Unit-3: Linear-time properties

B. Srivathsan

Chennai Mathematical Institute

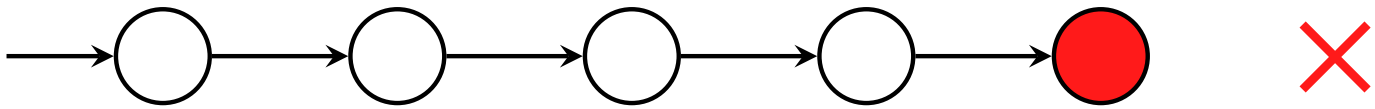
NPTEL-course

July - November 2015

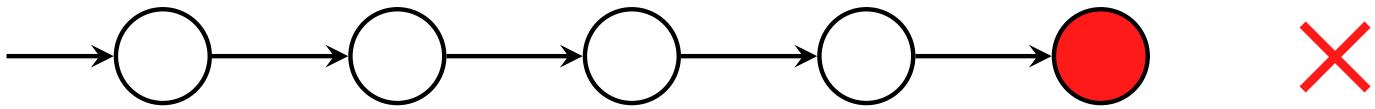
Module 5:

Liveness properties

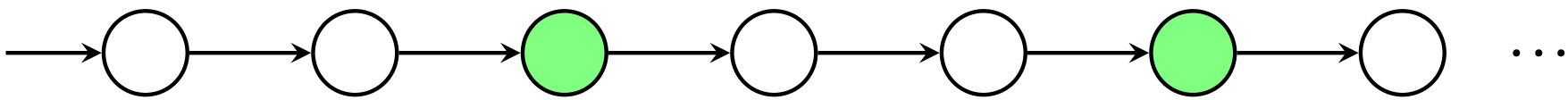
Safety: Something bad **never happens**



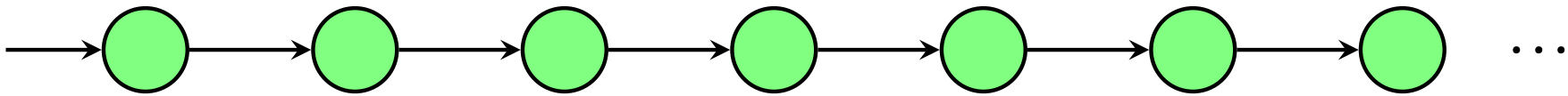
Safety: Something bad **never happens**



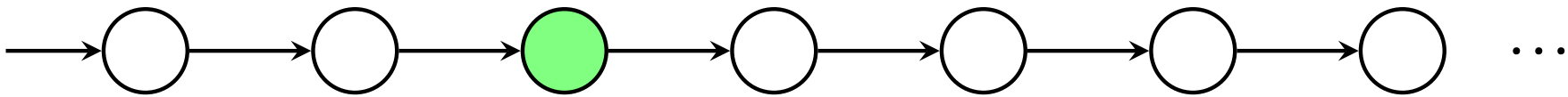
Liveness: Something **good** happens **infinitely often**



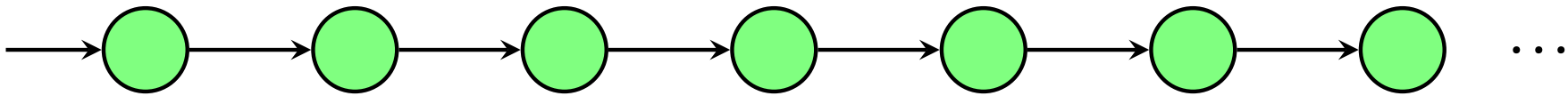
G p: Always p



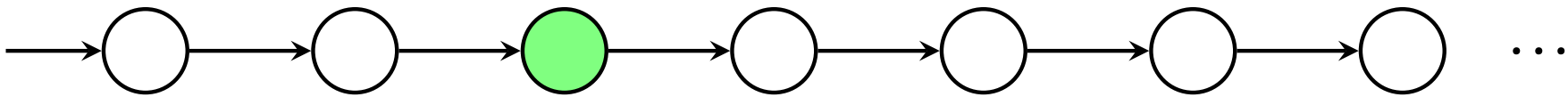
F p: Sometime p



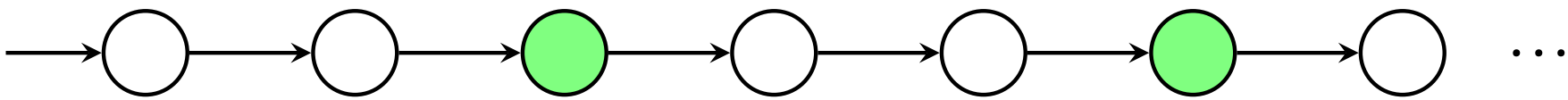
G p: Always p



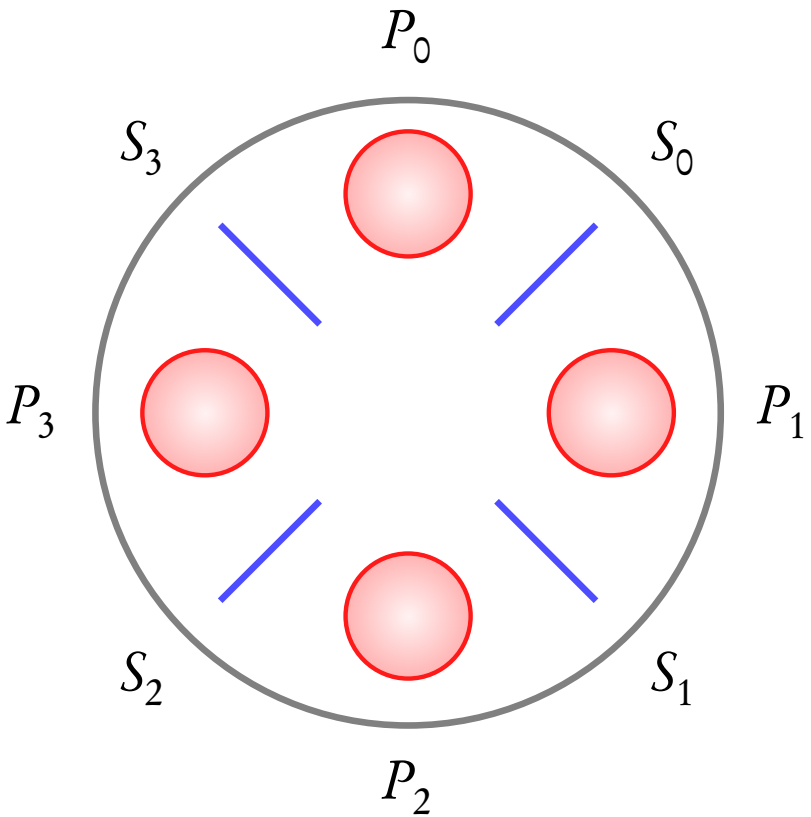
F p: Sometime p



G F p: Infinitely often p



Recall...



$P_0 \ldots P_3$: *philosophers*

$S_0 \ldots S_3$: *chop-sticks*

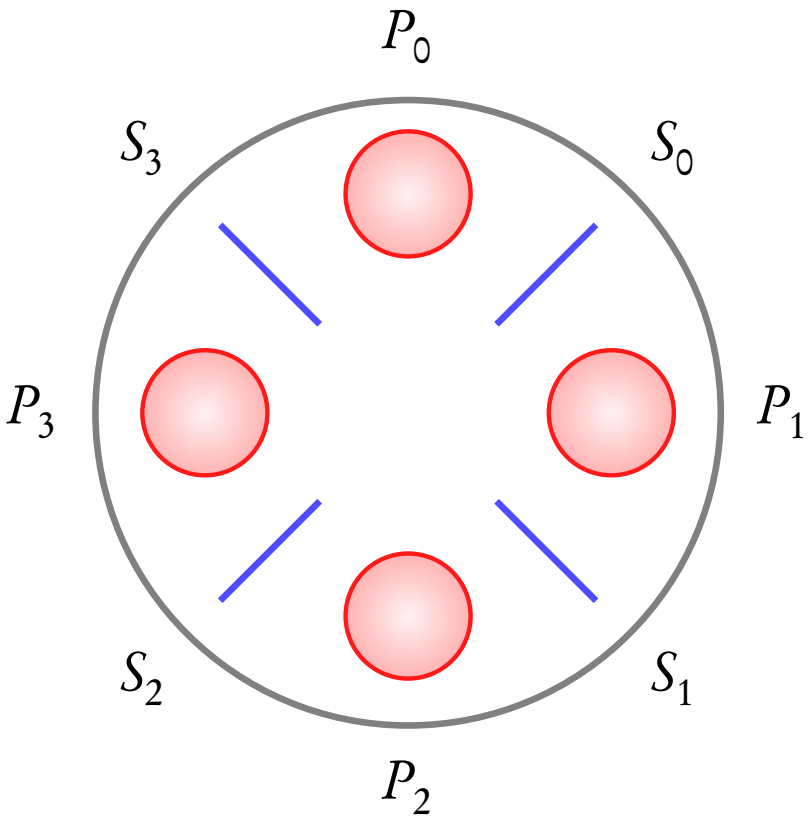
Philosopher P_i can **eat**

only if

he has access to **chop-sticks**

$S_{(i-1) \bmod 4}$ and $S_{i \bmod 4}$

Recall...



$P_0 \ldots P_3$: *philosophers*

$S_0 \ldots S_3$: *chop-sticks*

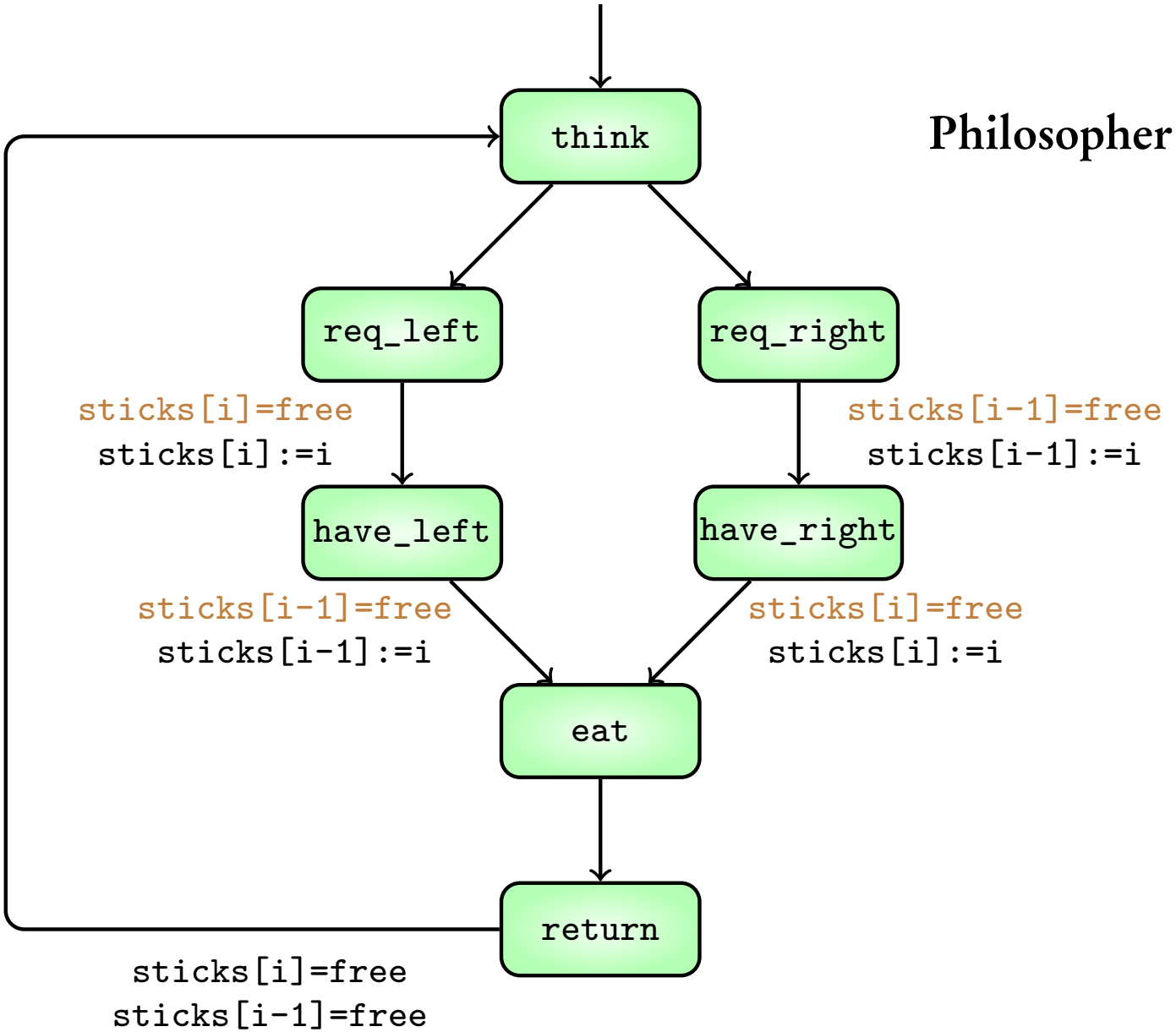
Philosopher P_i can **eat**

only if

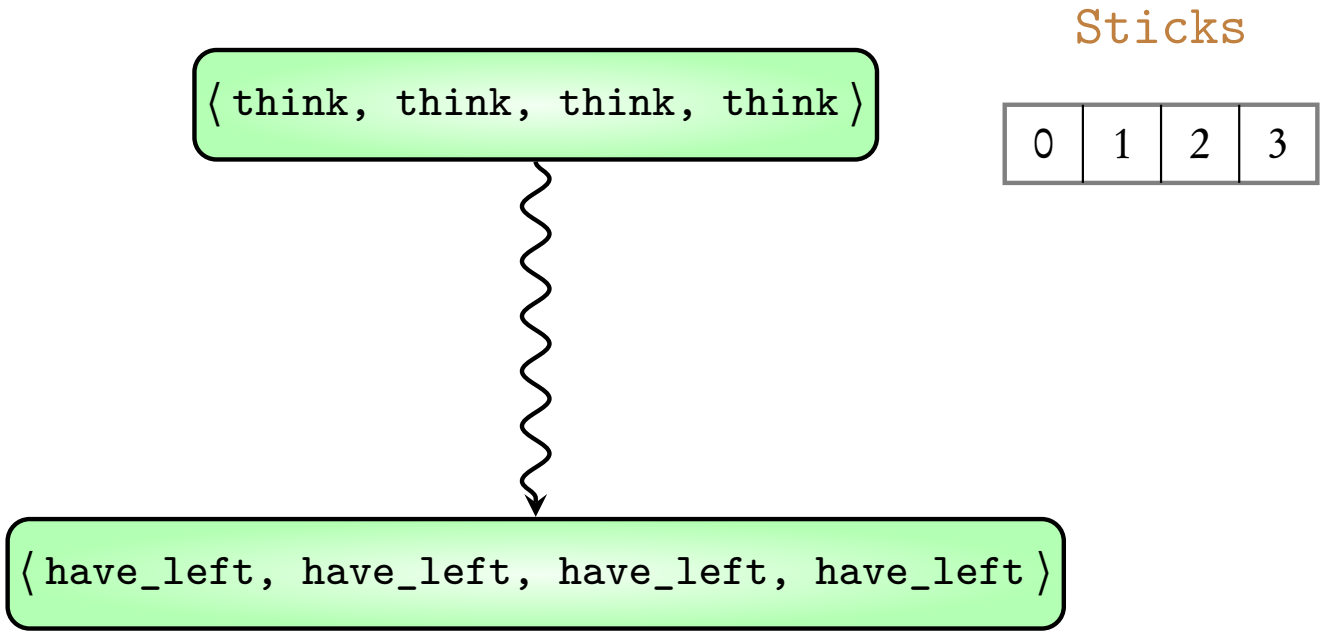
he has access to **chop-sticks**

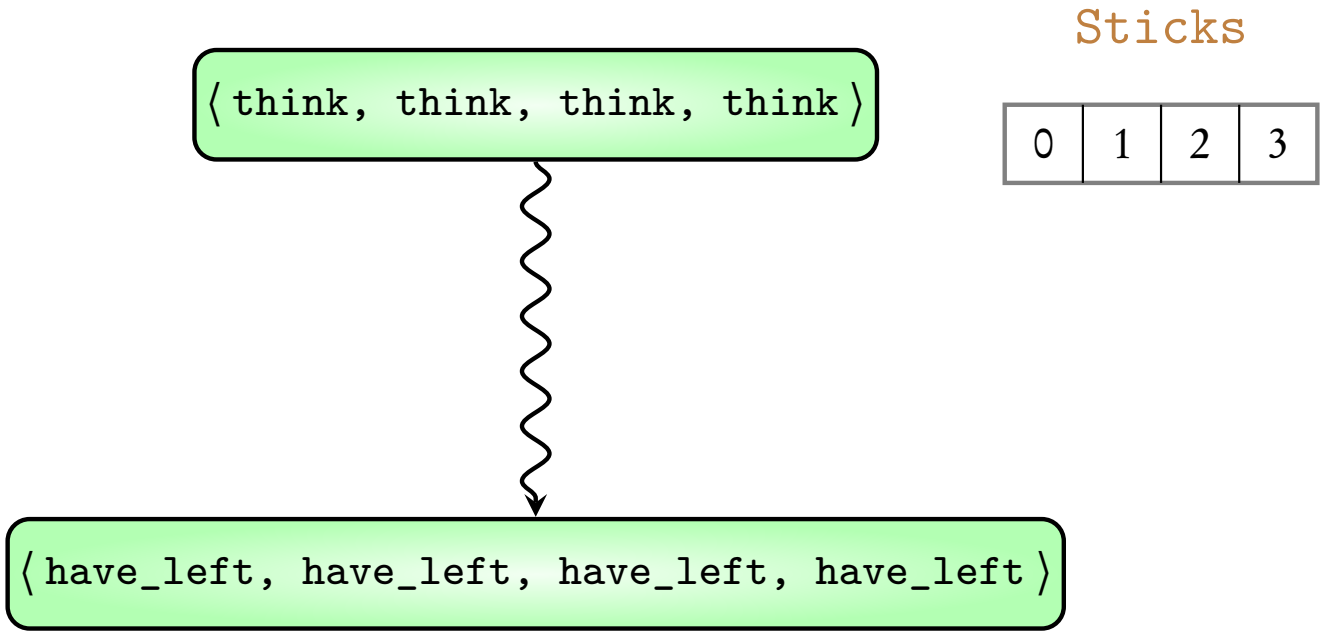
$S_{(i-1) \bmod 4}$ and $S_{i \bmod 4}$

What should the **protocol** be so that **every philosopher** can eat **infinitely often?**

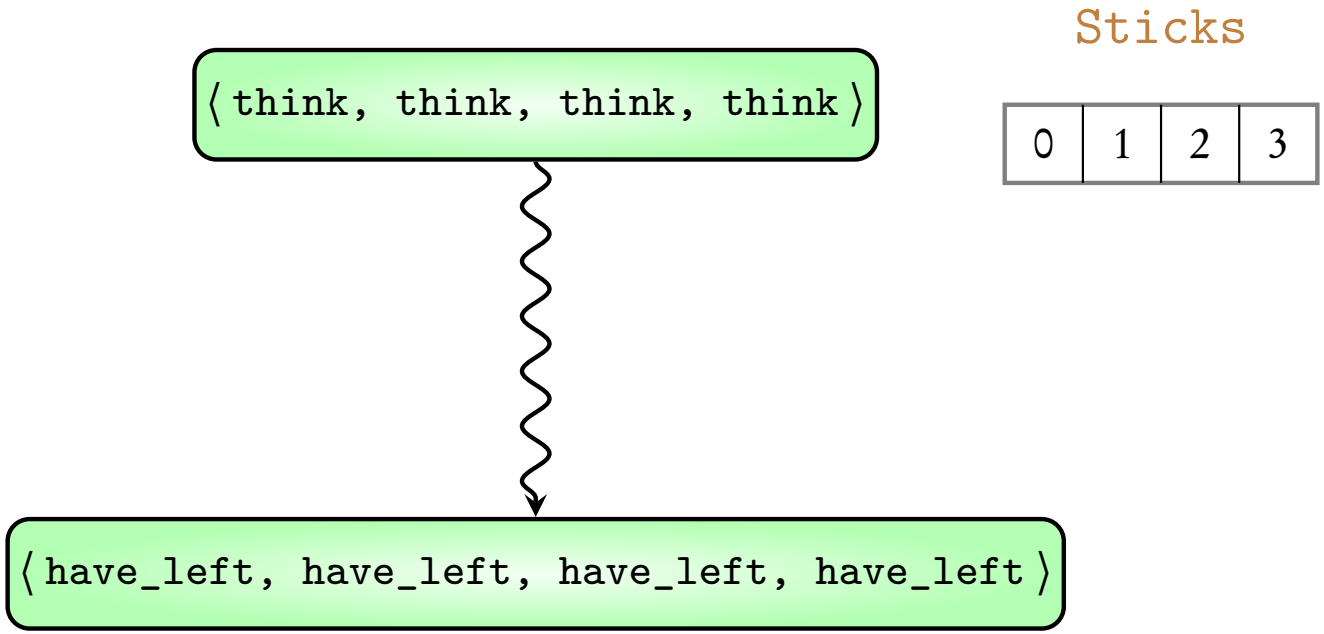


NuSMV code for the protocol


⟨ think, think, think, think ⟩
⟨ have_left, have_left, have_left, have_left ⟩
Sticks
0
1
2
3



What properties should be checked in order to **reveal the deadlock**?



What properties should be checked in order to **reveal the deadlock**?

G F (phil0.location=eat) & G F (phil1.location=eat) &
G F (phil2.location=eat) & G F (phil3.location=eat)

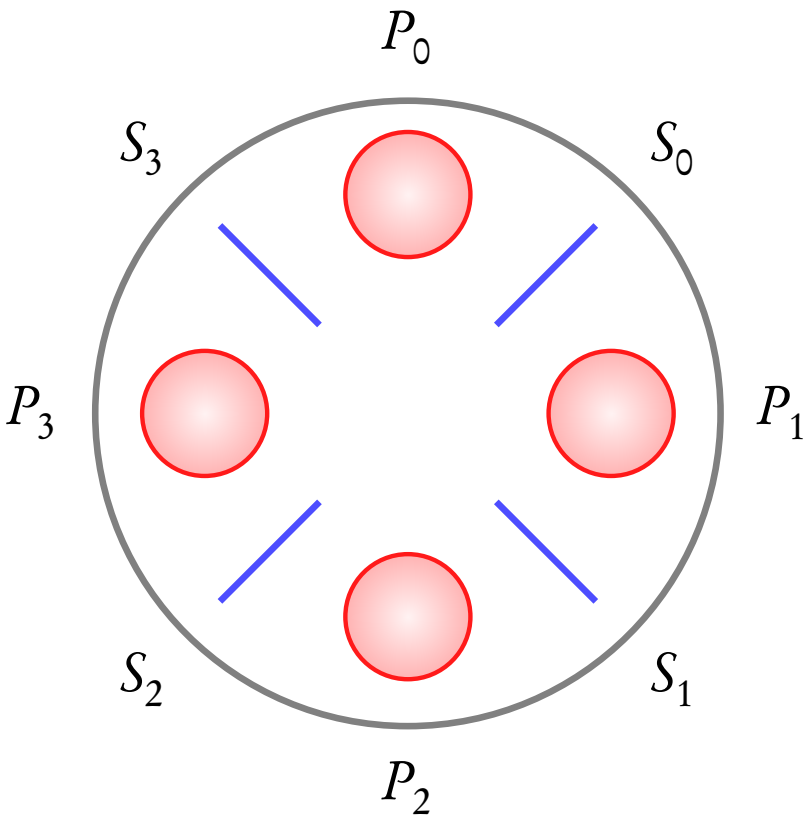
- If **counterexample** is due to only `main` process being scheduled
  - **Not a fair** scheduler

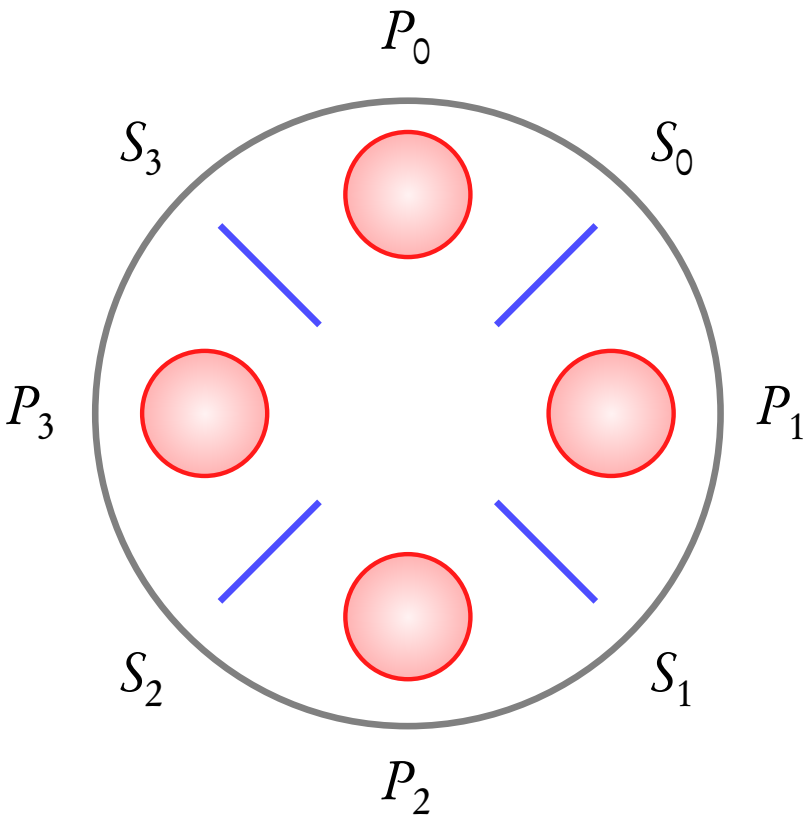
- If **counterexample** is due to only `main` process being scheduled
  - **Not a fair** scheduler
  - Add a `FAIRNESS running` in the philosopher module

- If **counterexample** is due to only `main` process being scheduled
  - **Not a fair** scheduler
  - Add a `FAIRNESS running` in the philosopher module

NuSMV demo

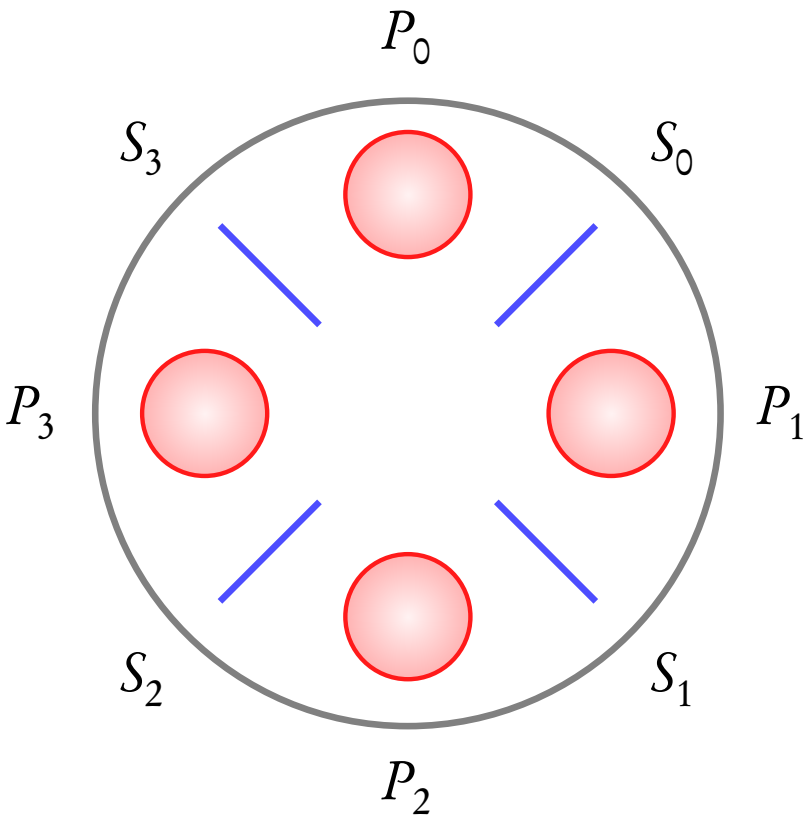
Coming next: Another solution for the dining philosophers problem


P_0
S_3
S_0
P_3
P_1
S_2
S_1
P_2

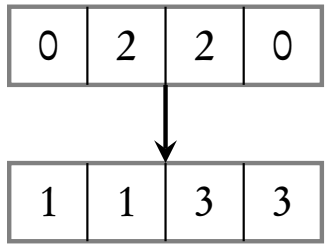


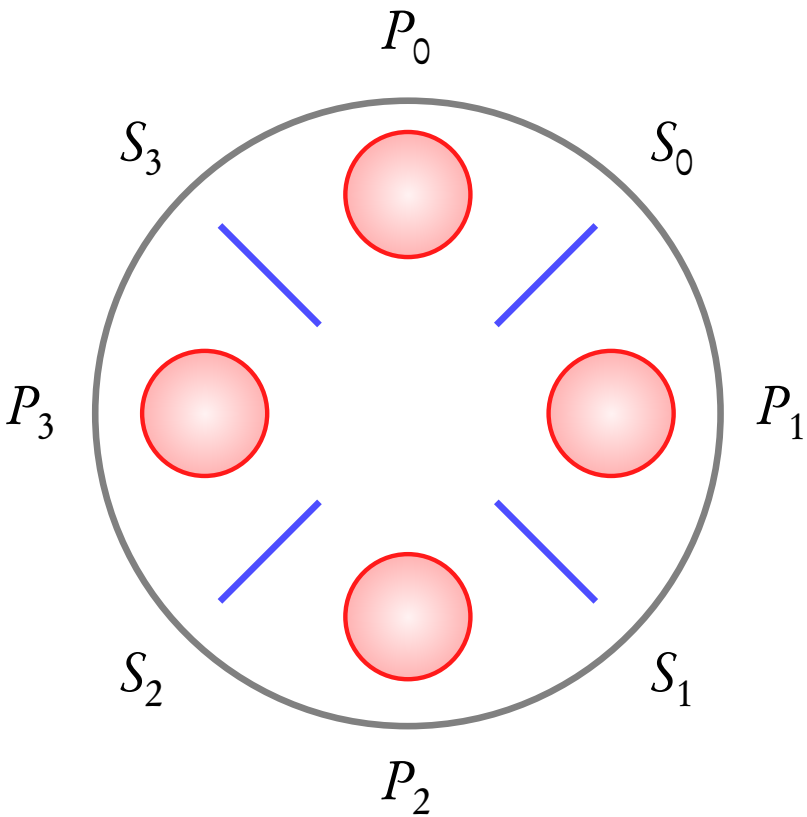
Sticks

0	2	2	0

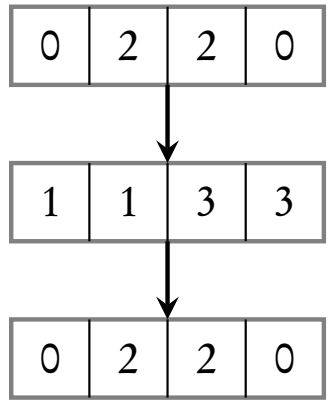


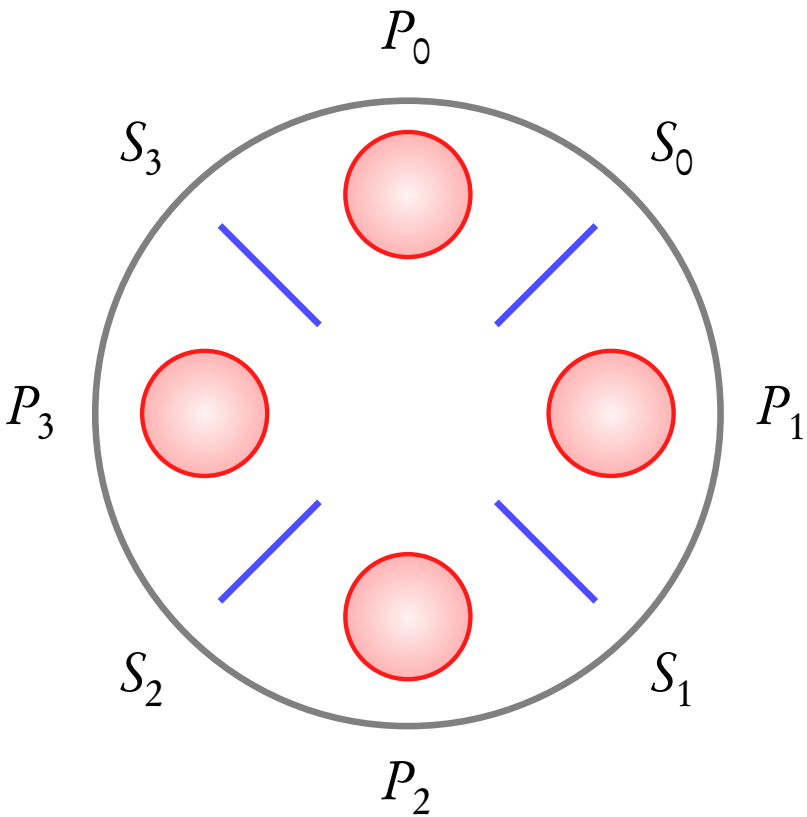
Sticks



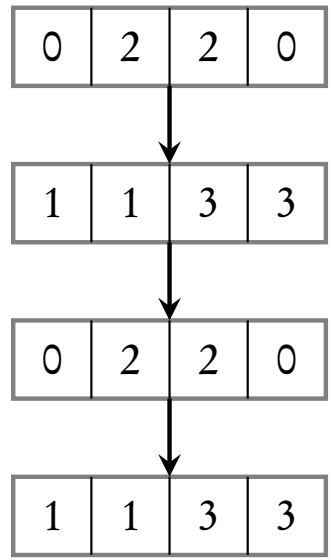


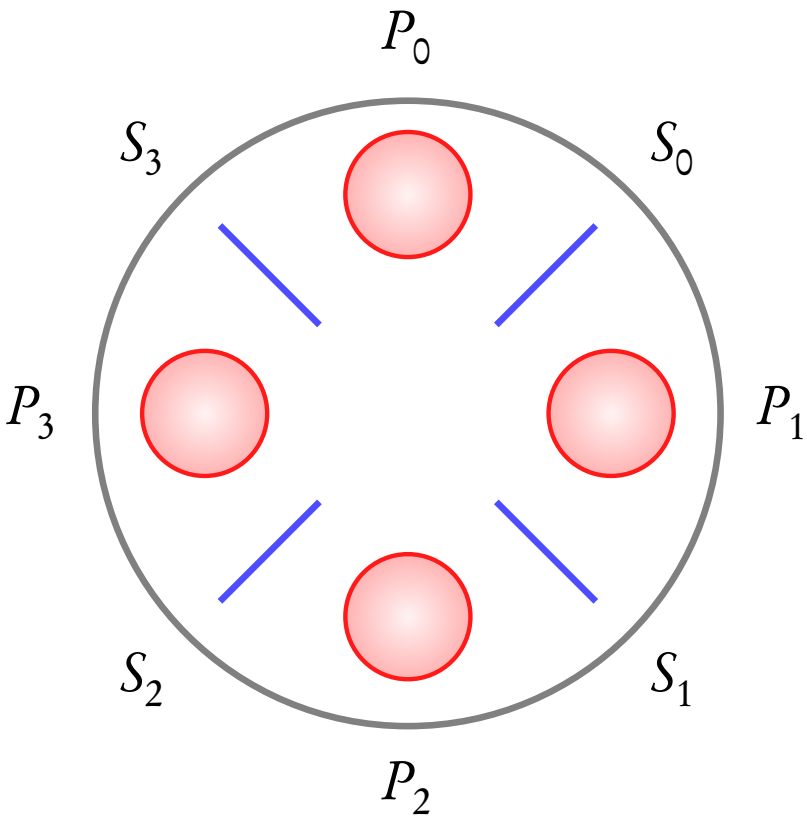
Sticks



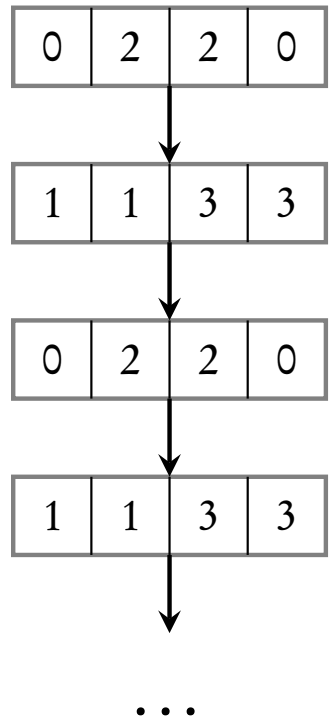


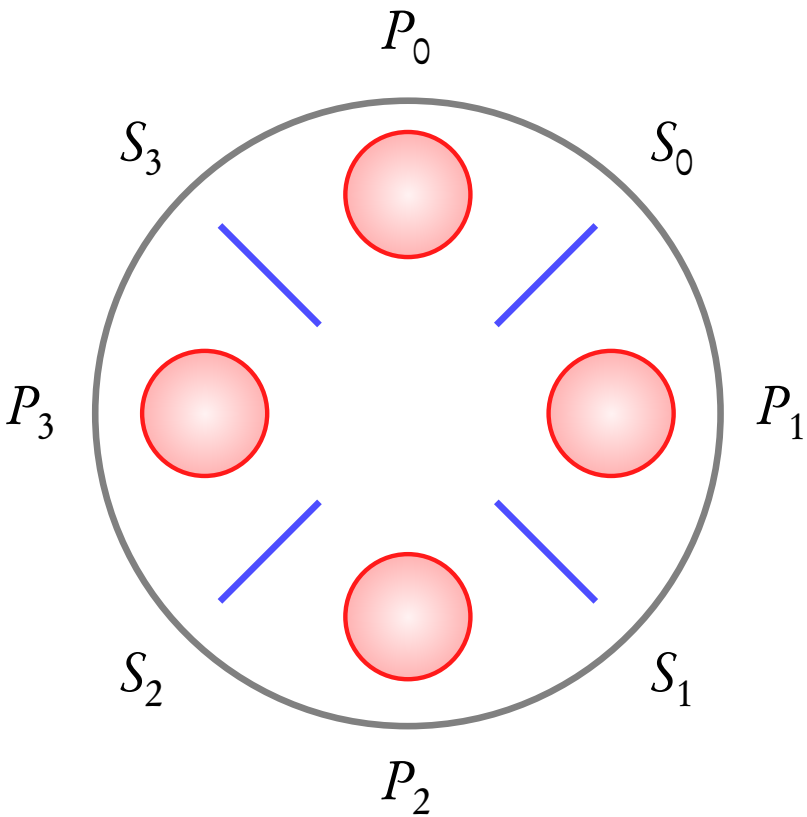
Sticks





Sticks





Sticks

0	2	2	0
1	1	3	3
0	2	2	0
1	1	3	3

...

This solution is deadlock-free

Liveness properties

Good happens **infinitely often**

`FAIRNESS running`