of Readers (nr)

**Observations:**
1. As we can see the large waiting time of reader in writer preferred solution, reader is being starved.
2. Worst case time seems to be increasing with increase in number of threads due to contention.
3. Except reader in writer preferred solution, all other threads have almost same worst waiting time.

**Experiment 4 :**



**Observations:**

1. As expected, the reader is being starved in writer preferred solution.
2. Worst case time is increasing as the number of writers increase due to increased contention.
3. Reader, writer in fair and writer in writer preferred solution have almost the same worst case time.