



CSCE4930-01

Multi-FSM simulator Report

Mohamed Ashraf Taha

ID: 900172754

Submitted to:

Dr. Sherif El-Kassas

Eng. Mahmoud Esmat

5/12/2021

I. Aim

This report aims to explain the changes made to the previous assignment of the FSM simulator. This assignment I introduced the concept of multi-threading and mutexes to the assignment.

II. Changes to FSM simulator

- A. In the FSM simulator I was terminating the FSM by doing `exit(0)` in the `endAction.cpp` `runAction`. However, I changed that since it terminates the whole program and not only the FSM instance running. I changed that by checking in the `runFSM` method by checking if the `ActionType` is `end` we return from the function to the main `TestingFunctionality.cpp`.
- B. Parsed the `mfsm` file and extracted the common variables and the machines names, used functions that I used before in the FSM assignment, so `MFSM` inherits from `FSM`. Then I stored the common variables in a vector of type `VAR`, which is a class that I already implemented from the last assignment. Then I passed the vector of common variables by reference (&) to the `FSM` and from the `FSM` I passed to `parse states` method then from `parse states` to the following three actions: `add`; responsible for handling the expressions, `wait`; responsible for waiting for user input from the screen and `out`; responsible for outputting to the screen values or string,
- C. For the mutexes part, my code mainly depends on two mutexes, one for the `add "Expression"` action and one for **both** the `wait` and the `out` actions.

Then I lock before the expression. I also lock, using the same mutex the wait and the out actions since we dont want a thread to out something while the user is inputting.

III. Debug of complex example

mfsmTwo:

```
Activities Applications Text Editor
Open
1 MFSM mFsmTwo
2 COMVAR X, Y, Z
3 machines:
4 fsmFour
5 fsmFive
6 fsmSix|
```

the three machines:

FsmFour

fsmFive

FsmSix

```
1 FSM fsmFour
2 VAR T
3 States:
4 a: out "state A", X=X+10, sleep 5, wait
5 b: out "state B", T=T+1, sleep 5, wait
6 c: out "thank you for using fsm4", out X, out T, end
7
8 Transitions:
9 a, b, 1
10 b, a, 2
11 a, a, 2
12 b, b, 1
13 b, c, 3

1 FSM fsmFive
2 VAR Q
3 States:
4 a: out "state A", X=X+2, Z=Z+6, sleep 5, wait
5 b: out "state B", Y=Y+1, Q=Q+1, sleep 5, wait
6 c: out "thank you for using fsm5", out X, out Y, out Q, out Z, end
7
8 Transitions:
9 a, b, 1
10 b, a, 2
11 a, a, 2
12 b, b, 1
13 b, c, 3

1 FSM fsmSix
2 VAR S
3 States:
4 a: out "state A", X=X+3, Z=Z+7, sleep 5, wait
5 b: out "state B", Y=Y+3, S=Y+4, sleep 5, wait
6 c: out "thank you for using fsm6", out X, out Y, out S, out Z, end
7
8 Transitions:
9 a, b, 1
10 b, a, 2
11 a, a, 2
12 b, b, 1
13 b, c, 3
```

```
~ For better testing/Output Each thread will create a file with the
output and user can trace any changes to the output as follows:
1. Open a terminal for each machine in the fsm file path, you should
open the terminals in the folder testfiles/outputFiles
2. use the command: tail -F OUTfsm#N.fsm , Where #N is the machine number
in words, i.e One,Two,...

Starting Machines....

"fsmFour" : Action: out statement: "state A"
"fsmSix" : Action: out statement: "state A"
"fsmFive" : Action: out statement: "state A"

"fsmFour": Current State: "a" Waiting for next transition input :

mohamedashraf@mohamedhp:/media/mohamedashraf/HDD/Uni/Uni/Fall 21/00D/Assignments/MohamedTaha_900172754-MFSM_Simulator/testfiles/outputFiles$ tail -F OUTfsmFour.fsm
Action: out "state A"
Output: "state A"
~~~ Running Addition action ~~~
Expression X=X+10
result: 10
~~~ Running SLEEP action ~~~
Action: sleep 5
Output: The FSM is halting for 5 seconds
~~~ Running WAIT action ~~~
Waiting for next transition input CurrentState: a Current Machine: fsmFour

mohamedashraf@mohamedhp:/media/mohamedashraf/HDD/Uni/Uni/Fall 21/00D/Assignments/MohamedTaha_900172754-MFSM_Simulator/testfiles/outputFiles$ tail -F OUTfsmFive.fsm
Expression X=X+2
result: 15
~~~ Running Addition action ~~~
Expression Z=Z+6
result: 13
~~~ Running SLEEP action ~~~
Action: sleep 5
Output: The FSM is halting for 5 seconds
~~~ Running WAIT action ~~~
Waiting for next transition input CurrentState: a Current Machine: fsmFive

mohamedashraf@mohamedhp:/media/mohamedashraf/HDD/Uni/Uni/Fall 21/00D/Assignments/MohamedTaha_900172754-MFSM_Simulator/testfiles/outputFiles$ tail -F OUTfsmSix.fsm
Expression X=X+3
result: 13
~~~ Running Addition action ~~~
Expression Z=Z+7
result: 7
~~~ Running SLEEP action ~~~
Action: sleep 5
Output: The FSM is halting for 5 seconds
~~~ Running WAIT action ~~~
Waiting for next transition input CurrentState: a Current Machine: fsmSix
```

as we can see here:

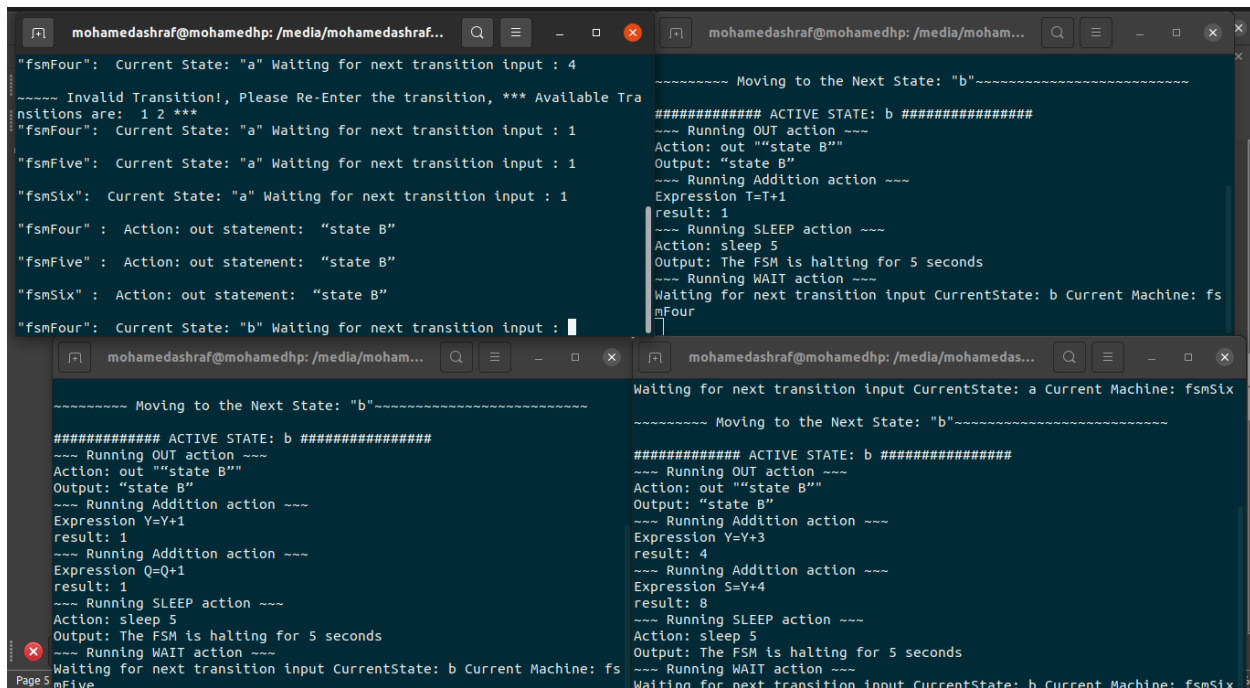
- machine four started first so value of x became 10
- then machine six next so value of x became 13
- then machine five next so value of x is now 15
- since six started first so value of Z became 7, then five started and added $z=z+6 \rightarrow 7+6=13$
- Fault tolerance part

```

"fsmFour": Current State: "a" Waiting for next transition input : 4
~~~~~ Invalid Transition!, Please Re-Enter the transition, *** Available Tra
nsitions are: 1 2 ***
"fsmFour": Current State: "a" Waiting for next transition input : 

```

- then we moved to state B



```

"fsmFour": Current State: "a" Waiting for next transition input : 4
~~~~~ Invalid Transition!, Please Re-Enter the transition, *** Available Tra
nsitions are: 1 2 ***
"fsmFour": Current State: "a" Waiting for next transition input : 1
"fsmFive": Current State: "a" Waiting for next transition input : 1
"fsmSix": Current State: "a" Waiting for next transition input : 1
"fsmFour" : Action: out statement: "state B"
"fsmFive" : Action: out statement: "state B"
"fsmSix" : Action: out statement: "state B"
"fsmFour": Current State: "b" Waiting for next transition input : 

~~~~~ Moving to the Next State: "b"~~~~~
##### ACTIVE STATE: b #####
~~~ Running OUT action ~~~
Action: out "state B"
Output: "state B"
~~~ Running Addition action ~~~
Expression T=T+1
result: 1
~~~ Running SLEEP action ~~~
Action: sleep 5
Output: The FSM is halting for 5 seconds
~~~ Running WAIT action ~~~
Waiting for next transition input CurrentState: b Current Machine: fs
mFour

~~~~~ Moving to the Next State: "b"~~~~~
##### ACTIVE STATE: b #####
~~~ Running OUT action ~~~
Action: out "state B"
Output: "state B"
~~~ Running Addition action ~~~
Expression Y=Y+1
result: 1
~~~ Running Addition action ~~~
Expression Q=Q+1
result: 1
~~~ Running SLEEP action ~~~
Action: sleep 5
Output: The FSM is halting for 5 seconds
~~~ Running WAIT action ~~~
Waiting for next transition input CurrentState: b Current Machine: fs
mFive

~~~~~ Moving to the Next State: "b"~~~~~
##### ACTIVE STATE: b #####
~~~ Running OUT action ~~~
Action: out "state B"
Output: "state B"
~~~ Running Addition action ~~~
Expression Y=Y+3
result: 4
~~~ Running Addition action ~~~
Expression S=Y+4
result: 8
~~~ Running SLEEP action ~~~
Action: sleep 5
Output: The FSM is halting for 5 seconds
~~~ Running WAIT action ~~~
Waiting for next transition input CurrentState: b Current Machine: fsmSix

```

- T (Local variable) became 1 in machine four

- Then Y(global variable became) 1 in machine five, then six ran next and Y became =
1+3 -> 4, then S= Y+4, and Y is now 4 then S = 8

```
mohamedashraf@mohamedhp: /media/mohamedashraf... mohamedashraf@mohamedhp: /media/moham...
"fsmFour": Current State: "b" Waiting for next transition input : 3
"fsmFive": Current State: "b" Waiting for next transition input : 1
"fsmSix": Current State: "b" Waiting for next transition input : 1
"fsmFour": Action: out statement: "thank you for using fsm4"
"fsmFive": Action: out statement: "state B"
"fsmFour": Action: out COMVAR X Value: 15
"fsmSix": Action: out statement: "state B"
"fsmFour": Action: out VAR T : Output: 1
"fsmSix": Current State: "b" Waiting for next transition input :

Waiting for next transition input CurrentState: b Current Machine: fsmFour
Moving to the Next State: "c"
##### ACTIVE STATE: c #####
~~~ Running OUT action ~~~
Action: out "thank you for using fsm4"
Output: "thank you for using fsm4"
~~~ Running OUT action ~~~
Action: out COMVAR X
Output: 15
~~~ Running OUT action ~~~
Action: out VAR T
Output: 1
##### FSM ENDED #####

mohamedashraf@mohamedhp: /media/moham... mohamedashraf@mohamedhp: /media/mohamedas...
Waiting for next transition input CurrentState: b Current Machine: fsmSix
Moving to the Next State: "b"
##### ACTIVE STATE: b #####
~~~ Running OUT action ~~~
Action: out "state B"
Output: "state B"
~~~ Running Addition action ~~~
Expression Y=Y+1
result: 5
~~~ Running Addition action ~~~
Expression Q=Q+1
result: 2
~~~ Running SLEEP action ~~~
Action: sleep 5
Output: The FSM is halting for 5 seconds
~~~ Running WAIT action ~~~
Waiting for next transition input CurrentState: b Current Machine: fsmFive

Waiting for next transition input CurrentState: b Current Machine: fsmSix
Moving to the Next State: "b"
##### ACTIVE STATE: b #####
~~~ Running OUT action ~~~
Action: out "state B"
Output: "state B"
~~~ Running Addition action ~~~
Expression Y=Y+3
result: 8
~~~ Running Addition action ~~~
Expression S=Y+4
result: 12
~~~ Running SLEEP action ~~~
Action: sleep 5
Output: The FSM is halting for 5 seconds
~~~ Running WAIT action ~~~
Waiting for next transition input CurrentState: b Current Machine: fsmSix
```

-Now I ended the fourth machine then it outputted the value of X it holds now, which is 15 and the value of T as well which is 1.

- then machines five and six moved to state B again , so Y became 5 in machine five and then in machine six we made $5+3$, so Y is now 8 and accordingly S is 12, we then increased Q to 2.

```
mohamedashraf@mohamedhp: /media/mohamedashraf... mohamedashraf@mohamedhp: /media/moham...
" fsmFive" : Action: out statement: "thank you for using fsm5"
" fsmSix" : Action: out COMVAR Y Value: 8
" fsmFive" : Action: out COMVAR X Value: 15
" fsmFive" : Action: out COMVAR Y Value: 8
" fsmSix" : Action: out VAR S : Output: 12
" fsmSix" : Action: out COMVAR Z Value: 13
" fsmFive" : Action: out VAR Q : Output: 2
" fsmFive" : Action: out COMVAR Z Value: 13
mohamedashraf@mohamedhp:/media/mohamedashraf/HDD/Uni/Uni/Fall 21/00D/Assignm
ents/MohamedTaha_900172754-MFSM_Simulator$
Waiting for next transition input CurrentState: b Current Machine: fsmFour
~~~~~ Moving to the Next State: "c"~~~~~
##### ACTIVE STATE: c #####
~~~ Running OUT action ~~~
Action: out "thank you for using fsm4"
Output: "thank you for using fsm4"
~~~ Running OUT action ~~~
Action: out COMVAR X
Output: 15
~~~ Running OUT action ~~~
Action: out VAR T
Output: 1
##### FSM ENDED #####

mohamedashraf@mohamedhp: /media/mohamedas...
~~~~~ Moving to the Next State: "c"~~~~~
##### ACTIVE STATE: c #####
~~~ Running OUT action ~~~
Action: out "thank you for using fsm5"
Output: "thank you for using fsm5"
~~~ Running OUT action ~~~
Action: out COMVAR X
Output: 15
~~~ Running OUT action ~~~
Action: out COMVAR Y
Output: 8
~~~ Running OUT action ~~~
Action: out VAR Q
Output: 2
~~~ Running OUT action ~~~
Action: out COMVAR Z
Output: 13
##### FSM ENDED #####

Page 6 ~~~~~ Moving to the Next State: "c"~~~~~
##### ACTIVE STATE: c #####
~~~ Running OUT action ~~~
Action: out "thank you for using fsm6"
Output: "thank you for using fsm6"
~~~ Running OUT action ~~~
Action: out COMVAR X
Output: 15
~~~ Running OUT action ~~~
Action: out COMVAR Y
Output: 8
~~~ Running OUT action ~~~
Action: out VAR S
Output: 12
~~~ Running OUT action ~~~
Action: out COMVAR Z
Output: 13
##### FSM ENDED #####
```

- FSMseven -> Has no variables but has the Keyword VAR

-FSMeight -> Has no VAR section