Analysing Mutual Exclusion using Process Algebra with Signals

Victor Dyseryn, Rob van Glabbeek & Peter Höfner

Ecole Polytechnique, Paris, France

Data61, CSIRO, Sydney, Australia

University of New South Wales, Sydney, Australia

September 2017

Overview

Do correct mutual exclusion protocols exists?

Overview

Do correct mutual exclusion protocols exists?

Can a correct mutual exclusion protocol be modelled in standard process algebras like CCS?

Overview

Do correct mutual exclusion protocols exists?

Can a correct mutual exclusion protocol be modelled in standard process algebras like CCS?

Which minimal extension of CCS do we need?

Overview

What makes a mutual exclusion protocol correct?

Do correct mutual exclusion protocols exists?

Can a correct mutual exclusion protocol be modelled in standard process algebras like CCS?

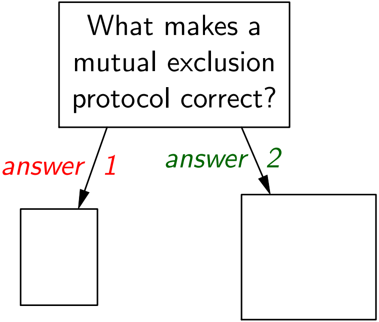
Which minimal extension of CCS do we need?

Disclaimer

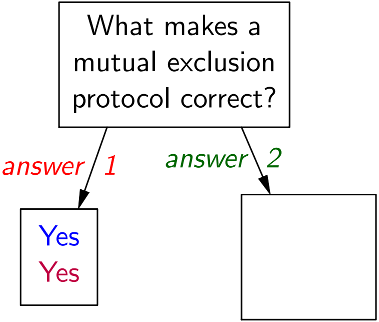
This talk is about concurrent systems.

We may assume nothing about the relative speed of parallel components.

Do correct mutual exclusion protocols exists?
and can they be modelled in CCS?

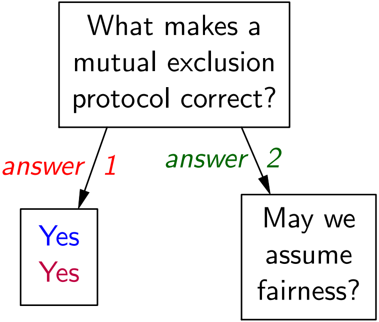


Do correct mutual exclusion protocols exists?
and can they be modelled in CCS?



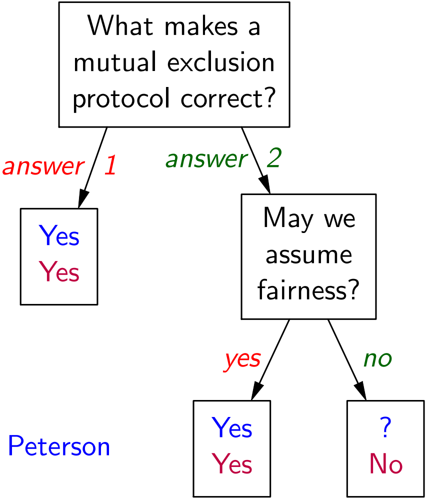
Peterson

Do correct mutual exclusion protocols exists?
and can they be modelled in CCS?



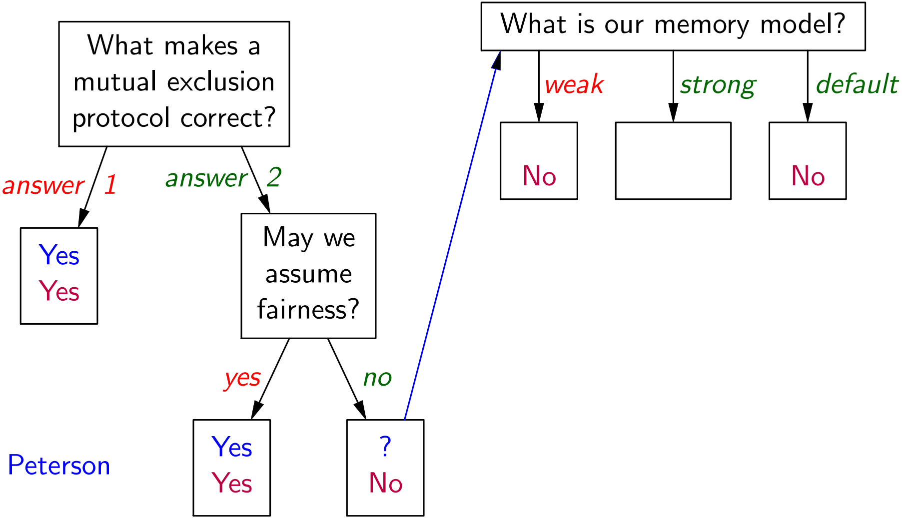
Peterson

Do correct mutual exclusion protocols exists?
and can they be modelled in CCS?

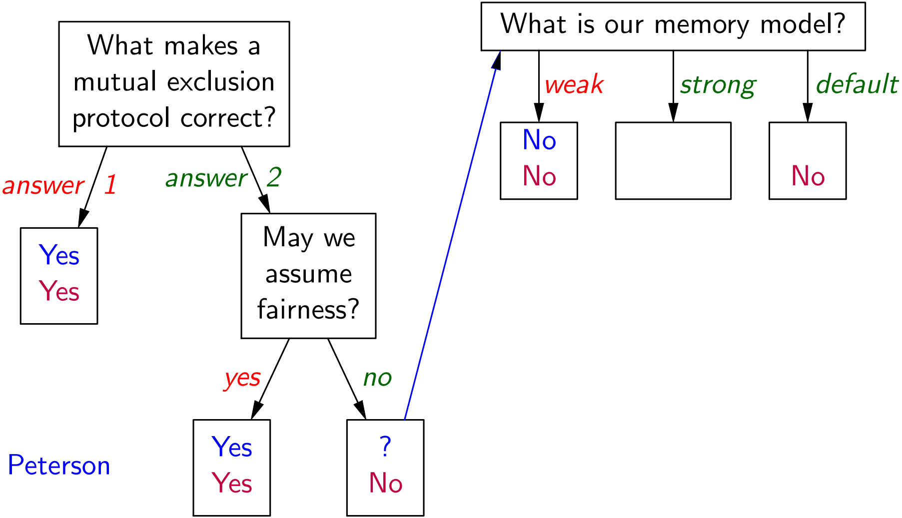


[Van Glabbeek & Hoefner 2015]

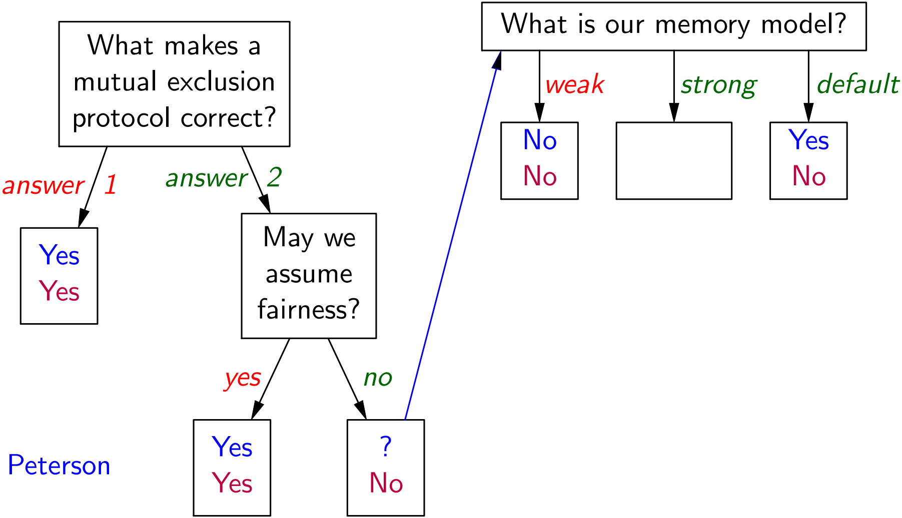
Do correct mutual exclusion protocols exists?
and can they be modelled in CCS?



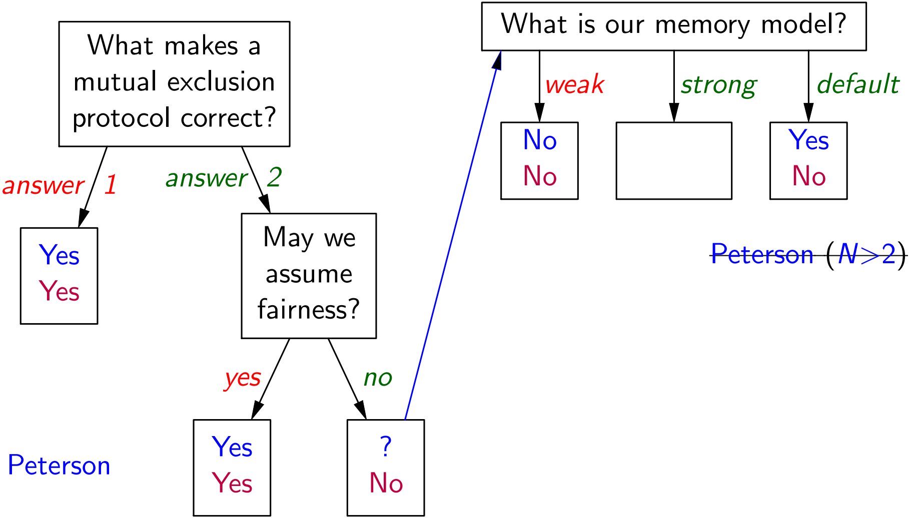
Do correct mutual exclusion protocols exists?
and can they be modelled in CCS?



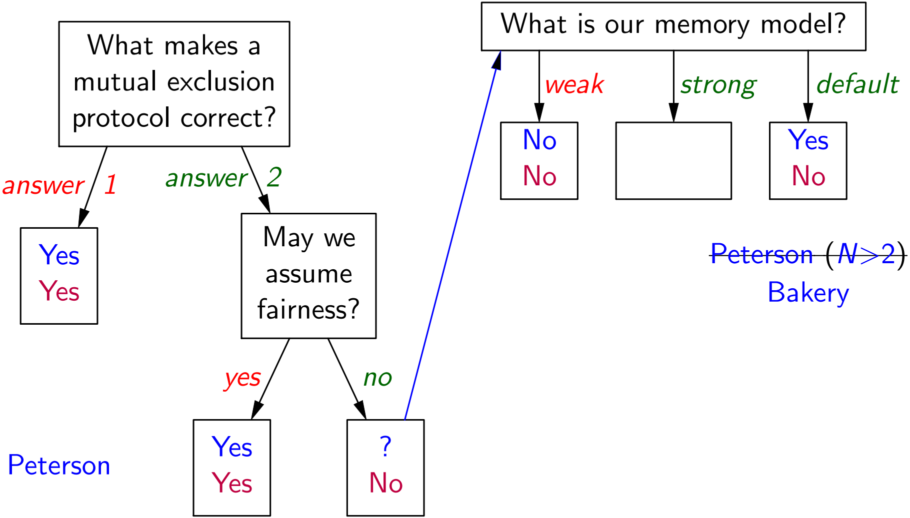
Do correct mutual exclusion protocols exists?
and can they be modelled in CCS?



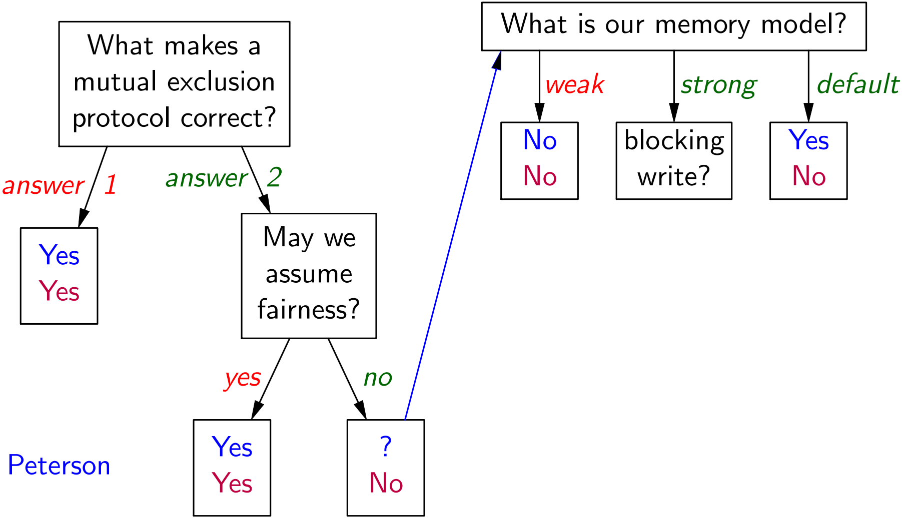
Do correct mutual exclusion protocols exists?
and can they be modelled in CCS?



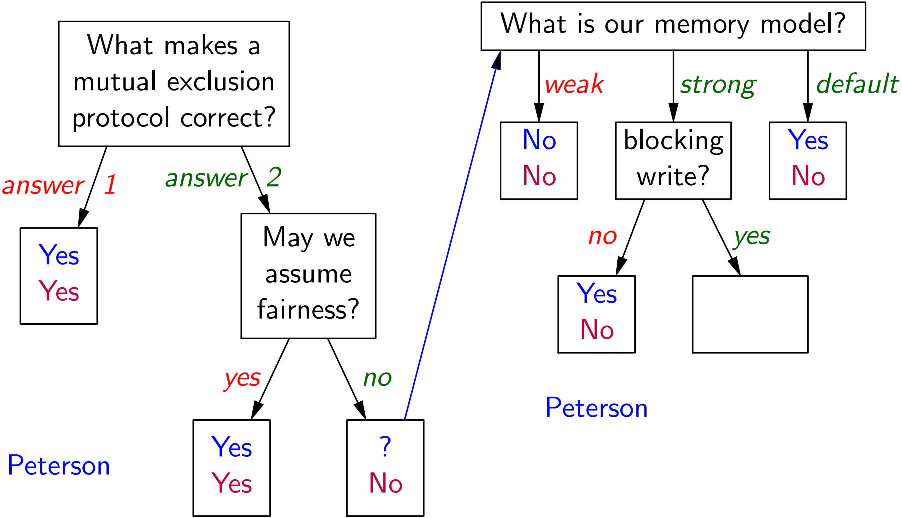
Do correct mutual exclusion protocols exists?
and can they be modelled in CCS?



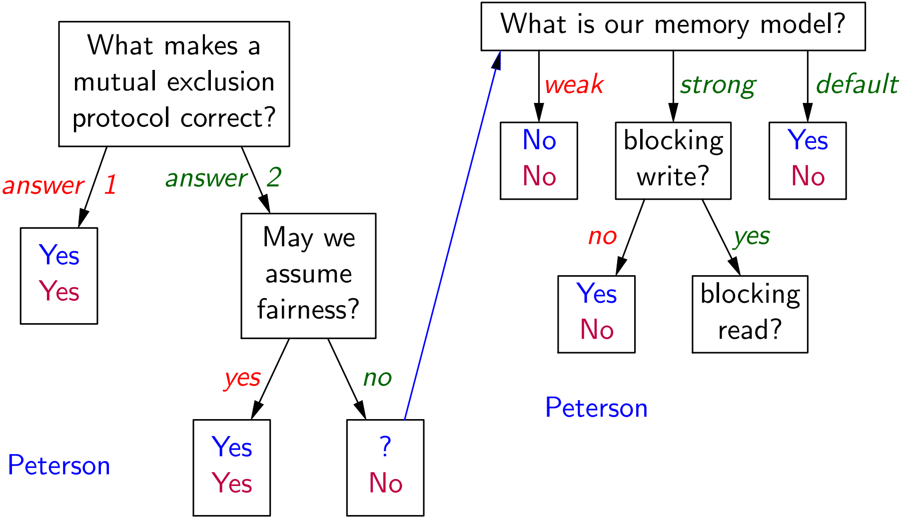
Do correct mutual exclusion protocols exists?
and can they be modelled in CCS?



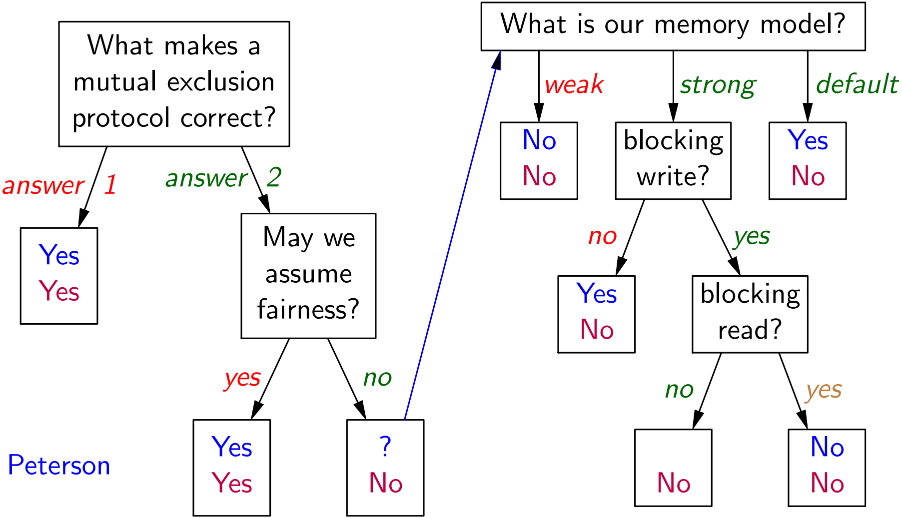
Do correct mutual exclusion protocols exists?
and can they be modelled in CCS?



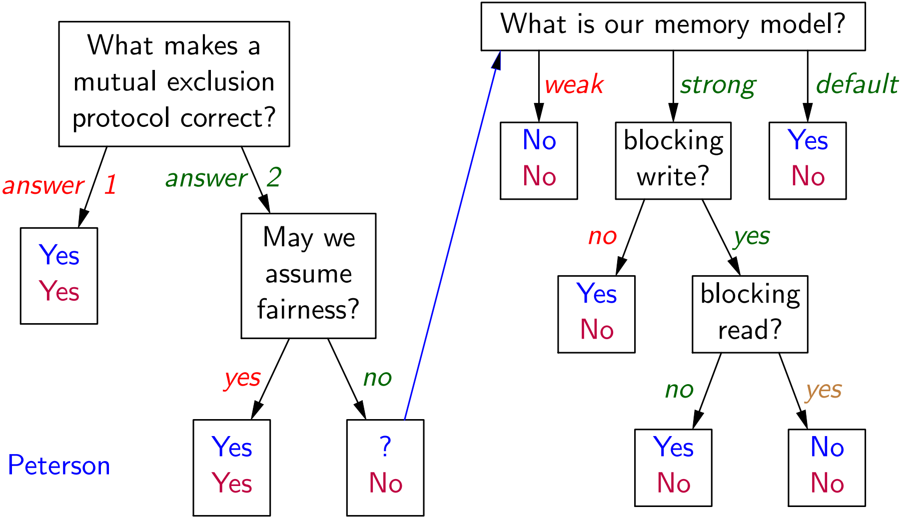
Do correct mutual exclusion protocols exists?
and can they be modelled in CCS?



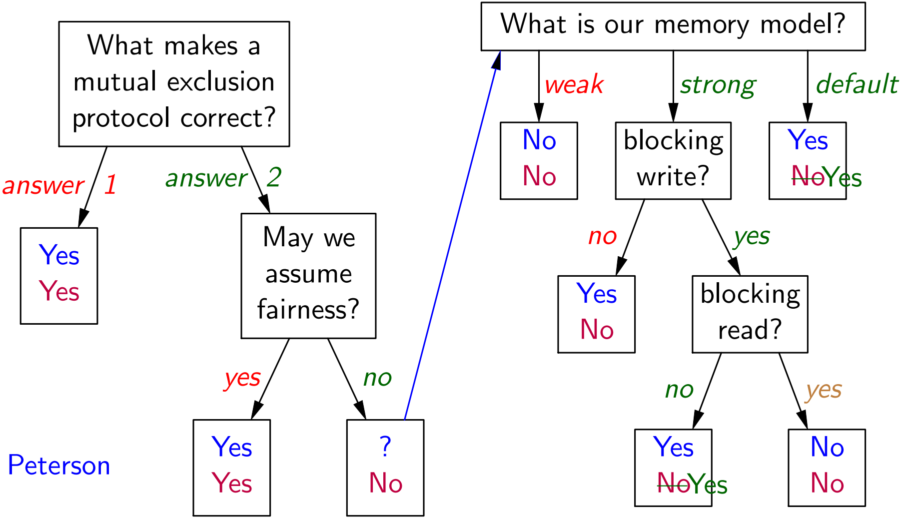
Do correct mutual exclusion protocols exists?
and can they be modelled in CCS?



Do correct mutual exclusion protocols exists?
and can they be modelled in CCS?



Do correct mutual exclusion protocols exists?
and can they be modelled in CCS with signals?



Correctness of Mutex Protocols

Process[*i*]

repeat forever

enter noncritical section[*i*]
exit noncritical section[*i*]

ready[i] := true	entry protocol
. . . [trying]	(doorway)

enter critical section[*i*]
exit critical section[*i*]

. . .	
ready[i] := false	exit protocol

Correctness of Mutex Protocols

Process[i]

repeat forever

enter noncritical section[i]
exit noncritical section[i]

ready[i] := true	entry protocol
... [trying]	(doorway)

enter critical section[i]
exit critical section[i]

...	
ready[i] := false	exit protocol

Correctness properties are quantified over all system runs (modelled as paths in the labelled trans. system representation).

Correctness of Mutex Protocols

Process[*i*]

repeat forever

enter noncritical section[*i*]
exit noncritical section[*i*]

ready[i] := true	entry protocol
. . . [trying]	(doorway)

enter critical section[*i*]
exit critical section[*i*]

. . .	
ready[i] := false	exit protocol

Correctness properties are quantified over all system runs (modelled as paths in the labelled trans. system representation).

Safety: There is no run in which **enter-crit[*i*]** is followed by **enter-crit[*j*]** without **exit-crit[*i*]** in between.

Correctness of Mutex Protocols

Process[*i*]

repeat forever

- **enter noncritical section[*i*]**
- **exit noncritical section[*i*]**
- *ready[i] := true* — entry protocol
- . . . [trying] — (doorway)
- **enter critical section[*i*]**
- **exit critical section[*i*]**
- . . .
- *ready[i] := false* — exit protocol

Correctness properties are quantified over all system runs (modelled as paths in the labelled trans. system representation).

Safety: There is no run in which **enter-crit[*i*]** is followed by **enter-crit[*j*]** without **exit-crit[*i*]** in between.

Liveness: In each run **exit-noncrit[*i*]** is followed by **enter-crit[*i*]**.

Correctness of Mutex Protocols

Process[i]

repeat forever

enter noncritical section[i]
exit noncritical section[i]

ready[i] := true	entry protocol
… [trying]	(doorway)

enter critical section[i]
exit critical section[i]

…

ready[i] := false	exit protocol

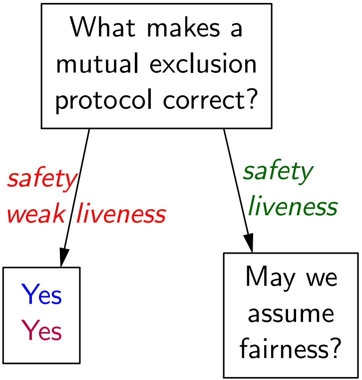
Correctness properties are quantified over all system runs (modelled as paths in the labelled trans. system representation).

Safety: There is no run in which **enter-crit[i]** is followed by **enter-crit[j]** without **exit-crit[i]** in between.

Liveness: In each run **exit-noncrit[i]** is followed by **enter-crit[i]**.

Weak liveness: In each run *ready[i]=true* is followed by **enter-crit[i]**.

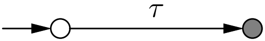
Do correct mutual exclusion protocols exists? and can they be modelled in CCS with signals?



Peterson

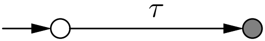
Progress, Justness and Fairness

Progress: Any process in a state that admits a non-blocking action will eventually perform an action.



Progress, Justness and Fairness

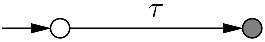
Progress: Any process in a state that admits a non-blocking action will eventually perform an action.



Justness or local progress: If a combination of components in a parallel composition is in a state admitting a non-blocking action, then one (or more) of them will eventually partake in an action.

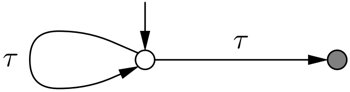
Progress, Justness and Fairness

Progress: Any process in a state that admits a non-blocking action will eventually perform an action.

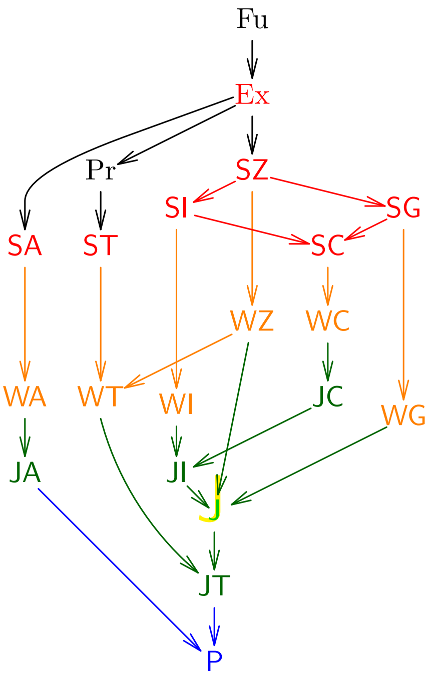


Justness or local progress: If a combination of components in a parallel composition is in a state admitting a non-blocking action, then one (or more) of them will eventually partake in an action.

Weak fairness: If (from some point onwards) a task is enabled perpetually, then it will eventually occur.

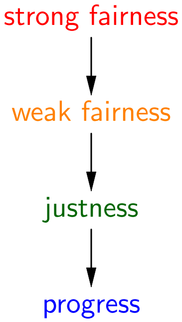


Strong fairness: If (from some point onwards) a task is enabled infinity often, then it will eventually occur.

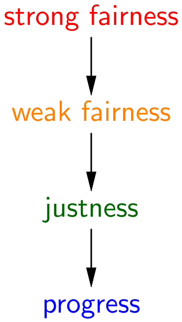


Fu
Ex
Pr
SZ
SI
SG
SA
ST
SC
WZ
WC
WA
WT
WI
JC
WG
JA
JI
J
JT
P

Fairness hierarchy



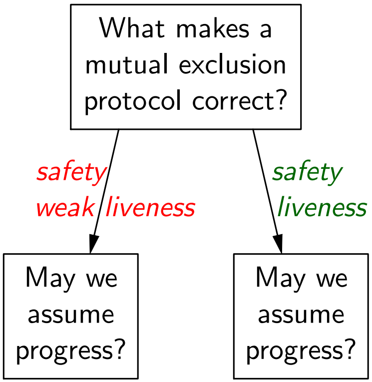
Fairness hierarchy



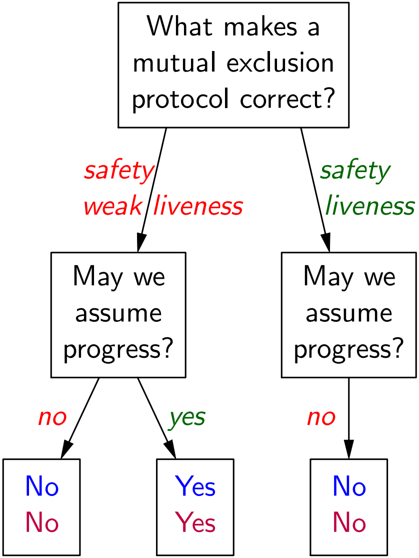
Liveness: In each run **exit-noncrit[*i*]** is followed by **enter-crit[*i*]**.

The stronger our progress/justness/fairness assumption, the fewer paths counts as runs, and the more likely it is that Liveness holds.

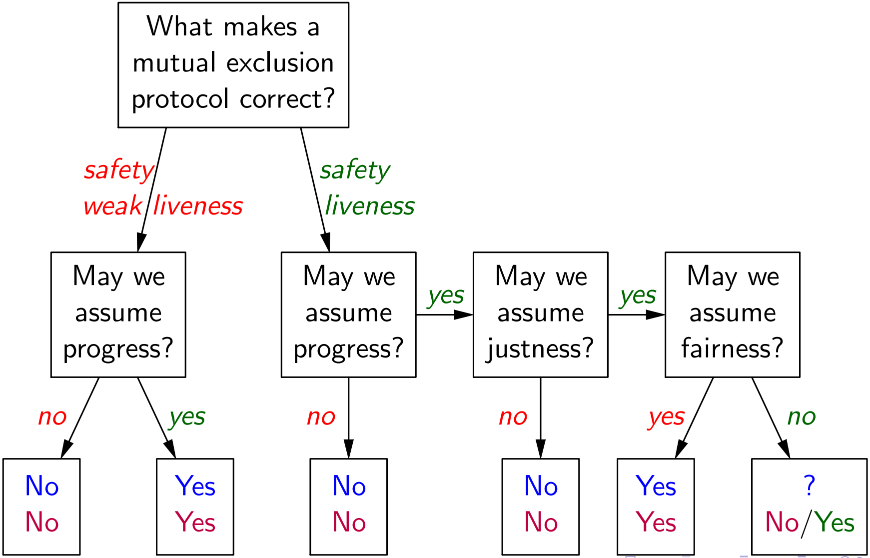
Do correct mutual exclusion protocols exists? and can they be modelled in CCS with signals?



Do correct mutual exclusion protocols exists?
and can they be modelled in CCS with signals?



Do correct mutual exclusion protocols exists?
and can they be modelled in CCS with signals?



Peterson's mutual exclusion protocol

Process A

repeat forever

$$\begin{cases} \ell_1 & \textbf{noncritical section} \\ \ell_2 & readyA := true \\ \ell_3 & turn := B \\ \ell_4 & \textbf{await}\ (readyB = false \lor turn = A) \\ \ell_5 & \textbf{critical section} \\ \ell_6 & readyA := false \end{cases}$$

Process B

repeat forever

$$\begin{cases} m_1 & \textbf{noncritical section} \\ m_2 & readyB := true \\ m_3 & turn := A \\ m_4 & \textbf{await}\ (readyA = false \lor turn = B) \\ m_5 & \textbf{critical section} \\ m_6 & readyB := false \end{cases}$$

Peterson's algorithm (pseudocode)

The Processes A and B can be modelled as

$$\mathrm{A} \stackrel{def}{=} \textbf{noncritA} \,.\, \overline{asgn^{true}_{readyA}} \,.\, \overline{asgn^{\mathrm{B}}_{turn}} \,.\, (n^{false}_{readyB} + n^{\mathrm{A}}_{turn}) \,.\, \textbf{critA} \,.\, \overline{asgn^{false}_{readyA}} \,.\, \mathrm{A}\,,$$

$$\mathrm{B} \stackrel{def}{=} \textbf{noncritB} \,.\, \overline{asgn^{true}_{readyB}} \,.\, \overline{asgn^{\mathrm{A}}_{turn}} \,.\, (n^{false}_{readyA} + n^{\mathrm{B}}_{turn}) \,.\, \textbf{critB} \,.\, \overline{asgn^{false}_{readyB}} \,.\, \mathrm{B}\,,$$

The variable *turn*:

$$Turn^{\mathrm{A}} \stackrel{def}{=} asgn^{\mathrm{A}}_{turn} \,.\, Turn^{\mathrm{A}} + asgn^{\mathrm{B}}_{turn} \,.\, Turn^{\mathrm{B}} + \overline{n^{\mathrm{A}}_{turn}} \,.\, Turn^{\mathrm{A}}$$

The problem

ReadyA = false
ReadyB = false
Turn = A

Process A

repeat forever

$$\left\{\begin{array}{ll} \ell_1 & \textbf{noncritical section} \\ \ell_2 & readyA := true \\ \ell_3 & turn := B \\ \ell_4 & \textbf{await}\,(readyB = false \vee turn = A) \\ \ell_5 & \textbf{critical section} \\ \ell_6 & readyA := false \end{array}\right.$$

Process B

repeat forever

$$\left\{\begin{array}{ll} m_1 & \textbf{noncritical section} \\ m_2 & readyB := true \\ m_3 & turn := A \\ m_4 & \textbf{await}\,(readyA = false \vee turn = B) \\ m_5 & \textbf{critical section} \\ m_6 & readyB := false \end{array}\right.$$

The problem

ReadyA = false
ReadyB = false
Turn = A

Process A

repeat forever

ℓ_1 **noncritical section**
ℓ_2 $readyA := true$
ℓ_3 $turn := B$
ℓ_4 **await** $(readyB = false \vee turn = A)$
ℓ_5 **critical section**
ℓ_6 $readyA := false$

Process B

repeat forever

m_1 **noncritical section**
m_2 $readyB := true$
m_3 $turn := A$
m_4 **await** $(readyA = false \vee turn = B)$
m_5 **critical section**
m_6 $readyB := false$

The problem

ReadyA = true
ReadyB = false
Turn = A

Process A

repeat forever

$$\left\{\begin{array}{ll} \ell_1 & \textbf{noncritical section} \\ \ell_2 & readyA := true \\ \ell_3 & turn := B \\ \ell_4 & \textbf{await}\ (readyB = false \vee turn = A) \\ \ell_5 & \textbf{critical section} \\ \ell_6 & readyA := false \end{array}\right.$$

Process B

repeat forever

$$\left\{\begin{array}{ll} m_1 & \textbf{noncritical section} \\ m_2 & readyB := true \\ m_3 & turn := A \\ m_4 & \textbf{await}\ (readyA = false \vee turn = B) \\ m_5 & \textbf{critical section} \\ m_6 & readyB := false \end{array}\right.$$

The problem

ReadyA = true
ReadyB = false
Turn = B

Process A

repeat forever

ℓ_1 **noncritical section**
ℓ_2 $readyA := true$
ℓ_3 $turn := B$
ℓ_4 **await** $(readyB = false \lor turn = A)$
ℓ_5 **critical section**
ℓ_6 $readyA := false$

Process B

repeat forever

m_1 **noncritical section**
m_2 $readyB := true$
m_3 $turn := A$
m_4 **await** $(readyA = false \lor turn = B)$
m_5 **critical section**
m_6 $readyB := false$

The problem

ReadyA = true
ReadyB = false
Turn = B

Process A

repeat forever

ℓ_1	**noncritical section**
ℓ_2	$readyA := true$
ℓ_3	$turn := B$
ℓ_4	**await** $(readyB = false \vee turn = A)$
ℓ_5	**critical section**
ℓ_6	$readyA := false$

Process B

repeat forever

m_1	**noncritical section**
m_2	$readyB := true$
m_3	$turn := A$
m_4	**await** $(readyA = false \vee turn = B)$
m_5	**critical section**
m_6	$readyB := false$

The problem

ReadyA = true
ReadyB = true
Turn = B

Process A

repeat forever

ℓ_1 **noncritical section**
ℓ_2 $readyA := true$
ℓ_3 $turn := B$
ℓ_4 **await** $(readyB = false \lor turn = A)$
ℓ_5 **critical section**
ℓ_6 $readyA := false$

Process B

repeat forever

m_1 **noncritical section**
m_2 $readyB := true$
m_3 $turn := A$
m_4 **await** $(readyA = false \lor turn = B)$
m_5 **critical section**
m_6 $readyB := false$

The problem

ReadyA = true
ReadyB = true
Turn = A

Here B is blocked
But the combination {turn,A} can take and action so one of them has to take an action (justness).

Process A

repeat forever

$$\begin{cases} \ell_1 & \textbf{noncritical section} \\ \ell_2 & readyA := true \\ \ell_3 & turn := B \\ \ell_4 & \textbf{await}\ (readyB = false \vee turn = A) \\ \ell_5 & \textbf{critical section} \\ \ell_6 & readyA := false \end{cases}$$

Process B

repeat forever

$$\begin{cases} m_1 & \textbf{noncritical section} \\ m_2 & readyB := true \\ m_3 & turn := A \\ m_4 & \textbf{await}\ (readyA = false \vee turn = B) \\ m_5 & \textbf{critical section} \\ m_6 & readyB := false \end{cases}$$

The problem

ReadyA = true
ReadyB = true
Turn = A

Here B is blocked
But the combination {turn,A} can take and action so one of them has to take an action (justness).

The only possible action is that A reads turn = A

Process A

repeat forever

ℓ_1	**noncritical section**
ℓ_2	$readyA := true$
ℓ_3	$turn := B$
ℓ_4	**await** $(readyB = false \vee turn = A)$
ℓ_5	**critical section**
ℓ_6	$readyA := false$

Process B

repeat forever

m_1	**noncritical section**
m_2	$readyB := true$
m_3	$turn := A$
m_4	**await** $(readyA = false \vee turn = B)$
m_5	**critical section**
m_6	$readyB := false$

The problem

ReadyA = true
ReadyB = true
Turn = A

Process A

repeat forever

ℓ_1 **noncritical section**
ℓ_2 $readyA := true$
ℓ_3 $turn := B$
ℓ_4 **await** $(readyB = false \vee turn = A)$
ℓ_5 **critical section**
ℓ_6 $readyA := false$

Process B

repeat forever

m_1 **noncritical section**
m_2 $readyB := true$
m_3 $turn := A$
m_4 **await** $(readyA = false \vee turn = B)$
m_5 **critical section**
m_6 $readyB := false$

The problem

ReadyA = true
ReadyB = true
Turn = A

Process A

repeat forever

ℓ_1 **noncritical section**
ℓ_2 $readyA := true$
ℓ_3 $turn := B$
ℓ_4 **await** $(readyB = false \lor turn = A)$
ℓ_5 **critical section**
ℓ_6 $readyA := false$

Process B

repeat forever

m_1 **noncritical section**
m_2 $readyB := true$
m_3 $turn := A$
m_4 **await** $(readyA = false \lor turn = B)$
m_5 **critical section**
m_6 $readyB := false$

The problem

ReadyA = false
ReadyB = true
Turn = A

Process A

repeat forever

ℓ_1 **noncritical section**
ℓ_2 $readyA := true$
ℓ_3 $turn := B$
ℓ_4 **await** $(readyB = false \vee turn = A)$
ℓ_5 **critical section**
ℓ_6 $readyA := false$

Process B

repeat forever

m_1 **noncritical section**
m_2 $readyB := true$
m_3 $turn := A$
m_4 **await** $(readyA = false \vee turn = B)$
m_5 **critical section**
m_6 $readyB := false$

The problem

ReadyA = false
ReadyB = true
Turn = A

Process A

repeat forever

ℓ_1 **noncritical section**
ℓ_2 $readyA := true$
ℓ_3 $turn := B$
ℓ_4 **await** $(readyB = false \lor turn = A)$
ℓ_5 **critical section**
ℓ_6 $readyA := false$

Process B

repeat forever

m_1 **noncritical section**
m_2 $readyB := true$
m_3 $turn := A$
m_4 **await** $(readyA = false \lor turn = B)$
m_5 **critical section**
m_6 $readyB := false$

The problem

ReadyA = false
ReadyB = true
Turn = A

Process A

repeat forever

ℓ_1 **noncritical section**
ℓ_2 $readyA := true$
ℓ_3 $turn := B$
ℓ_4 **await** $(readyB = false \vee turn = A)$
ℓ_5 **critical section**
ℓ_6 $readyA := false$

Process B

repeat forever

m_1 **noncritical section**
m_2 $readyB := true$
m_3 $turn := A$
m_4 **await** $(readyA = false \vee turn = B)$
m_5 **critical section**
m_6 $readyB := false$

The problem

ReadyA = false
ReadyB = false
Turn = A

Process A

repeat forever

ℓ_1 **noncritical section**
ℓ_2 $readyA := true$
ℓ_3 $turn := B$
ℓ_4 **await** $(readyB = false \vee turn = A)$
ℓ_5 **critical section**
ℓ_6 $readyA := false$

Process B

repeat forever

m_1 **noncritical section**
m_2 $readyB := true$
m_3 $turn := A$
m_4 **await** $(readyA = false \vee turn = B)$
m_5 **critical section**
m_6 $readyB := false$

The problem

ReadyA = false
ReadyB = false
Turn = A

Process A

repeat forever

ℓ_1 **noncritical section**
ℓ_2 $readyA := true$
ℓ_3 $turn := B$
ℓ_4 **await** $(readyB = false \vee turn = A)$
ℓ_5 **critical section**
ℓ_6 $readyA := false$

Process B

repeat forever

m_1 **noncritical section**
m_2 $readyB := true$
m_3 $turn := A$
m_4 **await** $(readyA = false \vee turn = B)$
m_5 **critical section**
m_6 $readyB := false$

The problem

ReadyA = false
ReadyB = false
Turn = A

Process A

repeat forever

$$\left\{\begin{array}{ll} \ell_1 & \textbf{noncritical section} \\ \ell_2 & readyA := true \\ \ell_3 & turn := B \\ \ell_4 & \textbf{await}\ (readyB = false \vee turn = A) \\ \ell_5 & \textbf{critical section} \\ \ell_6 & readyA := false \end{array}\right.$$

Process B

repeat forever

$$\left\{\begin{array}{ll} m_1 & \textbf{noncritical section} \\ m_2 & readyB := true \\ m_3 & turn := A \\ m_4 & \textbf{await}\ (readyA = false \vee turn = B) \\ m_5 & \textbf{critical section} \\ m_6 & readyB := false \end{array}\right.$$

The problem

ReadyA = false
ReadyB = true
Turn = A

Process A

repeat forever

ℓ_1 **noncritical section**
ℓ_2 $readyA := true$
ℓ_3 $turn := B$
ℓ_4 **await** $(readyB = false \vee turn = A)$
ℓ_5 **critical section**
ℓ_6 $readyA := false$

Process B

repeat forever

m_1 **noncritical section**
m_2 $readyB := true$
m_3 $turn := A$
m_4 **await** $(readyA = false \vee turn = B)$
m_5 **critical section**
m_6 $readyB := false$

The problem

ReadyA = false
ReadyB = true
Turn = A

Process A

repeat forever

ℓ_1 **noncritical section**
ℓ_2 $readyA := true$
ℓ_3 $turn := B$
ℓ_4 **await** $(readyB = false \lor turn = A)$
ℓ_5 **critical section**
ℓ_6 $readyA := false$

Process B

repeat forever

m_1 **noncritical section**
m_2 $readyB := true$
m_3 $turn := A$
m_4 **await** $(readyA = false \lor turn = B)$
m_5 **critical section**
m_6 $readyB := false$

The problem

ReadyA = false
ReadyB = true
Turn = A

Process A

repeat forever

ℓ_1	**noncritical section**
ℓ_2	$readyA := true$
ℓ_3	$turn := B$
ℓ_4	**await** $(readyB = false \lor turn = A)$
ℓ_5	**critical section**
ℓ_6	$readyA := false$

Process B

repeat forever

m_1	**noncritical section**
m_2	$readyB := true$
m_3	$turn := A$
m_4	**await** $(readyA = false \lor turn = B)$
m_5	**critical section**
m_6	$readyB := false$

The problem

ReadyA = false
ReadyB = true
Turn = A

Process A

repeat forever

ℓ_1 **noncritical section**
ℓ_2 $readyA := true$
ℓ_3 $turn := B$
ℓ_4 **await** $(readyB = false \lor turn = A)$
ℓ_5 **critical section**
ℓ_6 $readyA := false$

Process B

repeat forever

m_1 **noncritical section**
m_2 $readyB := true$
m_3 $turn := A$
m_4 **await** $(readyA = false \lor turn = B)$
m_5 **critical section**
m_6 $readyB := false$

The problem

ReadyA = false
ReadyB = false
Turn = A

Process A

repeat forever

$$\left\{\begin{array}{ll} \ell_1 & \textbf{noncritical section} \\ \ell_2 & readyA := true \\ \ell_3 & turn := B \\ \ell_4 & \textbf{await}\ (readyB = false \vee turn = A) \\ \ell_5 & \textbf{critical section} \\ \ell_6 & readyA := false \end{array}\right.$$

Process B

repeat forever

$$\left\{\begin{array}{ll} m_1 & \textbf{noncritical section} \\ m_2 & readyB := true \\ m_3 & turn := A \\ m_4 & \textbf{await}\ (readyA = false \vee turn = B) \\ m_5 & \textbf{critical section} \\ m_6 & readyB := false \end{array}\right.$$

The problem

ReadyA = false
ReadyB = false
Turn = A

Process A

repeat forever

ℓ_1 **noncritical section**
ℓ_2 $readyA := true$
ℓ_3 $turn := B$
ℓ_4 **await** $(readyB = false \vee turn = A)$
ℓ_5 **critical section**
ℓ_6 $readyA := false$

Process B

repeat forever

m_1 **noncritical section**
m_2 $readyB := true$
m_3 $turn := A$
m_4 **await** $(readyA = false \vee turn = B)$
m_5 **critical section**
m_6 $readyB := false$

The problem

ReadyA = false
ReadyB = true
Turn = A

Process A

repeat forever

ℓ_1 **noncritical section**
ℓ_2 $readyA := true$
ℓ_3 $turn := B$
ℓ_4 **await** $(readyB = false \vee turn = A)$
ℓ_5 **critical section**
ℓ_6 $readyA := false$

Process B

repeat forever

m_1 **noncritical section**
m_2 $readyB := true$
m_3 $turn := A$
m_4 **await** $(readyA = false \vee turn = B)$
m_5 **critical section**
m_6 $readyB := false$

The problem

ReadyA = false
ReadyB = true
Turn = A

Process A

repeat forever

ℓ_1	**noncritical section**
ℓ_2	$readyA := true$
ℓ_3	$turn := B$
ℓ_4	**await** $(readyB = false \vee turn = A)$
ℓ_5	**critical section**
ℓ_6	$readyA := false$

Process B

repeat forever

m_1	**noncritical section**
m_2	$readyB := true$
m_3	$turn := A$
m_4	**await** $(readyA = false \vee turn = B)$
m_5	**critical section**
m_6	$readyB := false$

The problem

ReadyA = false
ReadyB = true
Turn = A

Process A

repeat forever

ℓ_1	**noncritical section**
ℓ_2	$readyA := true$
ℓ_3	$turn := B$
ℓ_4	**await** $(readyB = false \vee turn = A)$
ℓ_5	**critical section**
ℓ_6	$readyA := false$

Process B

repeat forever

m_1	**noncritical section**
m_2	$readyB := true$
m_3	$turn := A$
m_4	**await** $(readyA = false \vee turn = B)$
m_5	**critical section**
m_6	$readyB := false$

The problem

ReadyA = false
ReadyB = true
Turn = A

Process A

repeat forever

ℓ_1 **noncritical section**
ℓ_2 $readyA := true$
ℓ_3 $turn := B$
ℓ_4 **await** $(readyB = false \lor turn = A)$
ℓ_5 **critical section**
ℓ_6 $readyA := false$

Process B

repeat forever

m_1 **noncritical section**
m_2 $readyB := true$
m_3 $turn := A$
m_4 **await** $(readyA = false \lor turn = B)$
m_5 **critical section**
m_6 $readyB := false$

The problem

ReadyA = false
ReadyB = true
Turn = A

Why is it possible ?

Because A cannot progress by itself,
it must communicate with ReadyA.

The combination {ReadyA,A} can take an action.

Justness says : at least one of them has to take an action.

Process A

repeat forever

ℓ_1 **noncritical section**
ℓ_2 $readyA := true$
ℓ_3 $turn := B$
ℓ_4 **await** $(readyB = false \lor turn = A)$
ℓ_5 **critical section**
ℓ_6 $readyA := false$

Process B

repeat forever

m_1 **noncritical section**
m_2 $readyB := true$
m_3 $turn := A$
m_4 **await** $(readyA = false \lor turn = B)$
m_5 **critical section**
m_6 $readyB := false$

The problem

ReadyA = false
ReadyB = true
Turn = A

Why is it possible ?

Justness says : at least one of {ReadyA,A} has to take an action.

But in our scenario {ReadyA} takes an action (communication with B at line m4).

The scenario is just. But liveness fails.

Process A

repeat forever

$$\left\{\begin{array}{ll} \ell_1 & \textbf{noncritical section} \\ \ell_2 & readyA := true \\ \ell_3 & turn := B \\ \ell_4 & \textbf{await}\ (readyB = false \vee turn = A) \\ \ell_5 & \textbf{critical section} \\ \ell_6 & readyA := false \end{array}\right.$$

Process B

repeat forever

$$\left\{\begin{array}{ll} m_1 & \textbf{noncritical section} \\ m_2 & readyB := true \\ m_3 & turn := A \\ m_4 & \textbf{await}\ (readyA = false \vee turn = B) \\ m_5 & \textbf{critical section} \\ m_6 & readyB := false \end{array}\right.$$

CCS with signals

$$E ::= 0 \mid \alpha.P \mid P + Q \mid P|Q \mid P\backslash L \mid P[f] \mid A$$

CCS with signals

$$E ::= 0 \;\mid\; \alpha.P \;\mid\; P + Q \;\mid\; P|Q \;\mid\; P\backslash L \;\mid\; P[f] \;\mid\; A \;\mid\; E\hat{\ }s$$

CCS with signals

$$E ::= 0 \mid \alpha.P \mid P + Q \mid P|Q \mid P\backslash L \mid P[f] \mid A \mid E\hat{\ }s$$

$$\alpha.P \xrightarrow{\alpha} P \qquad \frac{P \xrightarrow{\alpha} P'}{P + Q \xrightarrow{\alpha} P'} \qquad \frac{Q \xrightarrow{\alpha} Q'}{P + Q \xrightarrow{\alpha} Q'}$$

$$\frac{P \xrightarrow{\alpha} P'}{P|Q \xrightarrow{\alpha} P'|Q} \qquad \frac{P \xrightarrow{a} P', \ Q \xrightarrow{\bar{a}} Q'}{P|Q \xrightarrow{\tau} P'|Q'} \qquad \frac{Q \xrightarrow{\alpha} Q'}{P|Q \xrightarrow{\alpha} P|Q'}$$

$$\frac{P \xrightarrow{\alpha} P'}{P\backslash L \xrightarrow{\alpha} P'\backslash L} \ (\alpha, \bar{\alpha} \notin L) \qquad \frac{P \xrightarrow{\alpha} P'}{P[f] \xrightarrow{f(\alpha)} P'[f]} \qquad \frac{P \xrightarrow{\alpha} P'}{A \xrightarrow{\alpha} P'} \ (A \stackrel{def}{=} P)$$

CCS with signals

$$E ::= 0 \mid \alpha.P \mid P + Q \mid P|Q \mid P\backslash L \mid P[f] \mid A \mid E\hat{\ }s$$

$$\alpha.P \xrightarrow{\alpha} P \qquad \frac{P \xrightarrow{\alpha} P'}{P + Q \xrightarrow{\alpha} P'} \qquad \frac{Q \xrightarrow{\alpha} Q'}{P + Q \xrightarrow{\alpha} Q'}$$

$$\frac{P \xrightarrow{\alpha} P'}{P|Q \xrightarrow{\alpha} P'|Q} \qquad \frac{P \xrightarrow{a} P', \ Q \xrightarrow{\bar{a}} Q'}{P|Q \xrightarrow{\tau} P'|Q'} \qquad \frac{Q \xrightarrow{\alpha} Q'}{P|Q \xrightarrow{\alpha} P|Q'}$$

$$\frac{P \xrightarrow{\alpha} P'}{P\backslash L \xrightarrow{\alpha} P'\backslash L}\ (\alpha, \bar{\alpha} \notin L) \qquad \frac{P \xrightarrow{\alpha} P'}{P[f] \xrightarrow{f(\alpha)} P'[f]} \qquad \frac{P \xrightarrow{\alpha} P'}{A \xrightarrow{\alpha} P'}\ (A \stackrel{def}{=} P)$$

$$\frac{P \xrightarrow{\alpha} P'}{P\hat{\ }s \xrightarrow{\alpha} P'} \qquad (P\hat{\ }s)^{\curvearrowright s} \qquad \frac{P^{\curvearrowright s}}{(P\hat{\ }t)^{\curvearrowright s}} \qquad \frac{P^{\curvearrowright s}}{(P + Q)^{\curvearrowright s}} \qquad \frac{Q^{\curvearrowright s}}{(P + Q)^{\curvearrowright s}}$$

$$\frac{P^{\curvearrowright s}}{(P|Q)^{\curvearrowright s}} \qquad \frac{P^{\curvearrowright s}, \ Q \xrightarrow{s} Q'}{P|Q \xrightarrow{\tau} P|Q'} \qquad \frac{P \xrightarrow{s} P', \ Q^{\curvearrowright s}}{P|Q \xrightarrow{\tau} P'|Q} \qquad \frac{Q^{\curvearrowright s}}{(P|Q)^{\curvearrowright s}}$$

$$\frac{P^{\curvearrowright s}}{(P\backslash L)^{\curvearrowright s}}\ (s \notin L) \qquad \frac{P^{\curvearrowright s}}{P[f]^{\curvearrowright f(s)}} \qquad \frac{P^{\curvearrowright s}}{A^{\curvearrowright s}}\ (A \stackrel{def}{=} P)$$

Modelling variables in CCSS

$$Turn^{\mathrm{A}} \stackrel{def}{=} asgn^{\mathrm{A}}_{turn} \,.\, Turn^{\mathrm{A}} + asgn^{\mathrm{B}}_{turn} \,.\, Turn^{\mathrm{B}} + \overline{n^{\mathrm{A}}_{turn}} \,.\, Turn^{\mathrm{A}}$$

$$Turn^{\mathrm{A}} \stackrel{def}{=} (asgn^{\mathrm{A}}_{turn} \,.\, Turn^{\mathrm{A}} + asgn^{\mathrm{B}}_{turn} \,.\, Turn^{\mathrm{B}})\hat{\ } n^{\mathrm{A}}_{turn}$$

Do correct mutual exclusion protocols exists? and can they be modelled in CCS with signals?