Software Engineering

Assignment 5

Student Details

Name: Krunal Rank
Adm. No: U18CO081

1. There are four philosophers sitting around a round table. There are forks on the table, one between each pair of philosophers. The philosophers want to eat spaghetti from a large bowl in the center of the table. Unfortunately the spaghetti is of a particularly slippery type, and a philosopher needs both forks in order to eat it. The philosophers have agreed on the following protocol to obtain the forks: Initially philosophers think about philosophy, when they get hungry they do the following:

- Take the left fork

- Take the right fork and start eating

- Return both forks simultaneously, and repeat from the beginning.

Build a SPIN model for this scenario.

```
#define NUM 5
bool pthinking[NUM],phungry[NUM],peating[NUM]=false;
int forks[NUM] = -1;
proctype P(int i){
   int right=i;
   int left=(i+1)%NUM;
   Think:
       atomic{pthinking[i]=true;peating[i]=false;printf("%d is thinking.\n",i);};
   Hungry:
       atomic{phungry[i]=true;pthinking[i]=false;printf("%d is hungry.\n",i);};
       if
       :: left<right;
           atomic{forks[left] == -1 -> forks[left] = i;printf("%d grabbed left
fork.\n",i);};
           atomic{forks[right] == -1 ->forks[right]=i;printf("%d grabbed right
fork.\n",i);};
```

```
        :: right<left;
            atomic{forks[right]==-1->forks[right]=i;printf("%d grabbed right
fork.\n",i);};
            atomic{forks[left] == -1 -> forks[left] = i;printf("%d grabbed left
fork.\n",i);};
        fi;
    Eating:
        atomic{peating[i]=true;phungry[i]=false;printf("%d is eating.\n",i);};

    Done:
        forks[right]=-1;
        forks[left]=-1;
        printf("%d is done eating.\n",i);
        goto Think;
}


init {
    atomic{
        int i = 0;
        do
        :: i < NUM -> run P(i);i++;
        :: else break;
        od;
    }

}
```

Output:

(Spin Version 6.4.9 -- 17 December 2018)
+ Partial Order Reduction

Full statespace search for:

never claim	- (none specified)
assertion violations	+
cycle checks	- (disabled by -DSAFETY)
invalid end states	+

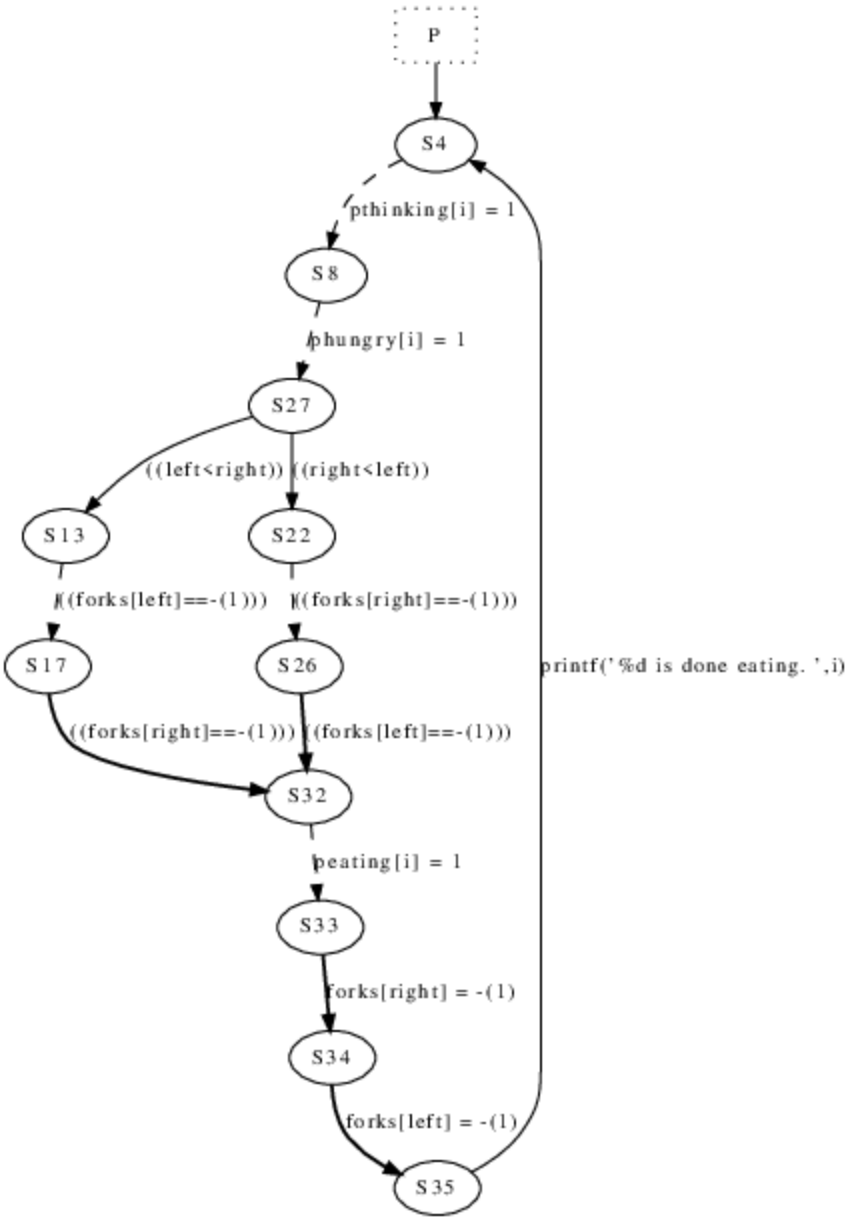
State-vector 120 byte, depth reached 9999, errors: 0
25246 states, stored
66347 states, matched
91593 transitions (= stored+matched)

```
      18 atomic steps
hash conflicts:       24 (resolved)

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


unreached in proctype P
        ./1.pml:27, state 37, "-end-"
        (1 of 37 states)
unreached in init
        (0 of 11 states)

pan: elapsed time 0.04 seconds
pan: rate    631150 states/second
```


P
S4
pthinking[i] = 1
S8
phungry[i] = 1
S27
((left<right))
((right<left))
S13
S22
((forks[left]==-(1)))
((forks[right]==-(1)))
S17
S26
printf('%d is done eating. ',i)
((forks[right]==-(1)))
((forks[left]==-(1)))
S32
peating[i] = 1
S33
forks[right] = -(1)
S34
forks[left] = -(1)
S35