

CS766

Analysis of Concurrent Programs

09 Feb 2021

Mutual exclusion protocols

Szymanski Algorithm

A (class) presentation by

Karnika Shivhare

IITB 204050010

SZYMANSKI ALGORITHM

Algorithm for mutual exclusion

```
or object to mirror  
mirror_mod.mirror_object  
operation == "MIRROR_X":  
mirror_mod.use_x = True  
mirror_mod.use_y = False  
mirror_mod.use_z = False  
operation == "MIRROR_Y":  
mirror_mod.use_x = False  
mirror_mod.use_y = True  
mirror_mod.use_z = False  
operation == "MIRROR_Z":  
mirror_mod.use_x = False  
mirror_mod.use_y = False  
mirror_mod.use_z = True  
  
selection at the end -add  
mirror_ob.select= 1  
mirror_ob.select=1  
context.scene.objects.active  
("Selected" + str(modifier  
mirror_ob.select = 0  
bpy.context.selected_obj  
data.objects[one.name].select  
  
print("please select exactly  
  
-- OPERATOR CLASSES ----  
  
types.Operator):  
X mirror to the selected  
object.mirror_mirror_x"  
mirror X"
```

SZYMANSKI ALGORITHM

Drawbacks of previous discussed algorithms:

PETERSON

$O(n^2)$ pre-protocol

BAKERY

Use unbounded ticket numbers

DEKKER

Don't scale well beyond 2 processes

Out of scope of the 15 minutes.





Ashutosh Gupta 1/22 1:55 PM Edited



Presentations and assignments

Lecture 8 on 9th **Feb** will be a lecture of presentations. You will be presenting 15-20 minutes each. This will be individual presentations, since topics are relatively straight forward. I need six volunteers. If I will not get them by the weekend, I will pick. Others will be making presentations on different topics or may choose to do assignments. If you prefer to present, please come forward as soon as possible. If you **do not want any presentation** please also post here.

Presentations and assignments

Critical Section = Presentation



Ashutosh Gupta 1/22 1:55 PM Edited



Presentations and assignments

Lecture 8 on 9th **Feb** will be a lecture of presentations. You will be presenting 15-20 minutes each. This will be individual presentations, since topics are relatively straight forward. I need six volunteers. If I will not get them by the weekend, I will pick. Others will be making presentations on different topics or may choose to do assignments. If you prefer to present, please come forward as soon as possible. If you **do not want any presentation** please also post here.

today



Ashutosh Gupta 1/22 1:55 PM Edited



Presentations and assignments

Lecture 8 on 9th **Feb** will be a lecture of presentations. You will be presenting 15-20 minutes each. This will be individual presentations, since topics are relatively straight forward. I need six volunteers. If I will not get them by the weekend, I will pick. Others will be making presentations on different topics or may choose to do assignments. If you prefer to present, please come forward as soon as possible. If you **do not want any presentation** please also post here.

$\text{Flag}[\text{self}] = 1$



Darshana Ajit Nafde 1/22 4:54 PM
I volunteer for the presentation

today



Ashutosh Gupta 1/22 1:55 PM Edited



Presentations and assignments

Lecture 8 on 9th **Feb** will be a lecture of presentations. You will be presenting 15-20 minutes each. This will be individual presentations, since topics are relatively straight forward. I need six volunteers. If I will not get them by the weekend, I will pick. Others will be making presentations on different topics or may choose to do assignments. If you prefer to present, please come forward as soon as possible. If you **do not want any presentation** please also post here.

Everyone raised their flags
= 1 who wished to enter
critical section at around
same time.



Kulkarni Shantanu Alias Chaitany Shashikant 1/22 5:28 PM

I also volunteer for presentation



Karnika Shivhare 1/22 5:29 PM

I volunteer for presentation



TUSHAR DHAWAL BARANWAL 1/22 5:29 PM

I volunteer for the presentation

today



Ashutosh Gupta 1/22 1:55 PM Edited



Presentations and assignments

Lecture 8 on 9th **Feb** will be a lecture of presentations. You will be presenting 15-20 minutes each. This will be individual presentations, since topics are relatively straight forward. I need six volunteers. If I will not get them by the weekend, I will pick. Others will be making presentations on different topics or may choose to do assignments. If you prefer to present, please come forward as soon as possible. If you **do not want any presentation** please also post here.



Darshana Ajit Nafde 1/22 4:54 PM
I volunteer for the presentation



Kulkarni Shantanu Alias Chaitany Shashikant 1/22 5:28 PM
I also volunteer for presentation



Karnika Shivhare 1/22 5:29 PM
I volunteer for presentation



TUSHAR DHAWAL BARANWAL 1/22 5:29 PM
I volunteer for the presentation

today



Ashutosh Gupta 1/22 1:55 PM Edited

Presentations and assignments

Lecture 8 on 9th Feb will be a lecture of presentations. You will be presenting 15-20 minutes each. This will be individual presentations, since topics are relatively straight forward. I need six volunteers. If I will not get them by the weekend, I will pick. Others will be making presentations on different topics or may choose to do assignments. If you prefer to present, please come forward as soon as possible. If you **do not want any presentation** please also post here.



Ashutosh Gupta 1/22 7:20 PM
I have six volunteers now.



Pooja Verma 1/22 8:02 PM
I m interested too for presentation



Ashutosh Gupta 1/22 8:03 PM
The first six presentations are already scheduled. Please look at the course page.



Darshana Ajit Nafde 1/22 4:54 PM
I volunteer for the presentation



Kulkarni Shantanu Alias Chaitany Shashikant 1/22 5:28 PM
I also volunteer for presentation



Karnika Shivhare 1/22 5:29 PM
I volunteer for presentation



TUSHAR DHAWAL BARANWAL 1/22 5:29 PM
I volunteer for the presentation

today



Ashutosh Gupta 1/22 1:55 PM Edited

Presentations and assignments

Lecture 8 on 9th Feb will be a lecture of presentations. You will be presenting 15-20 minutes each. This will be individual presentations, since topics are relatively straight forward. I need six volunteers. If I will not get them by the weekend, I will pick. Others will be making presentations on different topics or may choose to do assignments. **do not want any presentation** please also post here.



it Nafde 1/22 4:54 PM
for the presentation

ntanu Alias Chaitany Shashikant 1/22 5:28 PM
nteer for presentation

hare 1/22 5:29 PM
for presentation

AWAL BARANWAL 1/22 5:29 PM
for the presentation



Ashutosh Gupta 1/22 7:20 PM
I have six volunteers now.



Pooja Verma 1/22 8:02 PM
I m interested too for presentation



Ashutosh Gupta 1/22 8:03 PM
The first six presentations are already scheduled. Please look at the course page.

How Essential is a PhD?

- PhD is not absolutely essential to succeed in the technical world
 - Do not opt for a PhD at least in the US unless you are really serious about research
 - In the US, PhDs are not time-bound
 - You can always opt for a PhD after a few years of experience in the industry
- Advantages of PhD
 - can show your progression in the mainstream technical industry
 - make certain jobs inaccessible (e.g., academic and senior industrial research positions)
 - need not reduce your remuneration
- How does a computer scientist research?
 - Problem formulation tends to be a lot more sound, in depth and breadth
 - A PhD provides the experience to go after big problems and much better judgement for what can or cannot work

How Essential is a PhD?

- PhD is not absolutely essential to succeed in the technical world
 - Do not opt for a PhD at least in the US unless you are really serious about research
 - In the US, PhDs are not time-bound
 - You can always opt for a PhD after a few years of experience in the industry
- Advantages of PhD
 - can show your progression in the mainstream technical industry
 - make certain jobs inaccessible (e.g., academic and senior industrial research positions)
 - need not reduce your remuneration
- How does a computer scientist research?
 - Problem formulation tends to be a lot more sound, in depth and breadth
 - A PhD provides the experience to go after big problems and much better judgement for what can or cannot work

In Critical Section

SZYMANSKI ALGORITHM

Idea

The prologue is modeled after a waiting room with two doors.

SZYMANSKI ALGORITHM

Idea

The prologue is modeled after a waiting room with two doors.

All processes requesting entry to the CS at roughly the same time gather first in the waiting room.

SZYMANSKI ALGORITHM

Idea

The prologue is modeled after a waiting room with two doors – entry and exit.

All processes requesting entry to the CS at roughly the same time gather first in the waiting room.

Last of them closes the entry door and opens the exit door.

SZYMANSKI ALGORITHM

Idea

The prologue is modeled after a waiting room with two doors.

All processes requesting entry to the CS at roughly the same time gather first in the waiting room.

Last of them closes the entry door and opens the exit door.

From there, one by one, they enter their CS.

SZYMANSKI ALGORITHM

Idea

The prologue is modeled after a waiting room with two doors.

All processes requesting entry to the CS at roughly the same time gather first in the waiting room.

Last of them closes the entry door and opens the exit door.

From there, one by one, they enter their CS.

The last process to leave the CS, closes the exit door and reopens the entry door, so that the next batch of processes may enter.

SZYMANSKI ALGORITHM

Idea

The prologue is modeled after a waiting room with two doors.

All processes requesting entry to the CS at roughly the same time gather first in the waiting room.

Then, when there are no more processes requesting entry, waiting processes move to the end of the prologue.

From there, one by one, they enter their CS.

Any other process requesting entry to its CS at that time has to wait in the initial part of the prologue (before the waiting room).

SHARED VARIABLES

Each process exclusively writes a variable flag, which is read by all the other processes.

It assumes one of five values:

- 0 Executing non-CS
- 1 i wants to enter the CS
- 2 i waits for other processes to enter waiting room
- 3 i just entered the waiting room
- 4 i is leaving the waiting room. entry door is closed and exit door is open.

SZYMANSKI ALGORITHM

ENTRY PROTOCOL

```
flag[self] ← 1
await( all flag[1..N] ∈ {0, 1, 2})
flag [self] ← 3
if any flag[1..N] = 1:
    flag[self] ← 2
    await( any flag[1..N] = 4)
flag[self] ← 4
await( flag[1..self-1] ∈ {0, 1})
```

EXIT PROTOCOL

```
await( all flag [self+1..N] ∈ {0, 1, 4})
flag[self] ← 0
```

Critical Section

ENTRY PROTOCOL process 1

flag[self = 1] $\leftarrow$ 1

await(flag[2] $\in$ {0, 1, 2})

flag [self = 1] $\leftarrow$ 3

if flag[2] = 1:

 flag[self = 1] $\leftarrow$ 2

 await(if flag[2] = 4)

flag[self = 1] $\leftarrow$ 4

//await(flag[1..self-1] $\in$ {0, 1})

process 1 in Critical Section

EXIT PROTOCOL

await(flag[2] $\in$ {0, 1, 4})

flag[self = 1] $\leftarrow$ 0

ENTRY PROTOCOL process 2

flag[self = 2] $\leftarrow$ 1

await(flag[1] $\in$ {0, 1, 2})

flag [self = 2] $\leftarrow$ 3

if flag[1] = 1:

 flag[self = 2] $\leftarrow$ 2

 await(flag[1] = 4)

flag[self = 2] $\leftarrow$ 4

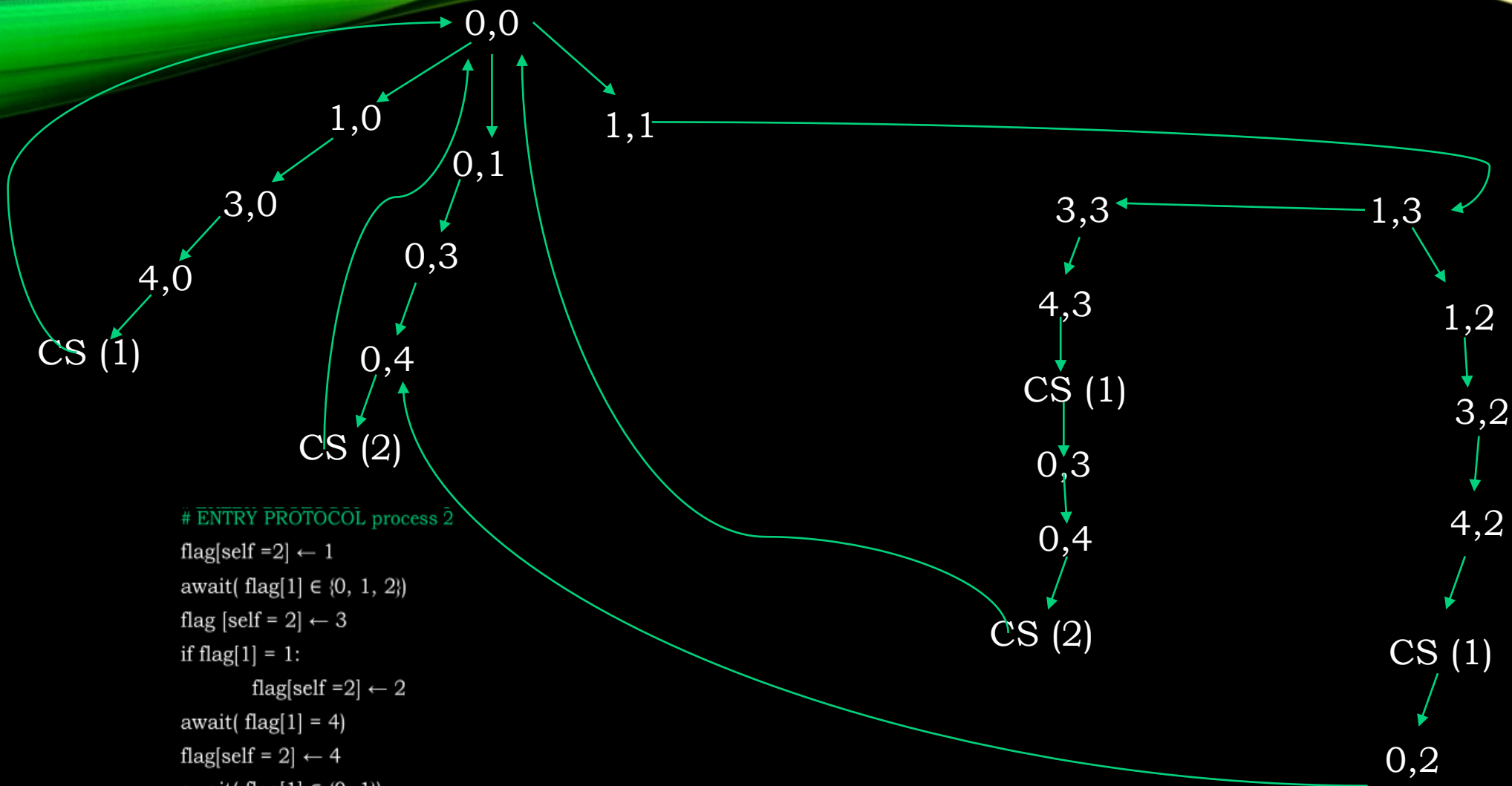
await(flag[1] $\in$ {0, 1})

Process 2 in Critical Section

EXIT PROTOCOL

await(flag[1] $\in$ {0, 1, 4})

flag[self = 2] $\leftarrow$ 0



ENTRY PROTOCOL process 1

```

flag[self =1] ← 1
await( flag[2] ∈ {0, 1, 2})
flag [self = 1] ← 3
if flag[2] = 1:
    flag[self =1] ← 2
await(if flag[2] = 4)
flag[self = 1] ← 4
//await( flag[1..self-1] ∈ {0, 1})

```

process 1 in Critical Section

EXIT PROTOCOL

```

await( flag[2] ∈ {0, 1, 4})
flag[self = 1] ← 0

```

ENTRY PROTOCOL process 2

```

flag[self =2] ← 1
await( flag[1] ∈ {0, 1, 2})
flag [self = 2] ← 3
if flag[1] = 1:
    flag[self =2] ← 2
await( flag[1] = 4)
flag[self = 2] ← 4
await( flag[1] ∈ {0, 1})

```

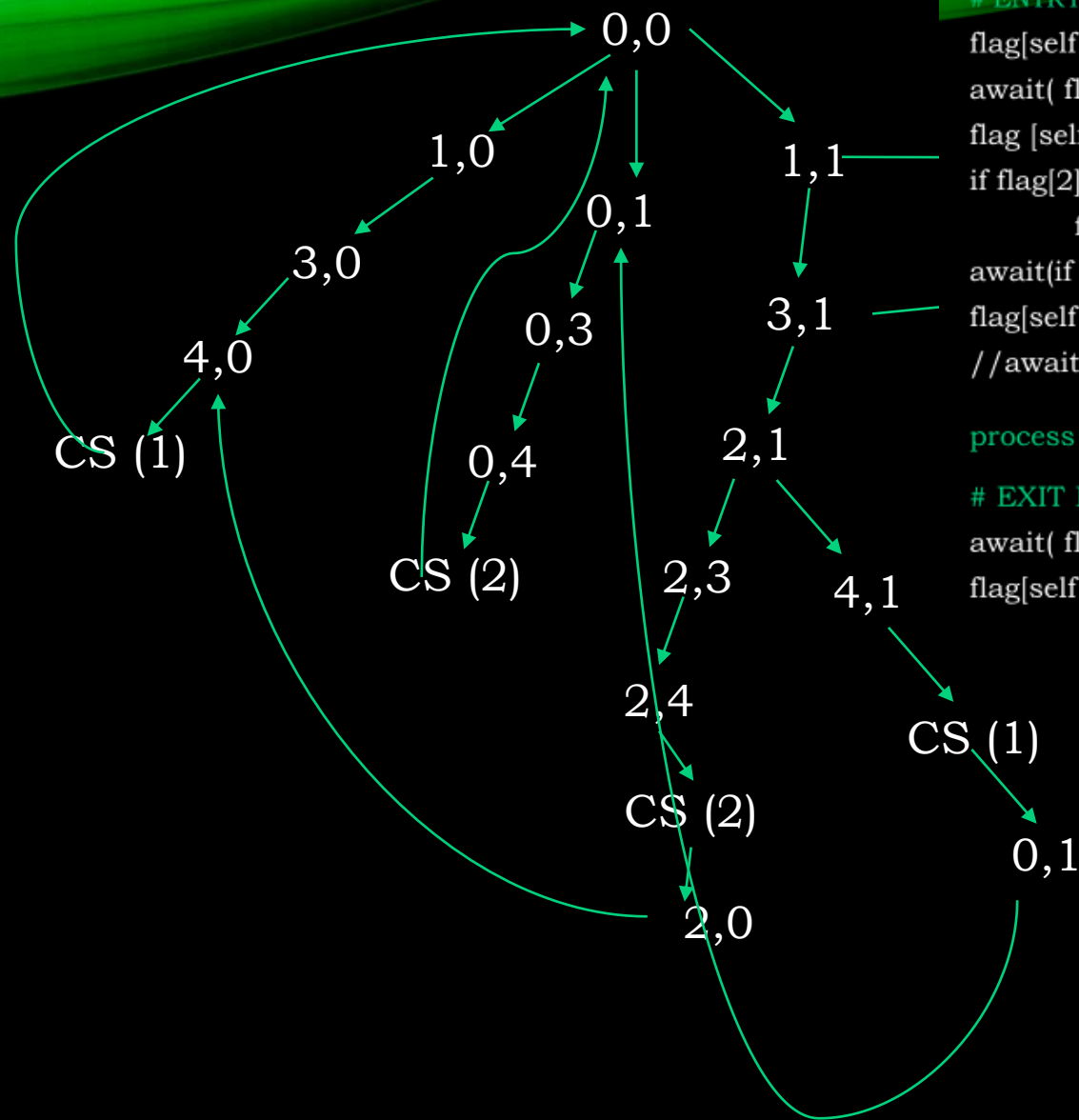
Process 2 in Critical Section

EXIT PROTOCOL

```

await( flag[1] ∈ {0, 1, 4})
flag[self = 2] ← 0

```



ENTRY PROTOCOL process 1

```

flag[self = 1] ← 1
await( flag[2] ∈ {0, 1, 2})
flag [self = 1] ← 3
if flag[2] = 1:
    flag[self = 1] ← 2
await(if flag[2] = 4)
flag[self = 1] ← 4
//await( flag[1..self-1] ∈ {0, 1})
  
```

process 1 in Critical Section

EXIT PROTOCOL

```

await( flag[2] ∈ {0, 1, 4})
flag[self = 1] ← 0
  
```

ENTRY PROTOCOL process 2

```

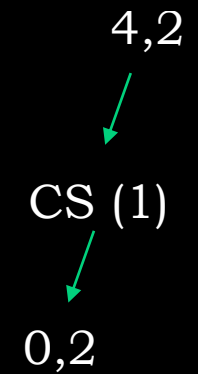
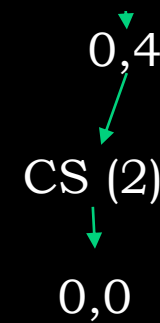
flag[self = 2] ← 1
await( flag[1] ∈ {0, 1, 2})
flag [self = 2] ← 3
if flag[1] = 1:
    flag[self = 2] ← 2
await( flag[1] = 4)
flag[self = 2] ← 4
await( flag[1] ∈ {0, 1})
  
```

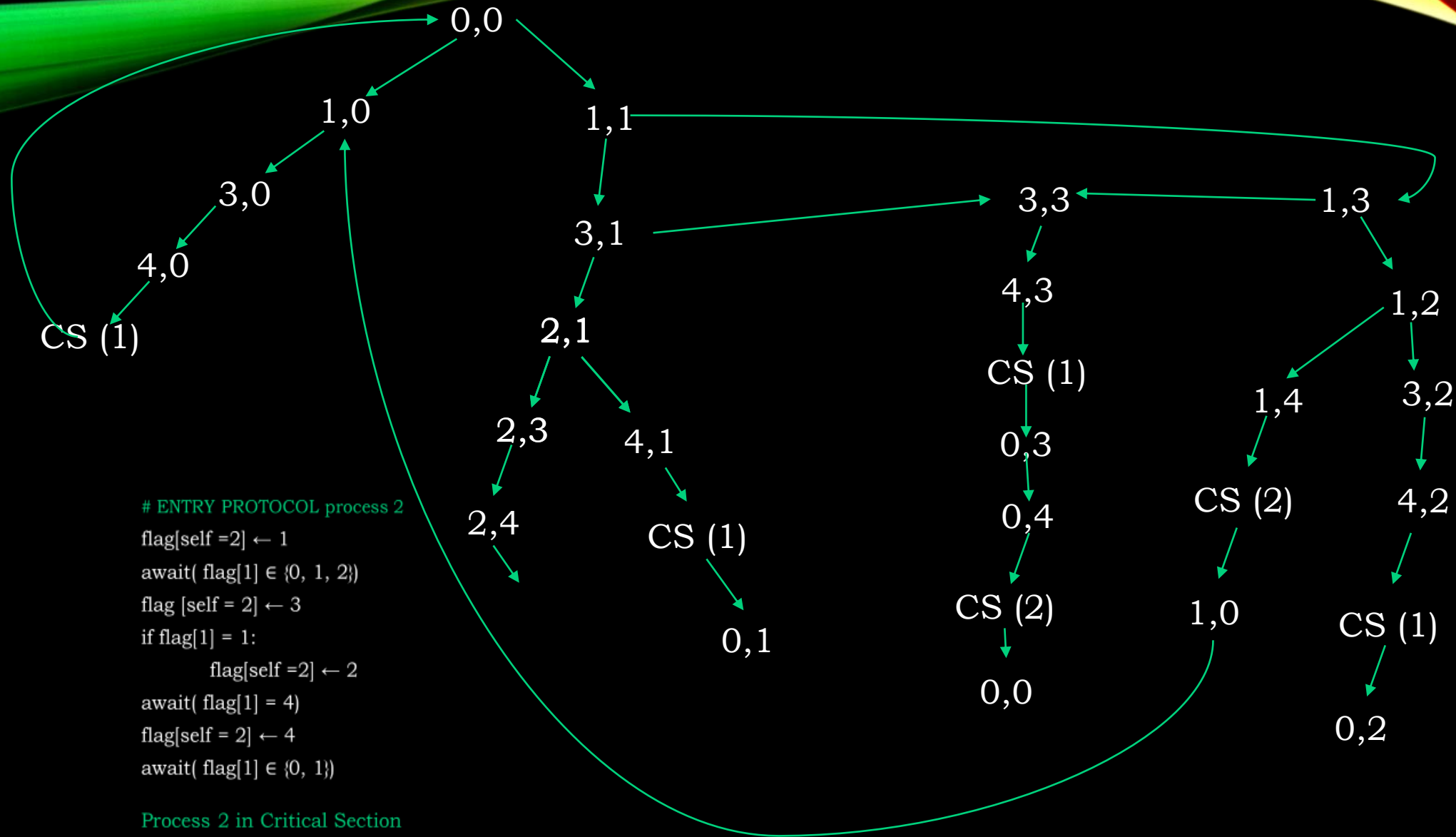
Process 2 in Critical Section

EXIT PROTOCOL

```

await( flag[1] ∈ {0, 1, 4})
flag[self = 2] ← 0
  
```





ENTRY PROTOCOL process 1

```

flag[self = 1] ← 1
await( flag[2] ∈ {0, 1, 2})
flag [self = 1] ← 3
if flag[2] = 1:
    flag[self = 1] ← 2
await(if flag[2] = 4)
flag[self = 1] ← 4
//await( flag[1..self-1] ∈ {0, 1})

```

process 1 in Critical Section

EXIT PROTOCOL

```

await( flag[2] ∈ {0, 1, 4})
flag[self = 1] ← 0

```

ENTRY PROTOCOL process 2

```

flag[self = 2] ← 1
await( flag[1] ∈ {0, 1, 2})
flag [self = 2] ← 3
if flag[1] = 1:
    flag[self = 2] ← 2
await( flag[1] = 4)
flag[self = 2] ← 4
await( flag[1] ∈ {0, 1})

```

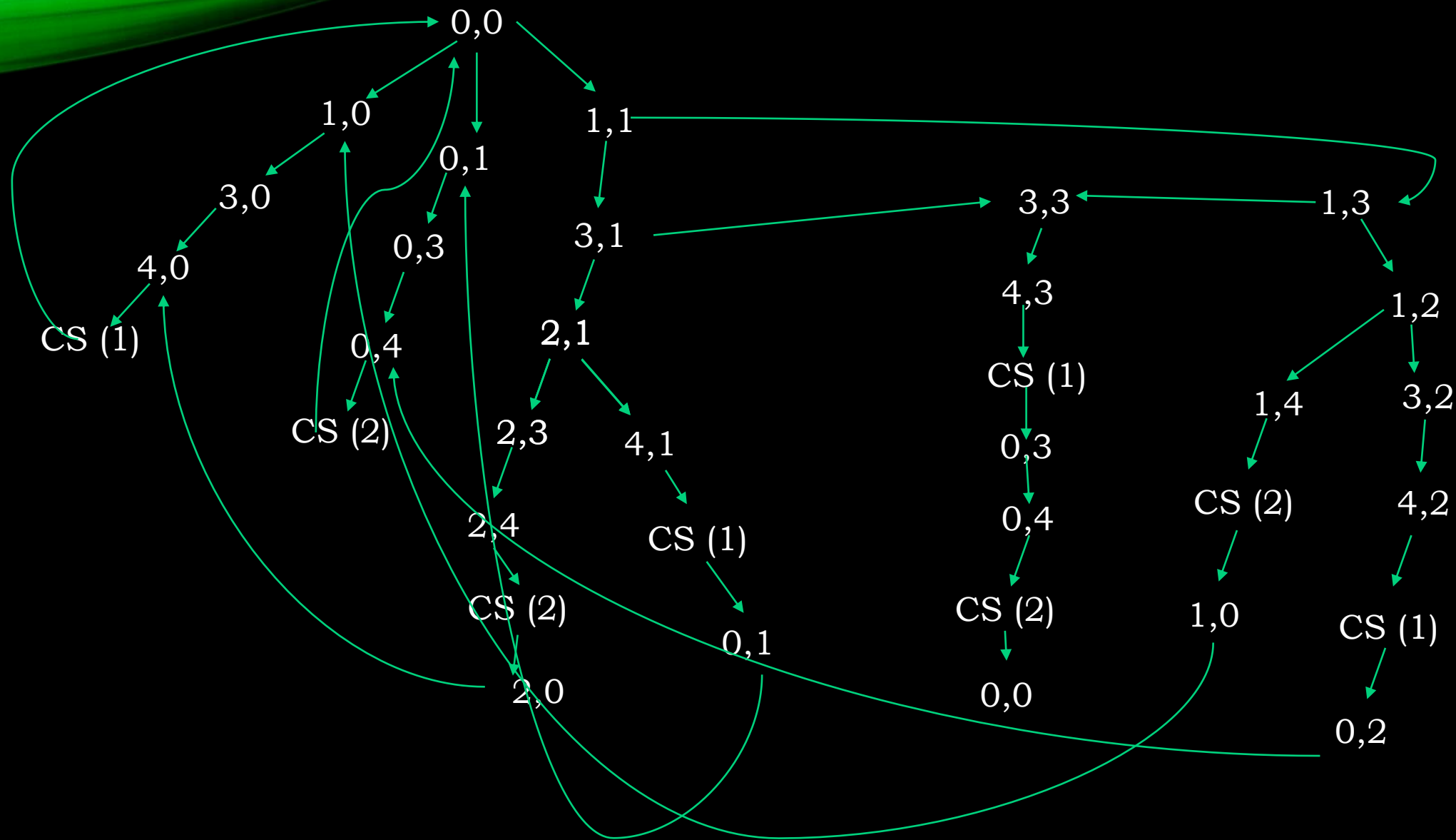
Process 2 in Critical Section

EXIT PROTOCOL

```

await( flag[1] ∈ {0, 1, 4})
flag[self = 2] ← 0

```



ENTRY PROTOCOL process 1

flag[self = 1] $\leftarrow$ 1

await(flag[2] $\in$ {0, 1, 2})

flag [self = 1] $\leftarrow$ 3

if flag[2] = 1:

 flag[self = 1] $\leftarrow$ 2

 await(if flag[2] = 4)

flag[self = 1] $\leftarrow$ 4

//await(flag[1..self-1] $\in$ {0, 1})

process 1 in Critical Section

EXIT PROTOCOL

await(flag[2] $\in$ {0, 1, 4})

flag[self = 1] $\leftarrow$ 0

ENTRY PROTOCOL process 2

flag[self = 2] $\leftarrow$ 1

await(flag[1] $\in$ {0, 1, 2})

flag [self = 2] $\leftarrow$ 3

if flag[1] = 1:

 flag[self = 2] $\leftarrow$ 2

 await(flag[1] = 4)

flag[self = 2] $\leftarrow$ 4

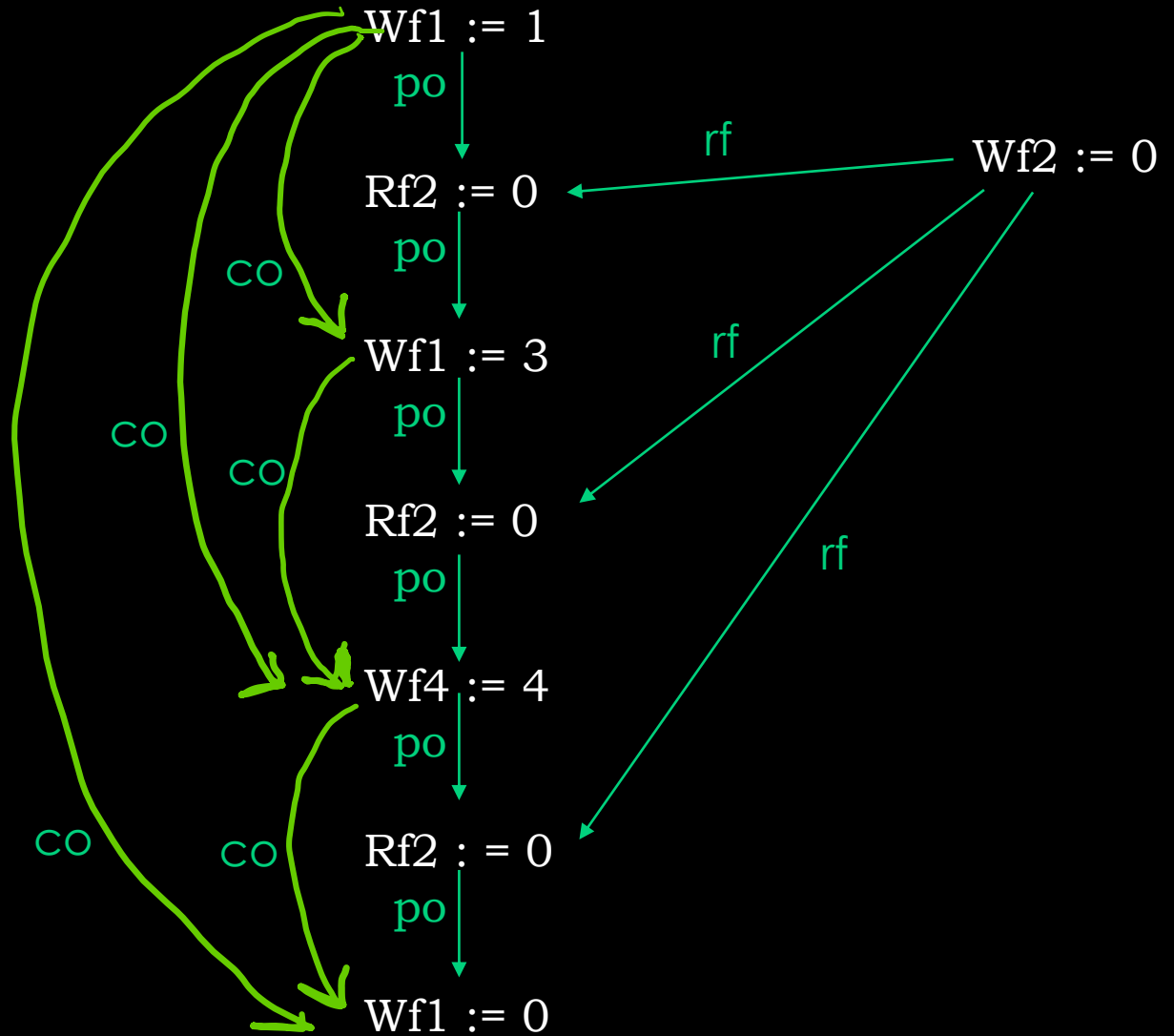
await(flag[1] $\in$ {0, 1})

Process 2 in Critical Section

EXIT PROTOCOL

await(flag[1] $\in$ {0, 1, 4})

flag[self = 2] $\leftarrow$ 0



ENTRY PROTOCOL process 1

```
flag[self = 1] ← 1
await( flag[2] ∈ {0, 1, 2})
flag [self = 1] ← 3
if flag[2] = 1:
    flag[self = 1] ← 2
    await(if flag[2] = 4)
flag[self = 1] ← 4
//await( flag[1..self-1] ∈ {0, 1})
```

process 1 in Critical Section

EXIT PROTOCOL

```
await( flag[2] ∈ {0, 1, 4})
flag[self = 1] ← 0
```

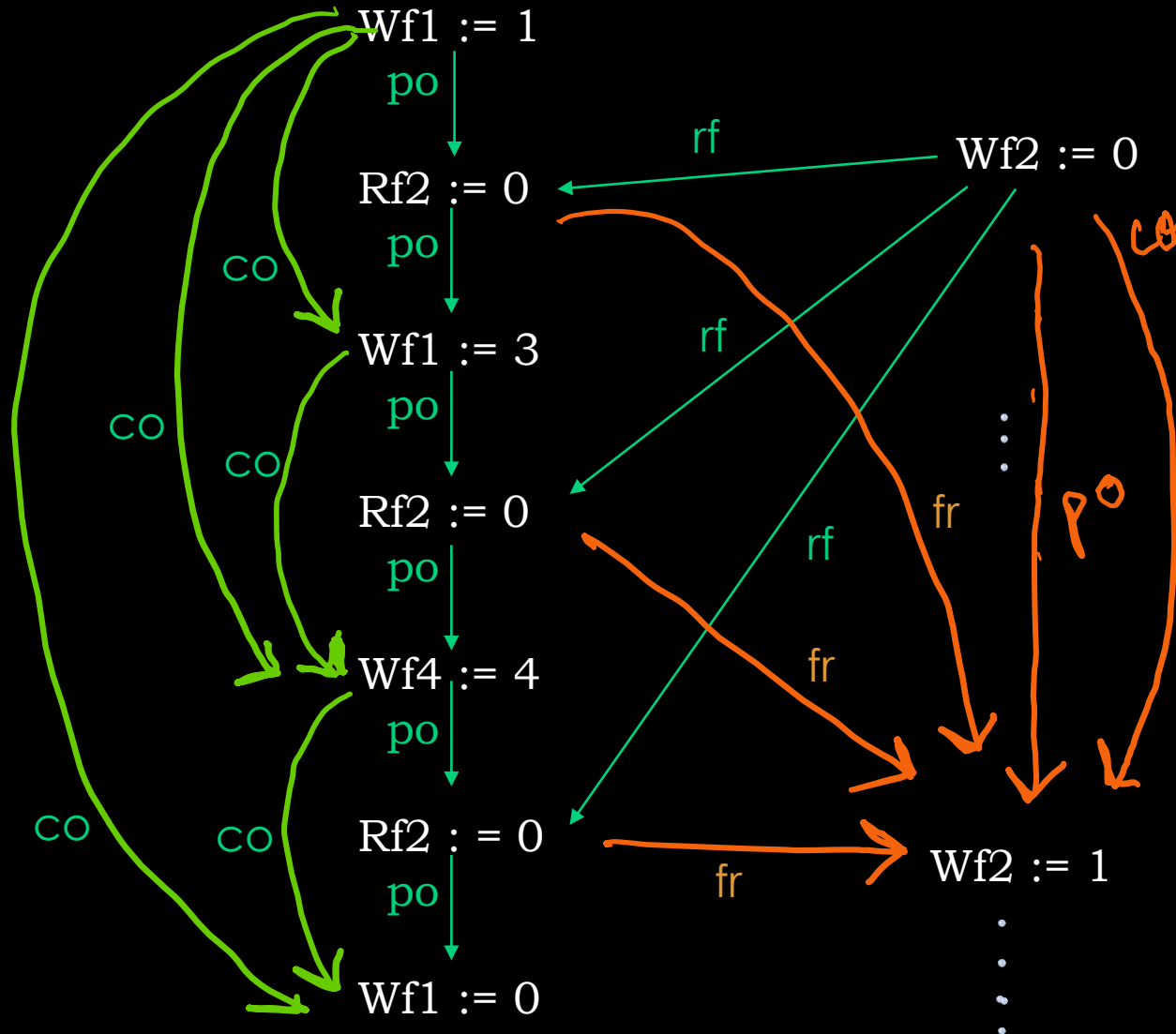
ENTRY PROTOCOL process 2

```
flag[self = 2] ← 1
await( flag[1] ∈ {0, 1, 2})
flag [self = 2] ← 3
if flag[1] = 1:
    flag[self = 2] ← 2
    await( flag[1] = 4)
flag[self = 2] ← 4
await( flag[1] ∈ {0, 1})
```

Process 2 in Critical Section

EXIT PROTOCOL

```
await( flag[1] ∈ {0, 1, 4})
flag[self = 2] ← 0
```



- order of the "all" and "any" tests must be uniform.
- Also the "any" tests should be satisfied by a thread other than self.

For example,

if the test is $\text{any flag}[1..N] = 1$ and only $\text{flag}[\text{self}] = 1$, then the test is said to have failed/returned 0

-
- # ENTRY PROTOCOL process 1**
- ```

flag[self] := 1
await(flag[2] ∈ {0, 1, 2})
flag[self] := 3
if flag[2] = 1:
 flag[self] := 2
 await(if flag[2] = 4)
flag[self] := 1
// await(flag[1].self-1 ∈ {0, 1})

```
- process 1 in Critical Section**
- # EXIT PROTOCOL**
- ```

await( flag[2] ∈ {0, 1, 4})
flag[self] := 1

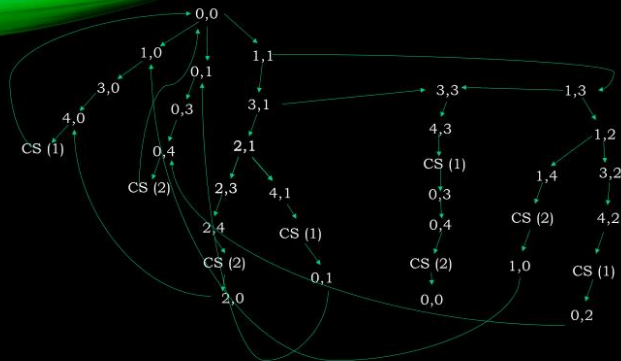
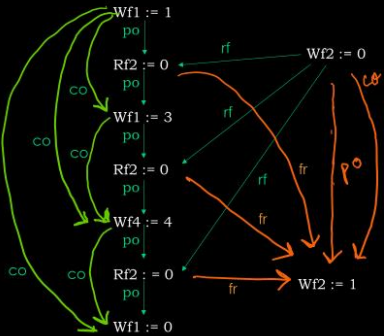
```
- # ENTRY PROTOCOL process 2**
- ```

flag[self] := 1
await(flag[1] ∈ {0, 1, 2})
flag[self] := 2
if flag[1] = 1:
 flag[self] := 2
 await(flag[1] = 4)
flag[self] := 2
// await(flag[1] ∈ {0, 1})

```
- Process 2 in Critical Section**
- # EXIT PROTOCOL**
- ```

await( flag[1] ∈ {0, 1, 4})
flag[self] := 2

```
- The diagram illustrates the execution of two processes, Process 1 and Process 2, in a critical section. The variables Wf1, Wf2, Rf1, and Rf2 represent the state of the processes. The diagram shows that Process 1 can execute its critical section multiple times, while Process 2 can only execute its critical section once. This is because Process 1's critical section is protected by a semaphore (Wf1) that is initialized to 1, and Process 2's critical section is protected by a semaphore (Wf2) that is initialized to 0. The diagram also shows that Process 1 can execute its critical section multiple times, while Process 2 can only execute its critical section once. This is because Process 1's critical section is protected by a semaphore (Wf1) that is initialized to 1, and Process 2's critical section is protected by a semaphore (Wf2) that is initialized to 0.



SZYMANSKI ALGORITHM

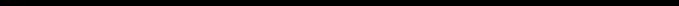
```

flag[self] ← 1
await( all flag[1..N] ∈ {0, 1, 2})
flag[self] ← 3
if any flag[1..N] = 1:
    flag[self] ← 2
    await( any flag[1..N] = 4)
flag[self] ← 4
await( flag[1..self-1] ∈ {0, 1})

```

```
await( all flag [self+1..N]  $\in$  {0, 1, 4})
    flag[self]  $\leftarrow$  0
```

```
flag[self] ← 4
await( flag[1..self-1] ∈ {0, 1} )
```

- 

Thank You

1.) “A simple solution to Lamport's concurrent programming problem with linear wait”

https://www.researchgate.net/publication/221235887_A_simple_solution_to_Lamport's_concurrent_programming_problem_with_linear_wait

2.) https://en.wikipedia.org/wiki/Szyma%C5%84ski%27s_algorithm

3.) CS766 <https://www.cse.iitb.ac.in/~akg/courses/2021-concurrency/>

4.) <https://www.sarc-iitb.org/events/core-talks/>

References

1.) “A simple solution to Lamport's concurrent programming problem with linear wait”

https://www.mimuw.edu.pl/~sl/teaching/13_14/PWiR/zadanie1/szymanski.88.pdf

2.) https://en.wikipedia.org/wiki/Szyma%C5%84ski%27s_algorithm

3.) The Art of Multiprocessor Programming by Nir Shavit

4.) <https://www.cse.iitb.ac.in/~akg/courses/2021-concurrency/lec-concurrency-how-to-think.pdf>

5.) <https://www.youtube.com/playlist?list=PLssOvQUpC9clEGbrWDTbroK63tzzx0Lwcu>

6.) <https://link.springer.com/content/pdf/10.1007%2FBFb0054187.pdf>