



The American University in Cairo

CSCE4930-01

FSM simulator User's Guide

Mohamed Ashraf Taha

ID: 900172754

Submitted to:

Dr. Sherif El-Kassas

Eng. Mahmoud Esmat

5/12/2021

User Guide for my FSM simulator:

I. *Compilation: **N.B: added -pthread argument***

Method 1:

make

Method 2:

g++ -pthread -std=c++11 -ggdb -fno-elide-constructors -o MFSMtest *.cpp

II. *Run:*

./MFSMtest testfiles/mFsm#N.mfsm

, where **#N** is the number of the mfsm machine in words, i.e:

One,Two,Three,etc ...

In my case (as an example):

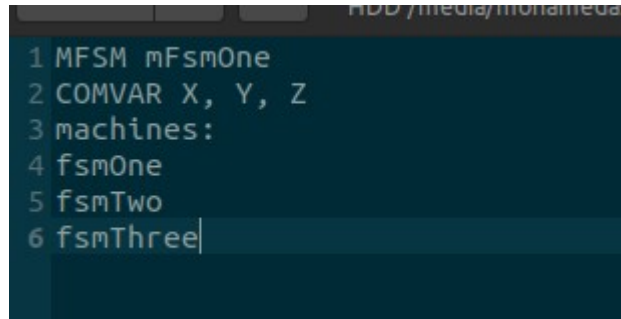
./MFSMtest testfiles/mFsmOne.fsm

III. *How to use the Simulator*

A. *Theoratically*

- A. First step is to create your FSMs each FSM has the its number "In words" ,i.e FsmOne, FsmTwo, etc..
- B. In each FSM, First start by defining the MFsm Machine name, Variables, states, required actions, and transitions.
- C. The machine name will be the same as the file name
- D. Initially all the variables are set to 0
- E. Then the first state in the file will be the initial state to start the machine
- F. Then the user will be prompted to enter the next transition and according to the transition lines stated in the file.
- G. There is also error reporting in the run if the user entered a transition that does not exist for a certain state.
- H. For the MFSM, start by creating a file with prefix mFsm and the mfsm number in words, i.e, mFsmOne, mFsmTwo, mFsmThree, etc.. .

- I. Then define a section called “COMVAR” this section will contain the common variable (global variables) to be shared between all FSMs.
- J. Then Define a section called “Machines” having the list of the FSM machines on separate lines.
- K. Below is a screenshot of how the mFsm file should look like:

A screenshot of a text editor window with a dark background. The text is as follows:

```
1 MFSM mFsmOne
2 COMVAR X, Y, Z
3 machines:
4 fsmOne
5 fsmTwo
6 fsmThree|
```

The cursor is at the end of the sixth line, after 'fsmThree'.

```
1 MFSM mFsmOne
2 COMVAR X, Y, Z
3 machines:
4 fsmOne
5 fsmTwo
6 fsmThree|
```

B. Practically: (this section contains two methods of running and visualizing the output):

1. Terminal outputs:

- Just run the machine and follow the instructions

2. Using the Tail Command which requires outputting to files and tracking the changes that happen to the files as the machines are running:

- *Steps to use this method:*

1. navigate to the outputFiles folder in the submission folder: `cd testfiles/outputFiles`
2. After running the machines open #N number of terminals, where #N is the number of machines in the mfsm file.
3. For example if we have three FSM machines in the MFSM file, we need to open three terminals. And run the command `tail -F OUTfsm#N`, where #N is the number of the machine in words.

Commands (example of three machines):

1. *After running the mfsm*
2. *cd testfiles/outputFiles*
3. open three terminals:
 - A. Terminal1 : `tail -F OUTfsmOne.fsm`
 - B. Terminal2 : `tail -F OUTfsmTwo.fsm`
 - C. Terminal3 : `tail -F OUTfsmThree.fsm`

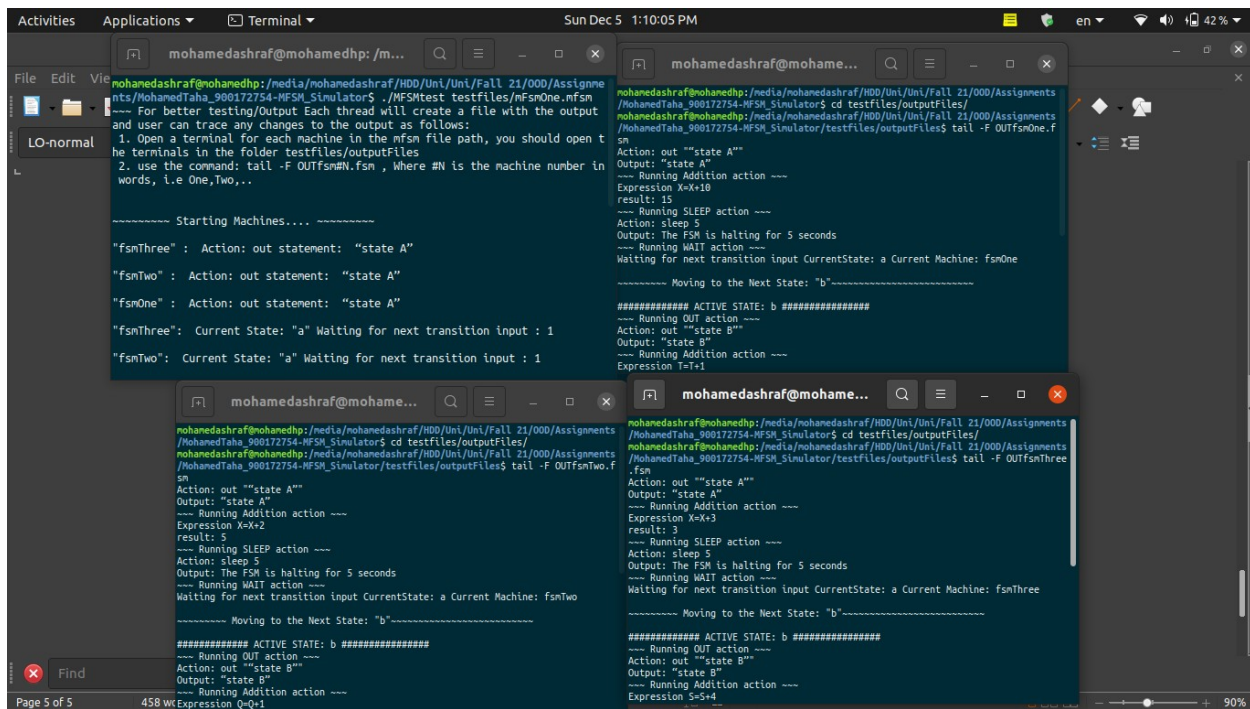
II. Below are screenshots of the output:

Note: Method 2 also supports method 1, method 1 is always used.

A. Method 1:

```
cc@ubuntu:~/repp$  
mohamedashraf@mohamedhp:/media/mohamedashraf/HDD/Uni/Fall 21/000/Assignments/MohamedTaha_980172754-MFSM_Simulator$ ./MFSMtest testfiles/mFsmOne.mfsm  
----- 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 mfsm 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.... -----  
"fsmOne" : Action: out statement: "state A"  
"fsmTwo" : Action: out statement: "state A"  
"fsmThree" : Action: out statement: "state A"  
"fsmOne": Current State: "a" Waiting for next transition input : 1  
"fsmThree": Current State: "a" Waiting for next transition input : 1  
"fsmTwo": Current State: "a" Waiting for next transition input : 1  
"fsmOne" : Action: out statement: "state B"  
"fsmThree" : Action: out statement: "state B"  
"fsmTwo" : Action: out statement: "state B"  
"fsmThree": Current State: "b" Waiting for next transition input : 3  
"fsmOne": Current State: "b" Waiting for next transition input : 3  
"fsmTwo": Current State: "b" Waiting for next transition input : 3  
"fsmThree" : Action: out statement: "thank you for using fsm3"  
"fsmOne" : Action: out statement: "thank you for using fsm1"  
"fsmThree" : Action: out COMVAR X Value: 15  
"fsmOne" : Action: out COMVAR X Value: 15  
"fsmTwo" : Action: out statement: "thank you for using fsm2"  
"fsmTwo" : Action: out COMVAR X Value: 15  
mohamedashraf@mohamedhp:/media/mohamedashraf/HDD/Uni/Fall 21/000/Assignments/MohamedTaha_980172754-MFSM_Simulator$
```

B. Method 2:



```
mohamedashraf@mohamedhp: /media/mohamedashraf/HDD/Uni/Uni/Fall 21/000/Assignments/MohamedTaha_900172754-MFSM_Simulator$ ./MFSMtest testfiles/mfsnOne.nfsn
---- 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 mfsn file path, you should open the terminals in the folder testfiles/outputfiles
2. use the command: tail -F OUTfsn#N.fsn, where #N is the machine number in words, i.e One,Two,..

----- Starting Machines.... -----
"fsnThree" : Action: out statement: "state A"
"fsnTwo" : Action: out statement: "state A"
"fsnOne" : Action: out statement: "state A"
"fsnThree": Current State: "a" Waiting for next transition input : 1
"fsnTwo": Current State: "a" Waiting for next transition input : 1

mohamedashraf@mohamedhp: /media/mohamedashraf/HDD/Uni/Uni/Fall 21/000/Assignments/MohamedTaha_900172754-MFSM_Simulator$ cd testfiles/outputfiles/
mohamedashraf@mohamedhp: /media/mohamedashraf/HDD/Uni/Uni/Fall 21/000/Assignments/MohamedTaha_900172754-MFSM_Simulator/testfiles/outputfiles$ tail -F OUTfsnOne.fsn
Action: out ""state A""
Output: "state A"
---- Running Addition action ----
Expression X=X+10
results: 15
---- 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: fsnOne

----- 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

mohamedashraf@mohamedhp: /media/mohamedashraf/HDD/Uni/Uni/Fall 21/000/Assignments/MohamedTaha_900172754-MFSM_Simulator$ cd testfiles/outputfiles/
mohamedashraf@mohamedhp: /media/mohamedashraf/HDD/Uni/Uni/Fall 21/000/Assignments/MohamedTaha_900172754-MFSM_Simulator/testfiles/outputfiles$ tail -F OUTfsnTwo.fsn
Action: out ""state A""
Output: "state A"
---- Running Addition action ----
Expression X=X+2
results: 5
---- 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: fsnTwo

----- Moving to the Next State: "b"-----
##### ACTIVE STATE: b #####
---- Running OUT action ----
Action: out ""state B""
Output: "state B"
---- Running Addition action ----
Expression Q=Q+1

mohamedashraf@mohamedhp: /media/mohamedashraf/HDD/Uni/Uni/Fall 21/000/Assignments/MohamedTaha_900172754-MFSM_Simulator$ cd testfiles/outputfiles/
mohamedashraf@mohamedhp: /media/mohamedashraf/HDD/Uni/Uni/Fall 21/000/Assignments/MohamedTaha_900172754-MFSM_Simulator/testfiles/outputfiles$ tail -F OUTfsnThree.fsn
Action: out ""state A""
Output: "state A"
---- Running Addition action ----
Expression X=X+3
results: 3
---- 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: fsnThree

----- Moving to the Next State: "b"-----
##### ACTIVE STATE: b #####
---- Running OUT action ----
Action: out ""state B""
Output: "state B"
---- Running Addition action ----
Expression S=S+4
```

```
mohamedashraf@mohamedhp: /m...
" fsmOne" : Action: out statement: "state B"
" fsmOne": Current State: "b" Waiting for next transition input : 3
" fsmTwo": Current State: "b" Waiting for next transition input : 3
" fsmThree": Current State: "b" Waiting for next transition input : 3
" fsmOne" : Action: out statement: "thank you for using fsm1"
" fsmTwo" : Action: out statement: "thank you for using fsm2"
" fsmOne" : Action: out COMVAR X Value: 15
" fsmTwo" : Action: out COMVAR X Value: 15
" fsmThree" : Action: out statement: "thank you for using fsm3"
" fsmThree" : Action: out COMVAR X Value: 15
mohamedashraf@mohamedhp:/media/mohamedashraf/HDD/Uni/Uni/Fall 21/000/Assignme
nts/MohamedTaha_900172754-MFSM_Simulator$

mohamedashraf@mohame...
Output: "state B"
~~~ Running Addition action ~~~
Expression T=I+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: fsmOne
~~~~~ Moving to the Next State: "c"~~~~~
##### ACTIVE STATE: c #####
~~~ Running OUT action ~~~
Action: out ""thank you for using fsm1""
Output: "thank you for using fsm1"
~~~ Running OUT action ~~~
Action: out COMVAR X
Output: 15
~~~ Running OUT action ~~~
Action: out VAR T
Output: 1
##### FSM ENDED #####

mohamedashraf@mohame...
Output: "state B"
~~~ 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: fsmTwo
~~~~~ Moving to the Next State: "c"~~~~~
##### ACTIVE STATE: c #####
~~~ Running OUT action ~~~
Action: out ""thank you for using fsm2""
Output: "thank you for using fsm2"
~~~ Running OUT action ~~~
Action: out COMVAR X
Output: 15
~~~ Running OUT action ~~~
Action: out VAR Q
Output: 1
##### FSM ENDED #####

mohamedashraf@mohame...
Output: "state B"
~~~ Running Addition action ~~~
Expression S=S+4
result: 4
~~~ 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: fsmThree
~~~~~ Moving to the Next State: "c"~~~~~
##### ACTIVE STATE: c #####
~~~ Running OUT action ~~~
Action: out ""thank you for using fsm3""
Output: "thank you for using fsm3"
~~~ Running OUT action ~~~
Action: out COMVAR X
Output: 15
~~~ Running OUT action ~~~
Action: out VAR S
Output: 4
##### FSM ENDED #####
```