System Validation

Student Number: 190336989
Student Name: Xinquan Li
MSc Advanced Computer Science

Name: Xinquan Li   Date:11.23.2019

part-1

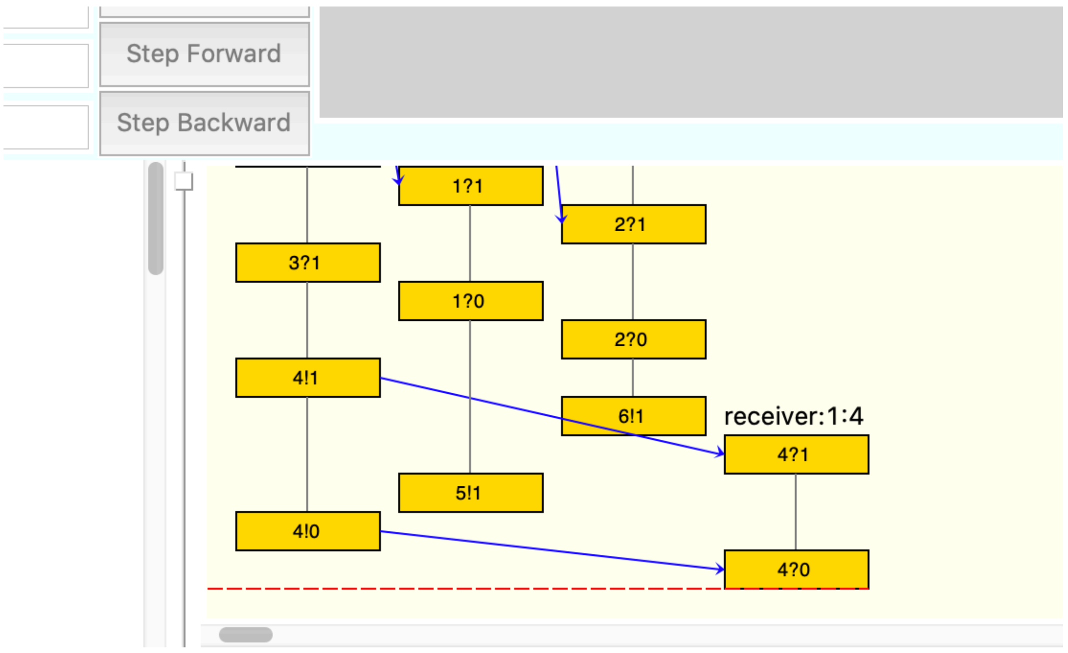
brief explanations

- In the part of prototype proc, ProA , ProB and ProC should have all the data in each prototype proc. So any of them can calculate the password and sent it to Rcv.
- In order to get all data in any outer process, they must send message to each other through channels by looping twice.(eg: A to B, B to C,C to A, A to B, B to C and C to A)
- First each outer process will get some data from init_data[] to my_init_data[], then they will send message to each other. There is a full_data[] to store all the data in three processes. In the 2 loops, there is four situations during sending messages.
- Use this formula(my_pswd[z] = full_data[z] || (full_data[z+2] && full_data[z+4])) to get a password from full_data.
- Finally, the process will send the password to the receiver.
- The length of pswd[] is L. If it sends from A, the id is 0. If it sends from B, the id is 1. If it sends from C, the id is 2. pswd[id*N+w]=psw, they will store different part of pswd[l]. But ,only one process will send it. Just for double check.

validation

```
verification result:
spin -a  procourse.pml
gcc -DMEMLIM=1024 -O2 -DXUSAFE -DNP -DNOCLAIM -w -o pan pan.c
./pan -m10000  -l
Pid: 40181

(Spin Version 6.5.0 -- 1 July 2019)
	+ Partial Order Reduction

Full statespace search for:
	never claim         	+ (:np_:)
	assertion violations	+ (if within scope of claim)
	non-progress cycles 	+ (fairness disabled)
	invalid end states	- (disabled by never claim)

State-vector 200 byte, depth reached 277, errors: 0
     9423 states, stored (13489 visited)
    12531 states, matched
    26020 transitions (= visited+matched)
        8 atomic steps
hash conflicts:        13 (resolved)

Stats on memory usage (in Megabytes):
    2.049	equivalent memory usage for states (stored*(State-vector + overhead))
    1.651	actual memory usage for states (compression: 80.57%)
         	state-vector as stored = 156 byte + 28 byte overhead
  128.000	memory used for hash table (-w24)
    0.534	memory used for DFS stack (-m10000)
  130.097	total actual memory usage


unreached in proctype proc
	procourse.pml:39, state 49, "-end-"
	(1 of 49 states)
unreached in proctype receiver
	procourse.pml:51, state 14, "id = 1"
	procourse.pml:53, state 17, "w = (w+1)"
	procourse.pml:53, state 20, "((w<2))"
```

```
        procourse.pml:53, state 20, "((w>=2))"
        procourse.pml:51, state 22, "from_B?psw"
        procourse.pml:59, state 26, "id = 2"
        procourse.pml:61, state 29, "w = (w+1)"
        procourse.pml:61, state 32, "((w<2))"
        procourse.pml:61, state 32, "((w>2))"
        procourse.pml:59, state 34, "from_C?psw"
        procourse.pml:66, state 37, "-end-"
        (9 of 37 states)
unreached in init
        (0 of 12 states)

pan: elapsed time 0.02 seconds
No errors found -- did you verify all claims?
```



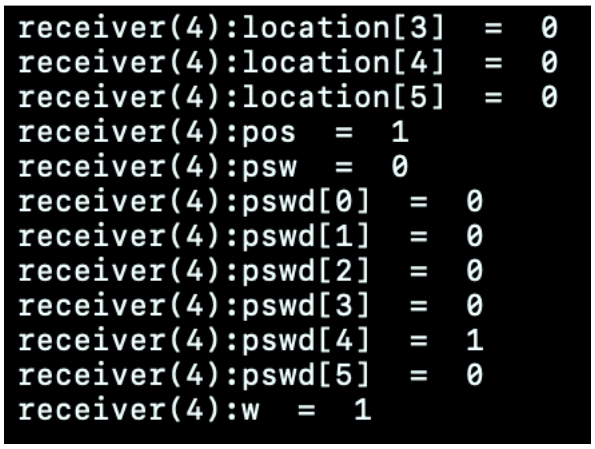
From N=1, i test it several times. N=25 maybe the largest value in my computer, because if it is bigger than 25,i had to wait a long time before it started to simulate.

part-2

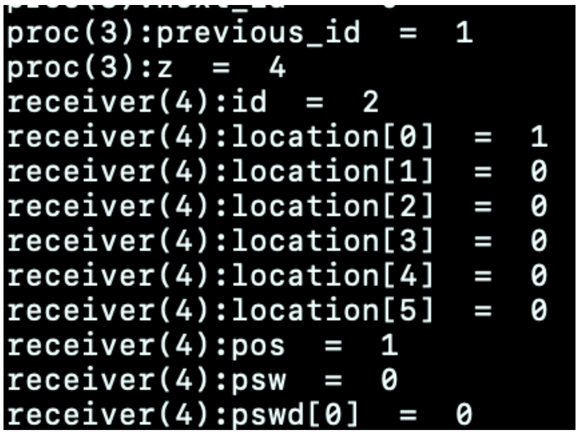
brief explanations

- The channels connected to receiver can duplicate message. So in the outer process , i let them to send the password 2N times in order to imitate this situation.
- The length of pswd[] is L. If it sends from A, the id is 0. If it sends from B, the id is 1. If it sends from C, the id is 2. pswd[id*N+w]=psw, they will store different part of pswd[l]. But ,only one process will send it. Just for double check.
- i define a variables(pos) to represent the location of each password. When (w>=2*N || pos>N-1) it breaks ,do not receive any more.
- The location[pos] means the state of each password. For example, location[pos]=1 means it has received, location[pos]=0 means it does not receive. When location[pos]=0, paswd[L] will store the password.

```
do
    :: from_A ? psw -> id=0->
                                    if
                                     ::(location[w] ==0)-> pswd[id*N+w]=psw;w++;location[pos]=1;pos++
                                    ::(w<2*N ||pos>N-1) ->break
                                    fi

    :: from_B? psw -> id=1->
                                    if
                                     ::(location[pos] ==0)-> pswd[id*N+w]=psw;w++;location[pos]=1;pos++
                                    ::(w<2*N || pos>N-1) ->break
                                    fi
    :: from_C? psw -> id=2->
                                    if
                                     ::(location[pos] ==0)-> pswd[id*N+w]=psw;w++;location[pos]=1;pos++
                                    ::(w<2*N || pos>N-1) ->break
                                    fi
    od
```



```
receiver(4):location[3]  =  0
receiver(4):location[4]  =  0
receiver(4):location[5]  =  0
receiver(4):pos  =  1
receiver(4):psw  =  0
receiver(4):pswd[0]  =  0
receiver(4):pswd[1]  =  0
receiver(4):pswd[2]  =  0
receiver(4):pswd[3]  =  0
receiver(4):pswd[4]  =  1
receiver(4):pswd[5]  =  0
receiver(4):w  =  1
```



```
proc(3):previous_id  =  1
proc(3):z  =  4
receiver(4):id  =  2
receiver(4):location[0]  =  1
receiver(4):location[1]  =  0
receiver(4):location[2]  =  0
receiver(4):location[3]  =  0
receiver(4):location[4]  =  0
receiver(4):location[5]  =  0
receiver(4):pos  =  1
receiver(4):psw  =  0
receiver(4):pswd[0]  =  0
```

validation

verification result:

```
spin -a  procourse2.pml
gcc -DMEMLIM=1024 -O2 -DXUSAFE -DNP -DNOCLAIM -w -o pan pan.c
./pan -m10000  -l
Pid: 40601
pan:1: assertion violated - invalid array index (at depth 217)
pan: wrote procourse2.pml.trail

(Spin Version 6.5.0 -- 1 July 2019)
Warning: Search not completed
	+ Partial Order Reduction

Full statespace search for:
	never claim         	+ (:np_:)
	assertion violations	+ (if within scope of claim)
	non-progress cycles 	+ (fairness disabled)
	invalid end states	- (disabled by never claim)

State-vector 208 byte, depth reached 246, errors: 1
     3412 states, stored (6717 visited)
     4997 states, matched
    11714 transitions (= visited+matched)
        8 atomic steps
hash conflicts:         0 (resolved)

Stats on memory usage (in Megabytes):
    0.768	equivalent memory usage for states (stored*(State-vector + overhead))
    0.772	actual memory usage for states
  128.000	memory used for hash table (-w24)
    0.534	memory used for DFS stack (-m10000)
  129.218	total actual memory usage



pan: elapsed time 0 seconds
To replay the error-trail, goto Simulate/Replay and select "Run"
```