

Introduction

## Modelling parallel systems

Transition systems

Modeling hard- and software systems

Parallelism and communication



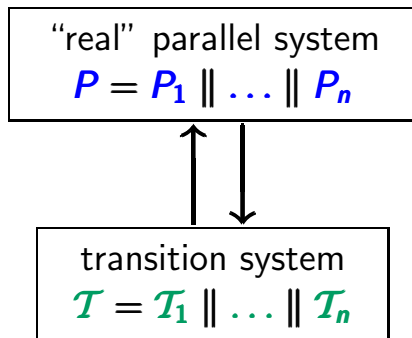
Linear Time Properties

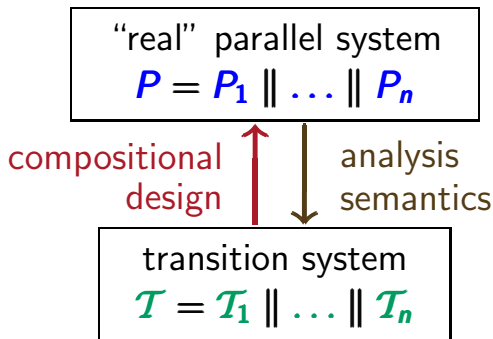
Regular Properties

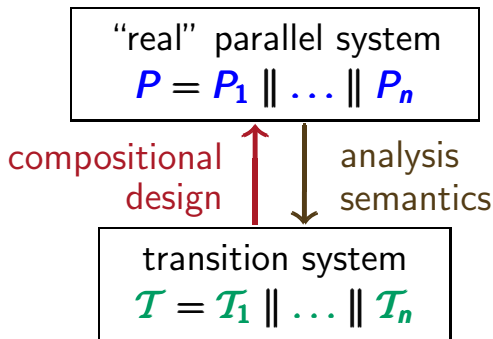
Linear Temporal Logic

Computation-Tree Logic

Equivalences and Abstraction





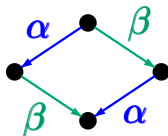
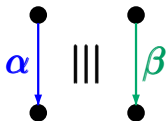


*goal:* define semantic parallel operators  
on transition systems or program graphs that  
model "real" parallel operators

- interleaving of concurrent, independent actions of parallel processes (modelled by TS)
- representation by nondeterministic choice:  
“which subprocess performs the next step?”

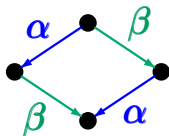
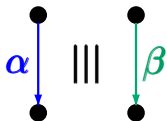
- interleaving of **concurrent, independent** actions of parallel processes (modelled by TS)
- representation by **nondeterministic choice**:  
“which subprocess performs the next step?”

$$\textit{effect}(\alpha ||| \beta) = \textit{effect}(\alpha; \beta + \beta; \alpha)$$



- interleaving of **concurrent, independent** actions of parallel processes (modelled by TS)
- representation by **nondeterministic choice**:  
“which subprocess performs the next step?”

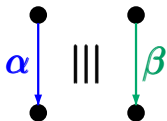
$$\text{effect}(\alpha ||| \beta) = \text{effect}(\alpha; \beta + \beta; \alpha)$$



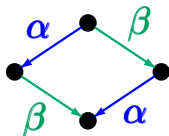
parallel execution  
of  $\alpha$  and  $\beta$  on  
*two processors*

- interleaving of **concurrent**, **independent** actions of parallel processes (modelled by TS)
- representation by **nondeterministic choice**:  
“which subprocess performs the next step?”

$$\text{effect}(\alpha ||| \beta) = \text{effect}(\alpha; \beta + \beta; \alpha)$$



parallel execution  
of  $\alpha$  and  $\beta$  on  
*two processors*  $\cong$



serial execution on  
a *single processor*  
in arbitrary order



# Interleaving operator $|||$ for TS

PC2.2-DEF-INTERLEAVING-TS

$$\mathcal{T}_1 = (S_1, Act_1, \longrightarrow_1, S_{0,1}, AP_1, L_1)$$

$$\mathcal{T}_2 = (S_2, Act_2, \longrightarrow_2, S_{0,2}, AP_2, L_2)$$

$$\mathcal{T}_1 = (S_1, Act_1, \longrightarrow_1, S_{0,1}, AP_1, L_1)$$

$$\mathcal{T}_2 = (S_2, Act_2, \longrightarrow_2, S_{0,2}, AP_2, L_2)$$

The transition system  $\mathcal{T}_1 ||| \mathcal{T}_2$  is defined by:

$$\mathcal{T}_1 ||| \mathcal{T}_2 = (S_1 \times S_2, Act_1 \cup Act_2, \longrightarrow, S_{0,1} \times S_{0,2}, AP, L)$$

where the transition relation  $\longrightarrow$  is given by:

# Interleaving operator $|||$ for TS

$$\mathcal{T}_1 = (S_1, Act_1, \longrightarrow_1, S_{0,1}, AP_1, L_1)$$

$$\mathcal{T}_2 = (S_2, Act_2, \longrightarrow_2, S_{0,2}, AP_2, L_2)$$

The transition system  $\mathcal{T}_1 ||| \mathcal{T}_2$  is defined by:

$$\mathcal{T}_1 ||| \mathcal{T}_2 = (S_1 \times S_2, Act_1 \cup Act_2, \longrightarrow, S_{0,1} \times S_{0,2}, AP, L)$$

where the transition relation  $\longrightarrow$  is given by:

$$\frac{s_1 \xrightarrow{\alpha}_1 s'_1}{\langle s_1, s_2 \rangle \xrightarrow{\alpha} \langle s'_1, s_2 \rangle} \quad \frac{s_2 \xrightarrow{\alpha}_2 s'_2}{\langle s_1, s_2 \rangle \xrightarrow{\alpha} \langle s_1, s'_2 \rangle}$$

# Interleaving operator $|||$ for TS

$$\mathcal{T}_1 = (S_1, Act_1, \longrightarrow_1, S_{0,1}, AP_1, L_1)$$

$$\mathcal{T}_2 = (S_2, Act_2, \longrightarrow_2, S_{0,2}, AP_2, L_2)$$

The transition system  $\mathcal{T}_1 ||| \mathcal{T}_2$  is defined by:

$$\mathcal{T}_1 ||| \mathcal{T}_2 = (S_1 \times S_2, Act_1 \cup Act_2, \longrightarrow, S_{0,1} \times S_{0,2}, AP, L)$$

where the transition relation  $\longrightarrow$  is given by:

$$\frac{s_1 \xrightarrow{\alpha}_1 s'_1}{\langle s_1, s_2 \rangle \xrightarrow{\alpha} \langle s'_1, s_2 \rangle} \quad \frac{s_2 \xrightarrow{\alpha}_2 s'_2}{\langle s_1, s_2 \rangle \xrightarrow{\alpha} \langle s_1, s'_2 \rangle}$$

atomic propositions:  $AP = AP_1 \uplus AP_2$

labeling function:  $L(\langle s_1, s_2 \rangle) = L_1(s_1) \cup L_2(s_2)$

just a simple notation for operational semantics

$$\frac{\text{premise}}{\text{conclusion}}$$

just a simple notation for **operational semantics**

$$\frac{\text{premise}}{\text{conclusion}}$$

E.g., “the relation  $\longrightarrow$  is given by ...”

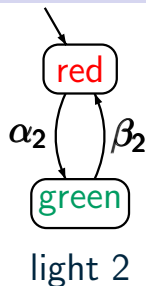
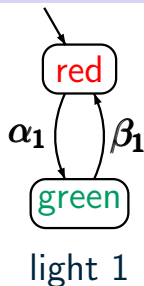
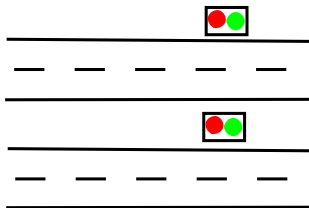
$$\frac{s_1 \xrightarrow{\alpha}_1 s'_1}{\langle s_1, s_2 \rangle \xrightarrow{\alpha} \langle s'_1, s_2 \rangle} \quad \frac{s_2 \xrightarrow{\alpha}_2 s'_2}{\langle s_1, s_2 \rangle \xrightarrow{\alpha} \langle s_1, s'_2 \rangle}$$

means that  $\longrightarrow$  is the **smallest relation** such that:

- (1) If  $s_1 \xrightarrow{\alpha}_1 s'_1$ , then  $\langle s_1, s_2 \rangle \xrightarrow{\alpha} \langle s'_1, s_2 \rangle$
- (2) If  $s_2 \xrightarrow{\alpha}_2 s'_2$ , then  $\langle s_1, s_2 \rangle \xrightarrow{\alpha} \langle s_1, s'_2 \rangle$

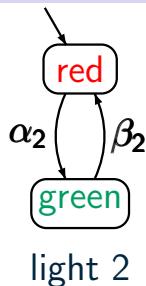
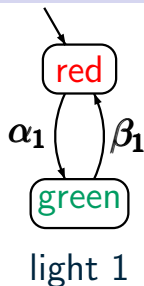
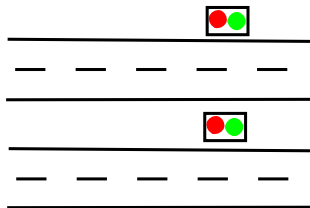
# Useless lights for non-crossing streets

PC2.2-4

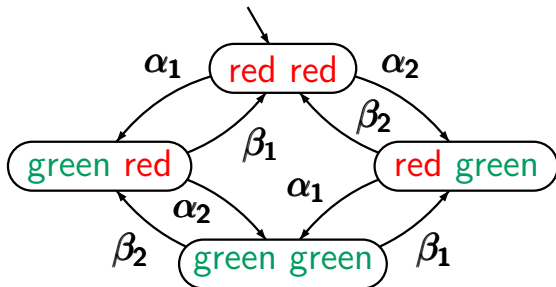


# Useless lights for non-crossing streets

PC2.2-4



light 1 ||| light 2



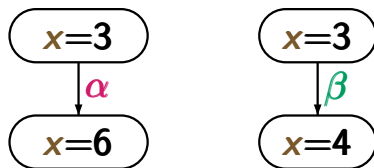


# Dependent actions

PC2.2-5

dependent actions  $\alpha \hat{=} x := 2x$  and  $\beta \hat{=} x := x + 1$

representations in  
transition systems

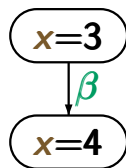
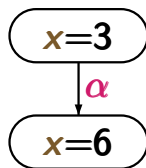


# Dependent actions

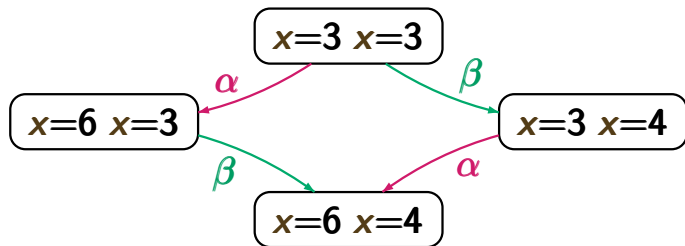
PC2.2-5

dependent actions  $\alpha \hat{=} x := 2x$  and  $\beta \hat{=} x := x + 1$

representations in  
transition systems



interleaving operator  $|||$

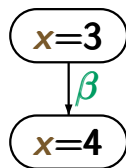
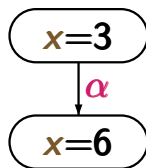


# Interleaving fails for dependent actions

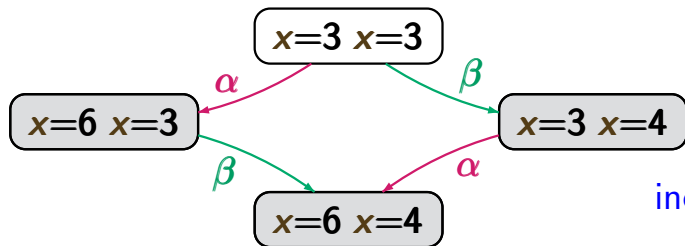
PC2.2-5

dependent actions  $\alpha \hat{=} x := 2x$  and  $\beta \hat{=} x := x + 1$

representations in  
transition systems



interleaving operator  $|||$  for transition systems “fails”



inconsistent  
states



... for modeling **parallel systems** with  
subprocesses communicating via **shared variables**

# Interleaving for program graphs

PC2.2-6

program graph  $\mathcal{P}_1$   
( $Loc_1, \dots, \hookrightarrow_1, \dots$ )

program graph  $\mathcal{P}_2$   
( $Loc_2, \dots, \hookrightarrow_2, \dots$ )

interleaving operator

$$\mathcal{P}_1 ||| \mathcal{P}_2 = (Loc_1 \times Loc_2, \dots, \hookrightarrow, \dots)$$

# Interleaving for program graphs

PC2.2-6

program graph  $\mathcal{P}_1$   
( $\text{Loc}_1, \dots, \hookrightarrow_1, \dots$ )

program graph  $\mathcal{P}_2$   
( $\text{Loc}_2, \dots, \hookrightarrow_2, \dots$ )

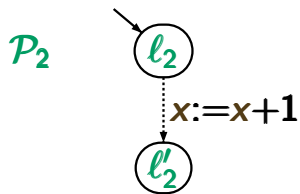
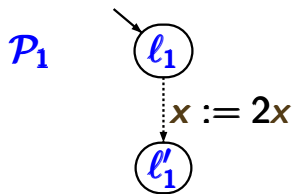
interleaving operator

$$\mathcal{P}_1 ||| \mathcal{P}_2 = (\text{Loc}_1 \times \text{Loc}_2, \dots, \hookrightarrow, \dots)$$

$$\frac{l_1 \xrightarrow[g:\alpha]{}_1 l'_1}{\langle l_1, l_2 \rangle \xrightarrow[g:\alpha]{} \langle l'_1, l_2 \rangle} \quad \frac{l_2 \xrightarrow[g:\alpha]{}_2 l'_2}{\langle l_1, l_2 \rangle \xrightarrow[g:\alpha]{} \langle l_1, l'_2 \rangle}$$

# Example: interleaving for PG

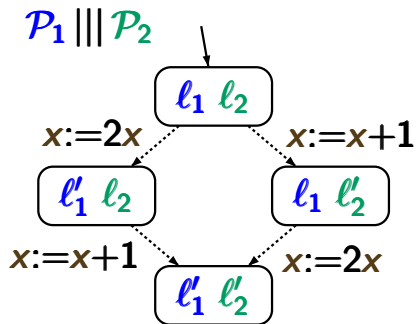
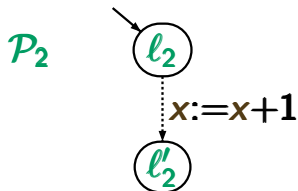
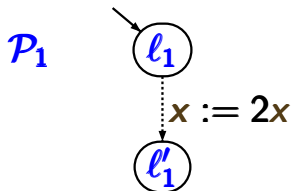
PG2.2-7





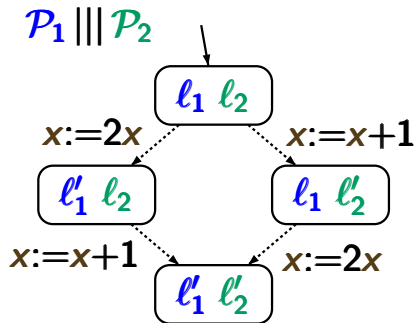
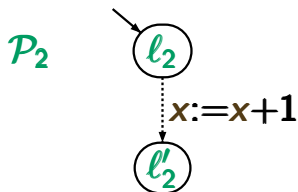
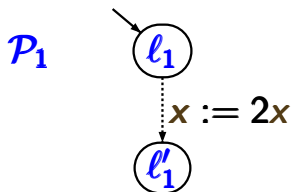
# Example: interleaving for PG

PC2.2-7

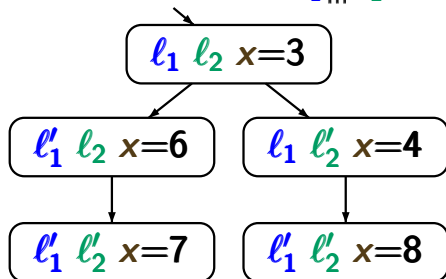


# Example: interleaving for PG

PC2.2-7

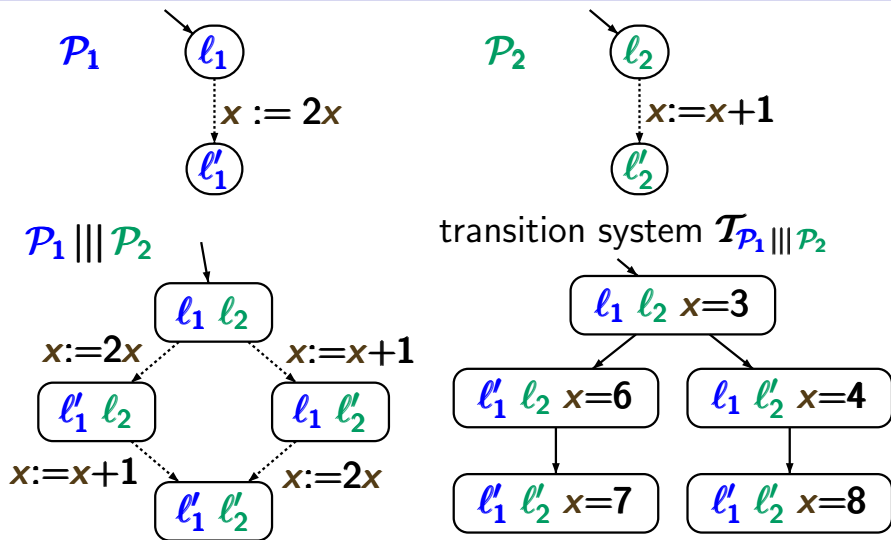


transition system  $\mathcal{T}_{\mathcal{P}_1 \parallel \mathcal{P}_2}$



# Example: interleaving for PG

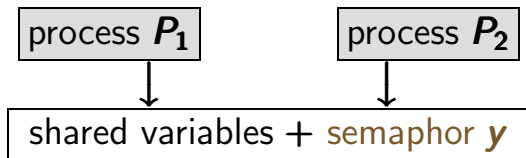
PG2.2-7



note:  $\mathcal{T}_{\mathcal{P}_1 ||| \mathcal{P}_2} \neq \mathcal{T}_{\mathcal{P}_1} ||| \mathcal{T}_{\mathcal{P}_2}$

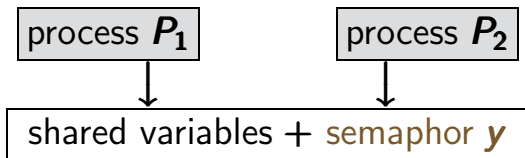
# Mutual exclusion with semaphore

PC2.2-9



# Mutual exclusion with semaphore

PC2.2-9

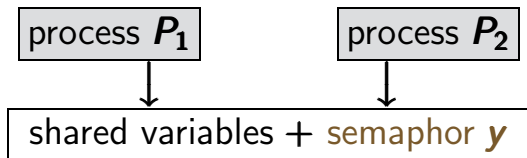


protocol for process  $P_i$

```
LOOP FOREVER
  noncritical actions;
  AWAIT  $y > 0$  DO
     $y := y - 1$ 
  OD
  critical actions;
   $y := y + 1$ 
END LOOP
```

# Mutual exclusion with semaphore

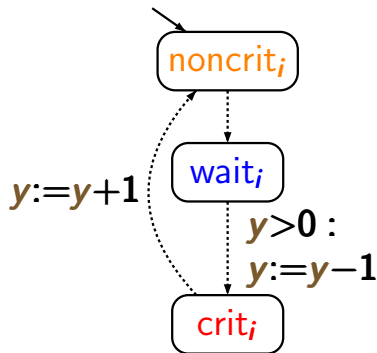
PC2.2-9



protocol for process  $P_i$

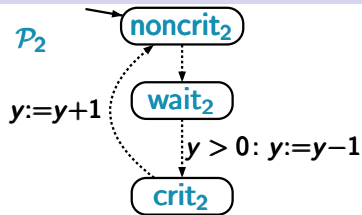
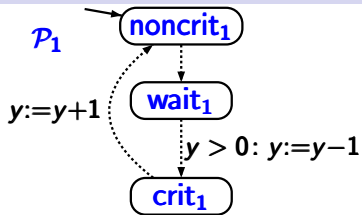
```
LOOP FOREVER
  noncritical actions;
  AWAIT  $y > 0$  DO
     $y := y - 1$ 
  OD
  critical actions;
   $y := y + 1$ 
END LOOP
```

program graph  $\mathcal{P}_i$



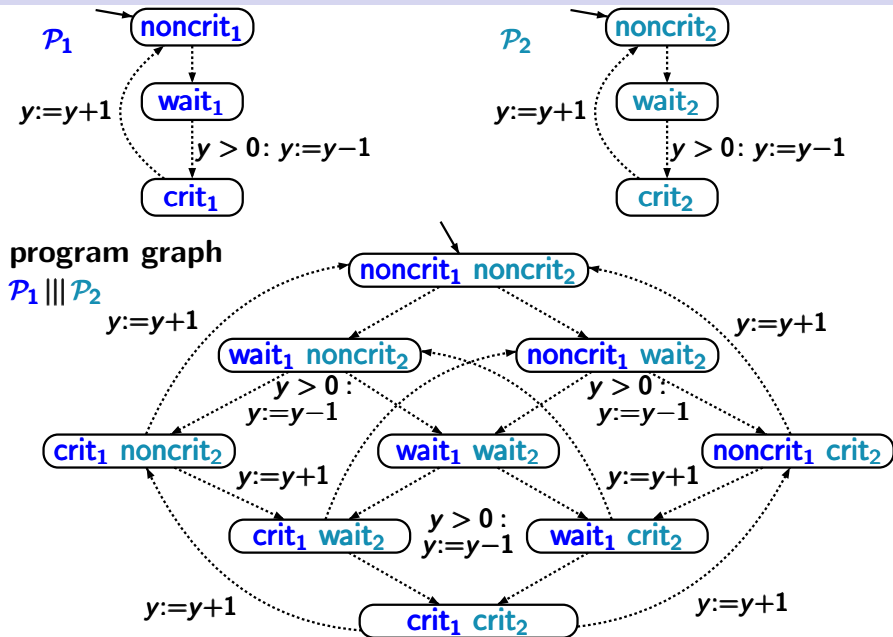
# Mutual exclusion with semaphore

PC2.2-10



# Mutual exclusion with semaphore

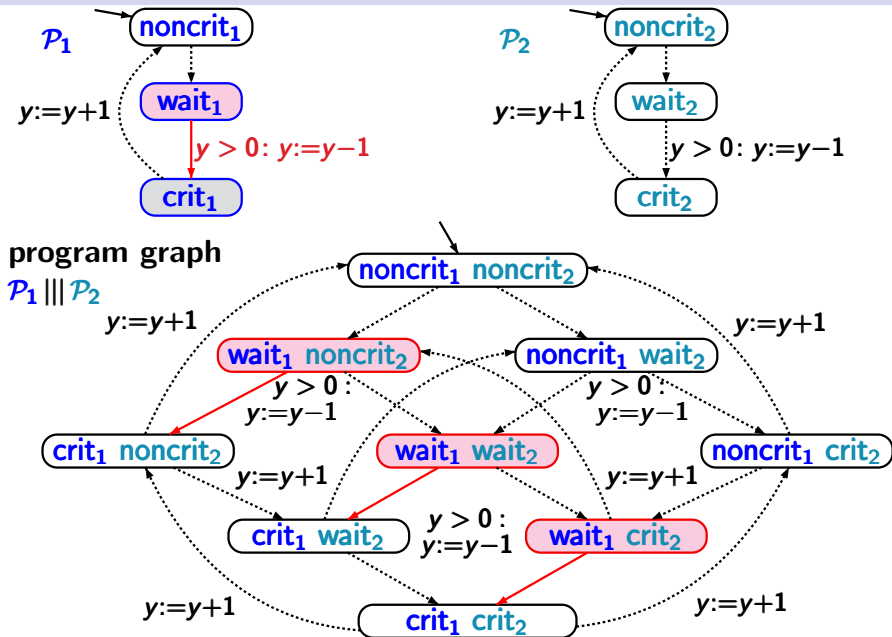
PC2.2-10





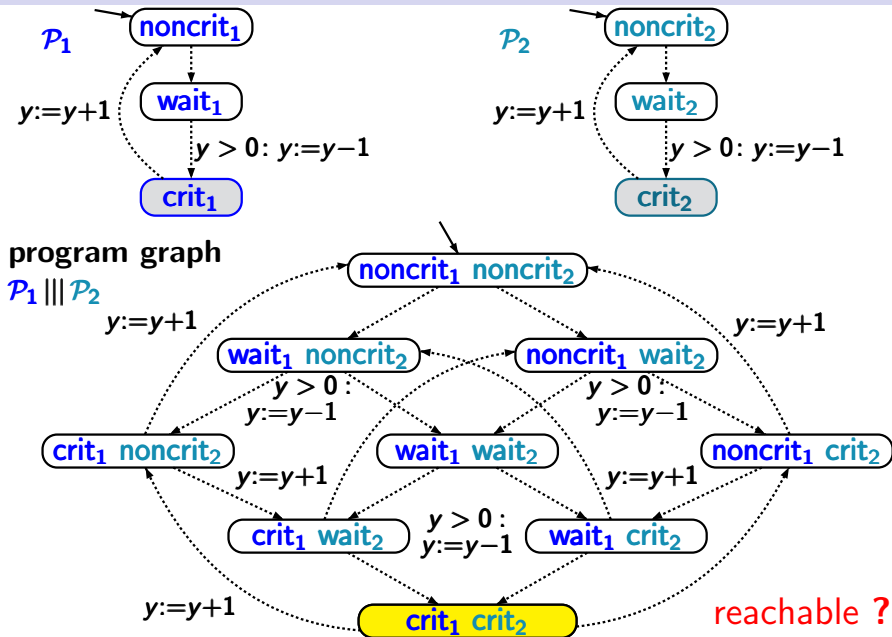
# Mutual exclusion with semaphore

PC2.2-10



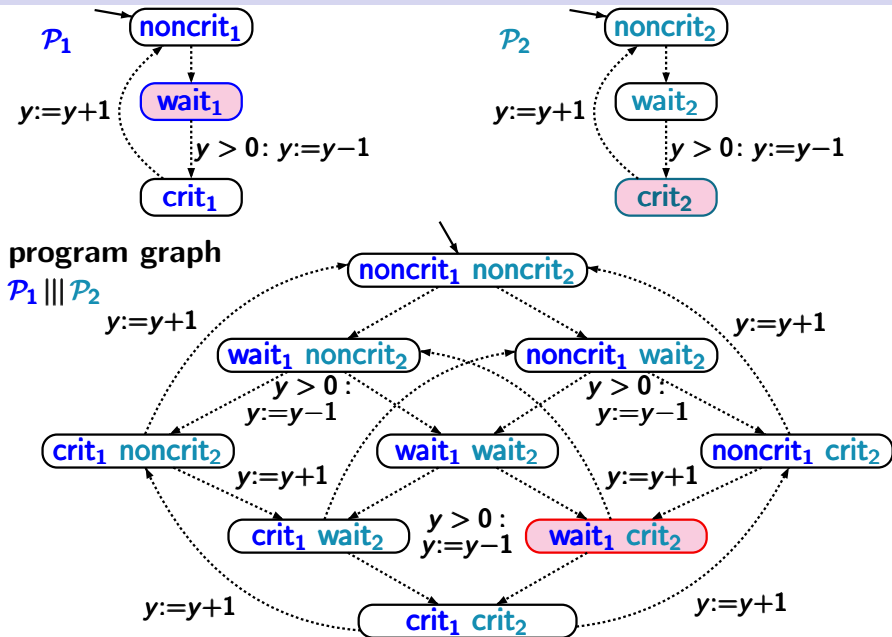
# Mutual exclusion with semaphore

PC2.2-10



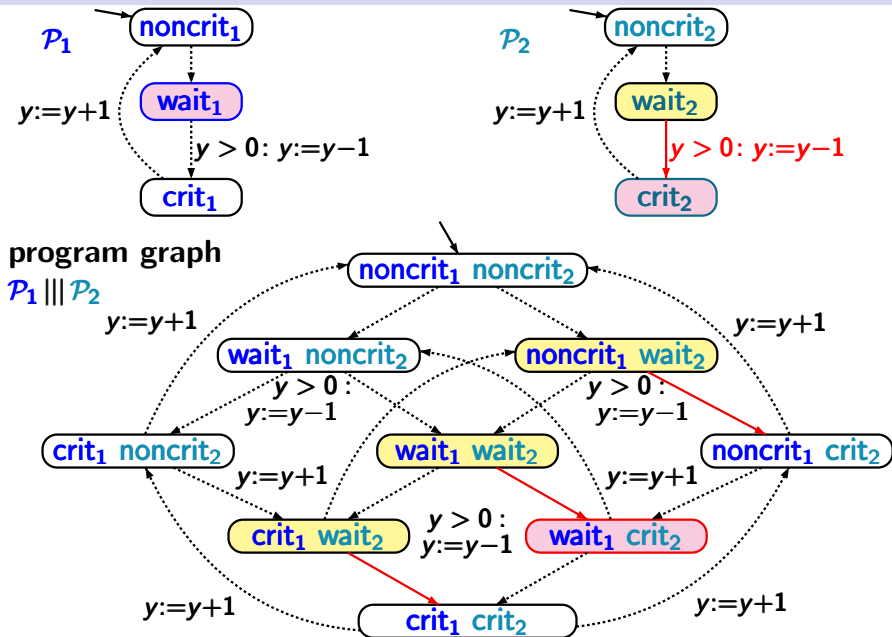
# Mutual exclusion with semaphore

PC2.2-10



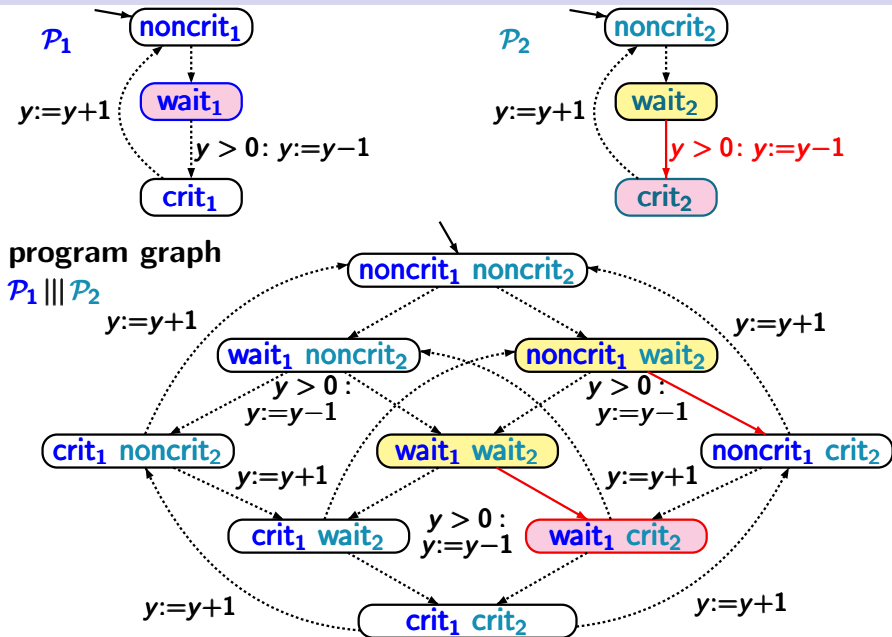
# Mutual exclusion with semaphore

PC2.2-10



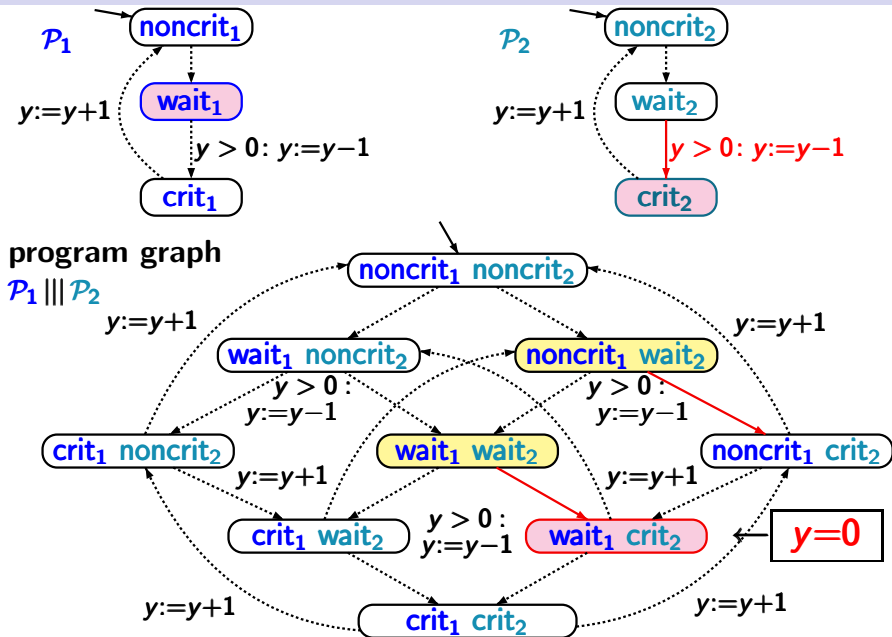
# Mutual exclusion with semaphore

PC2.2-10



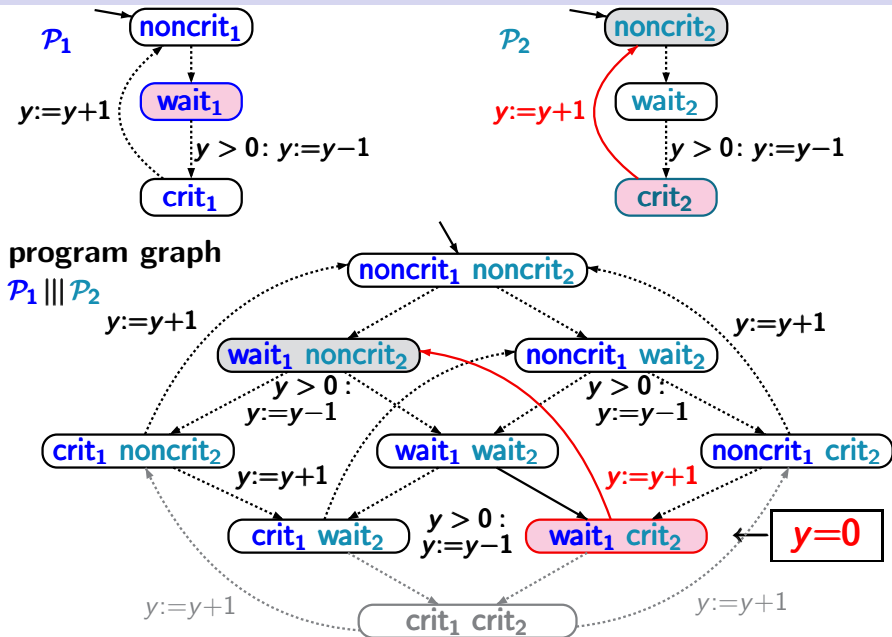
# Mutual exclusion with semaphore

PC2.2-10



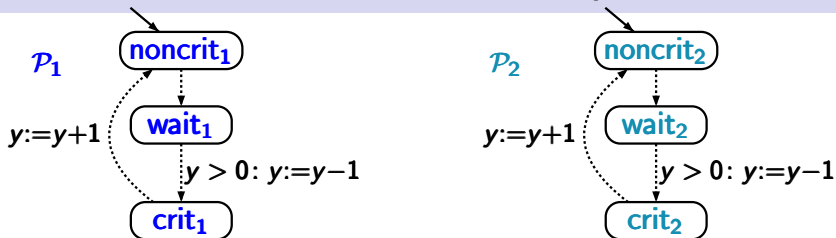
# Mutual exclusion with semaphore

PC2.2-10

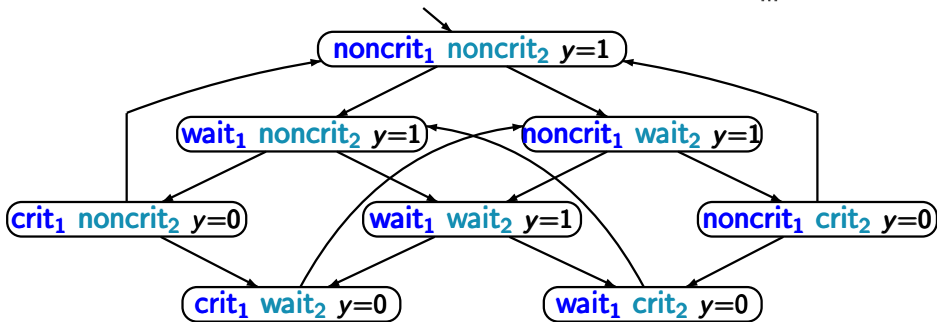


# TS for mutual exclusion with semaphore

PC2.2-11



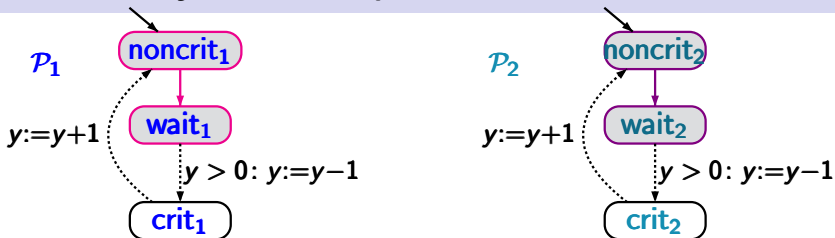
reachable fragment of the transition system  $\mathcal{T}_{\mathcal{P}_1 \parallel \mathcal{P}_2}$



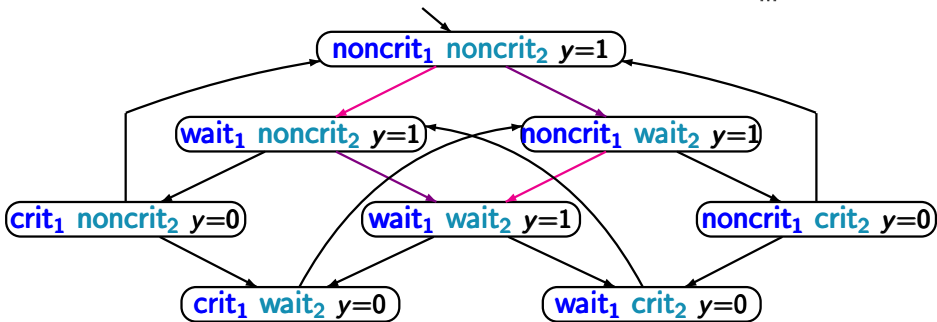


# Concurrency of the request actions

PC2.2-11

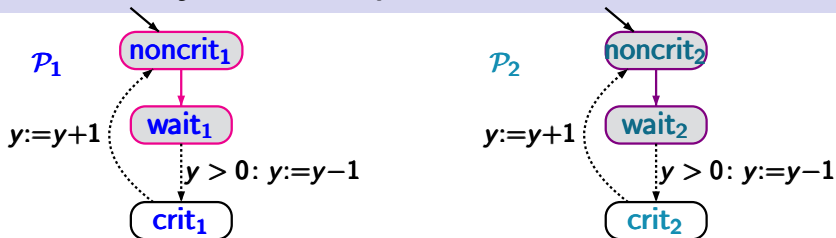


reachable fragment of the transition system  $\mathcal{T}_{\mathcal{P}_1 \parallel \mathcal{P}_2}$

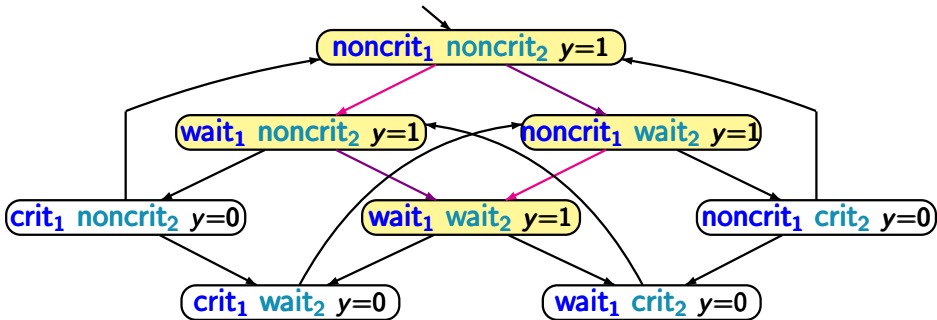


# Concurrency of the request actions

PC2.2-11

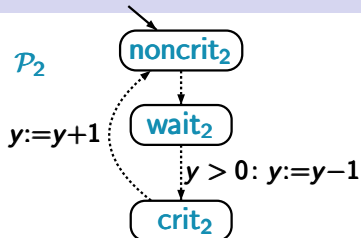
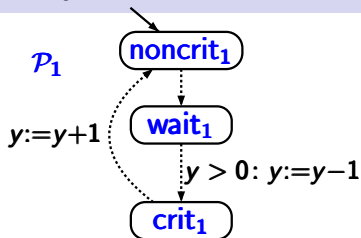


**interleaving** of the independent request actions

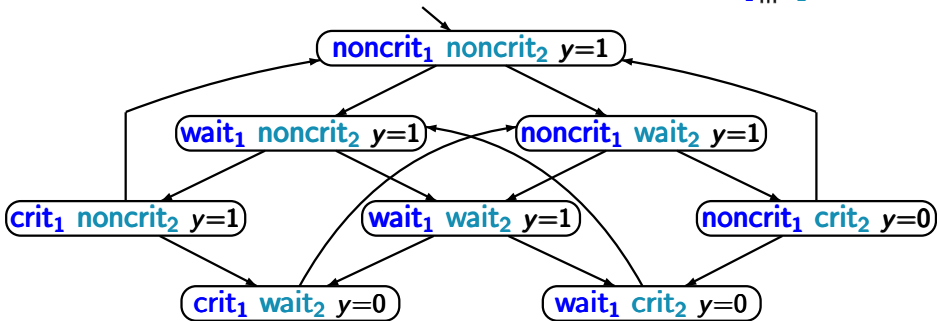


# Competition

PC2.2-11A

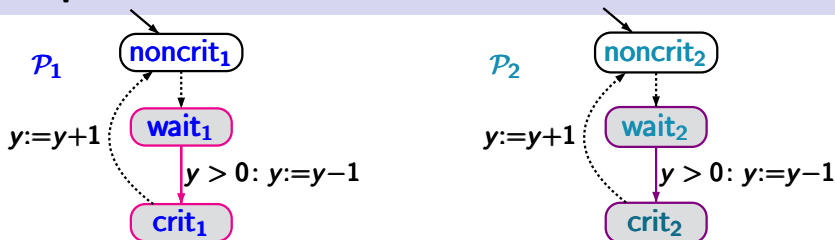


reachable fragment of the transition system  $\mathcal{T}_{\mathcal{P}_1 \parallel \mathcal{P}_2}$

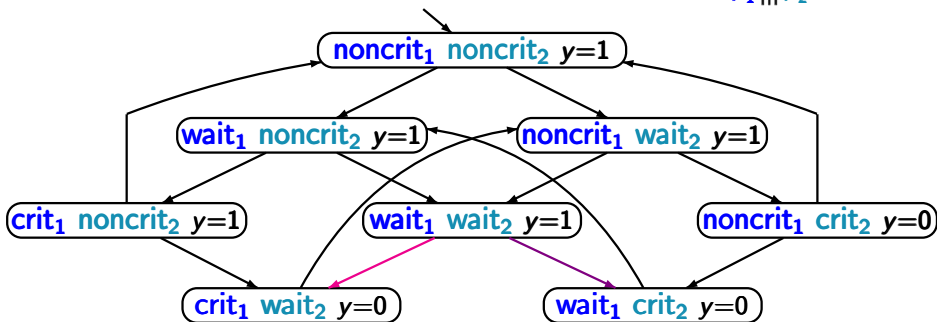


# Competition

PC2.2-11A

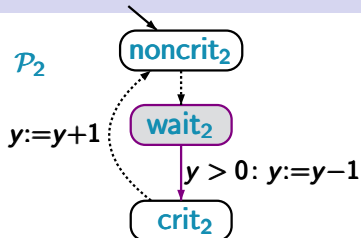
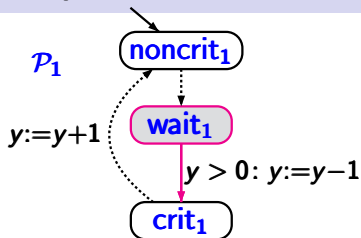


reachable fragment of the transition system  $\mathcal{T}_{\mathcal{P}_1 \parallel \mathcal{P}_2}$

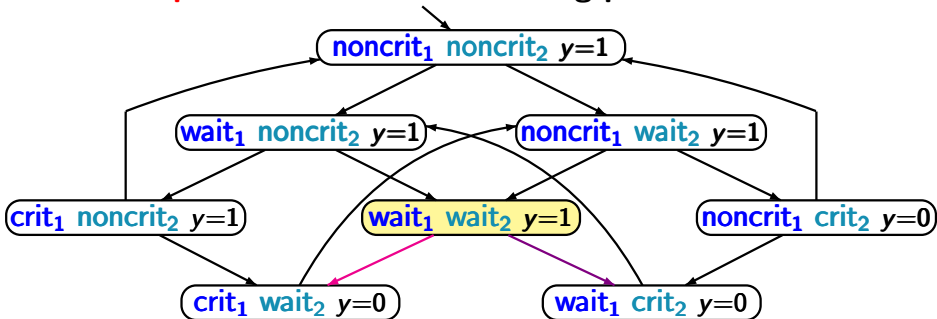


# Competition

PC2.2-11A



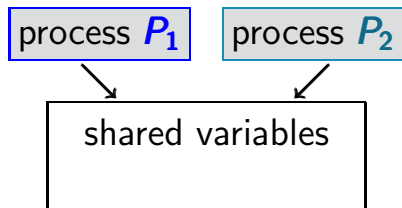
... **competition** between the waiting processes ...





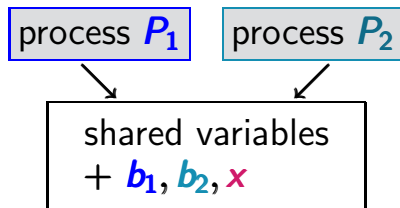
# Peterson algorithm for mutual exclusion

PC2.2-12



# Peterson algorithm for mutual exclusion

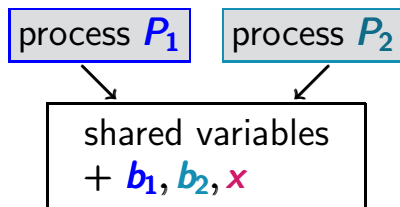
PC2.2-12



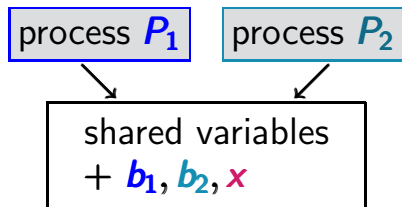


# Peterson algorithm for mutual exclusion

PC2.2-12



$b_1, b_2$  Boolean variables,  $x \in \{1, 2\}$

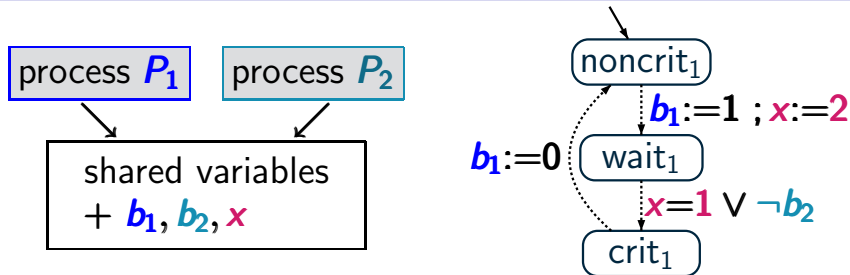


$b_1, b_2$  Boolean variables,  $x \in \{1, 2\}$

```
LOOP FOREVER                                (* protocol for  $P_1$  *)  
    noncritical actions;  
     $b_1 := 1$  ;  $x := 2$ ;  
    AWAIT  $x = 1 \vee \neg b_2$  DO critical section OD  
     $b_1 := 0$   
END LOOP
```

# Peterson algorithm for mutual exclusion

PC2.2-12

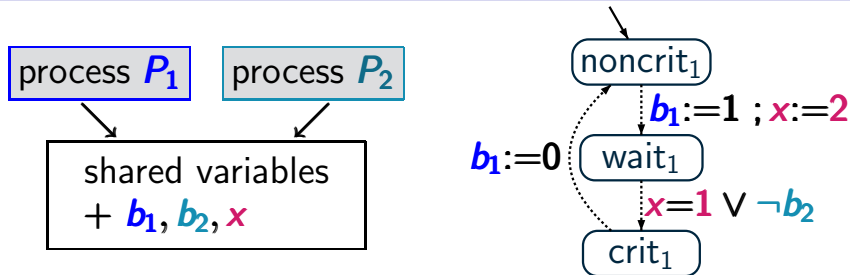


$b_1, b_2$  Boolean variables,  $x \in \{1, 2\}$

```
LOOP FOREVER                                (* protocol for  $P_1$  *)  
    noncritical actions;  
     $b_1 := 1$  ;  $x := 2$ ;  
    AWAIT  $x = 1 \vee \neg b_2$  DO critical section OD  
     $b_1 := 0$   
END LOOP
```

# Peterson algorithm for mutual exclusion

PC2.2-12

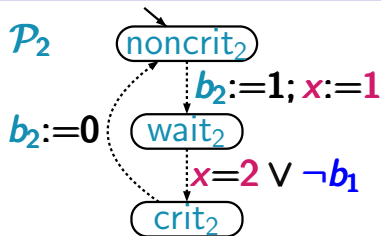
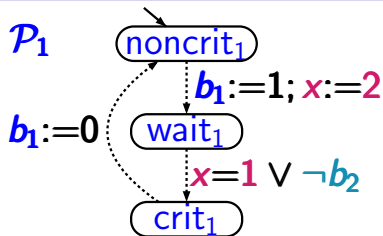


$b_1, b_2$  Boolean variables,  $x \in \{1, 2\}$

```
LOOP FOREVER                                (* protocol for  $P_1$  *)  
  noncritical actions;  
  atomic{  $b_1 := 1$  ;  $x := 2$  };  
  AWAIT  $x = 1 \vee \neg b_2$  DO critical section OD  
   $b_1 := 0$   
END LOOP
```

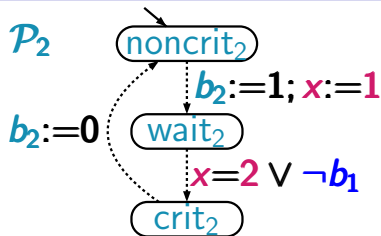
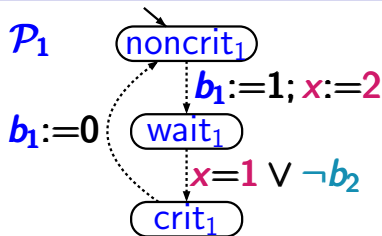
# Program graphs for Peterson algorithm

PC2.2-13



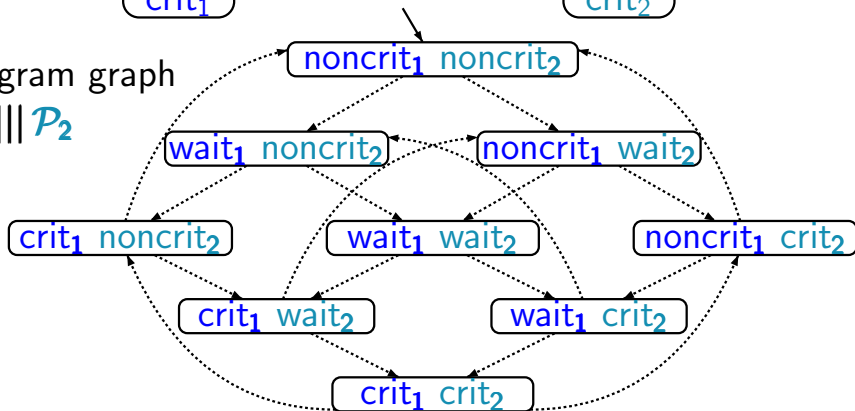
# Program graphs for Peterson algorithm

PC2.2-13



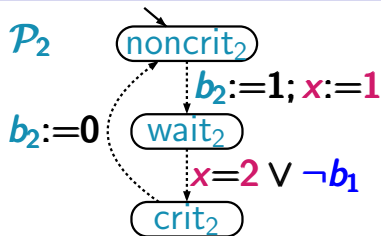
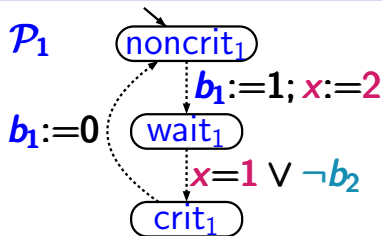
program graph

$\mathcal{P}_1 \parallel \mathcal{P}_2$



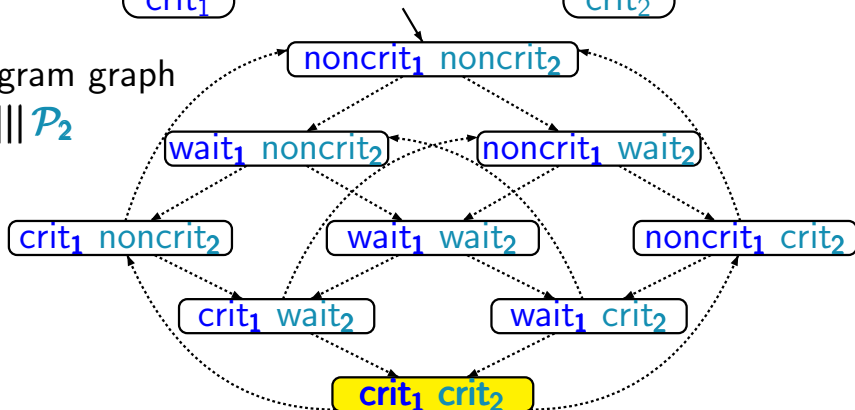
# Program graphs for Peterson algorithm

PC2.2-13



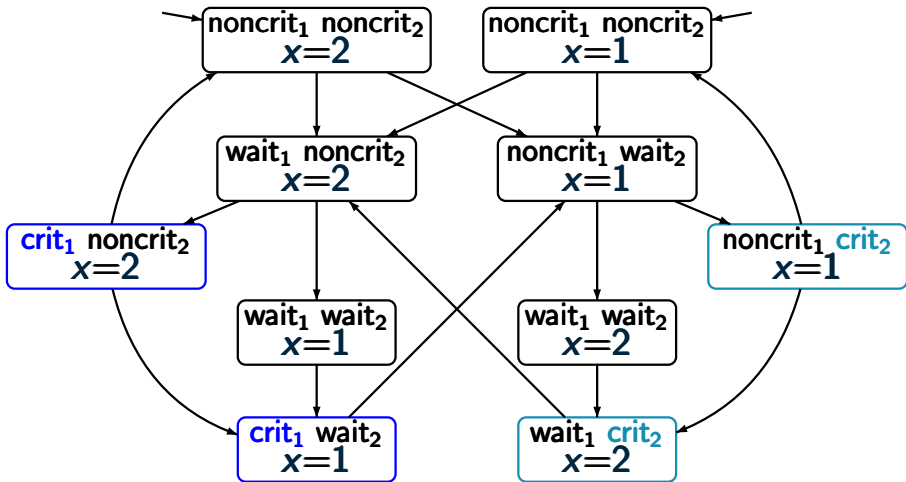
program graph

$\mathcal{P}_1 \parallel \mathcal{P}_2$



# TS for the Peterson algorithm

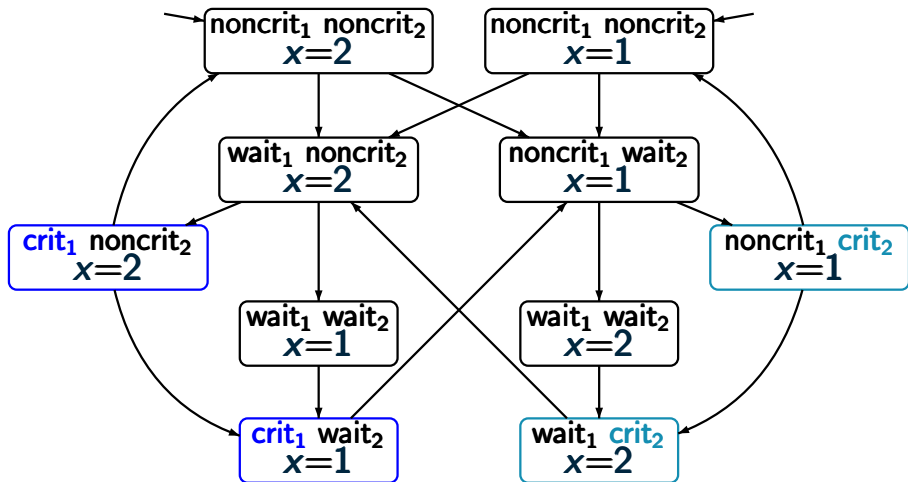
PC2.2-14





# TS for the Peterson algorithm

PC2.2-14

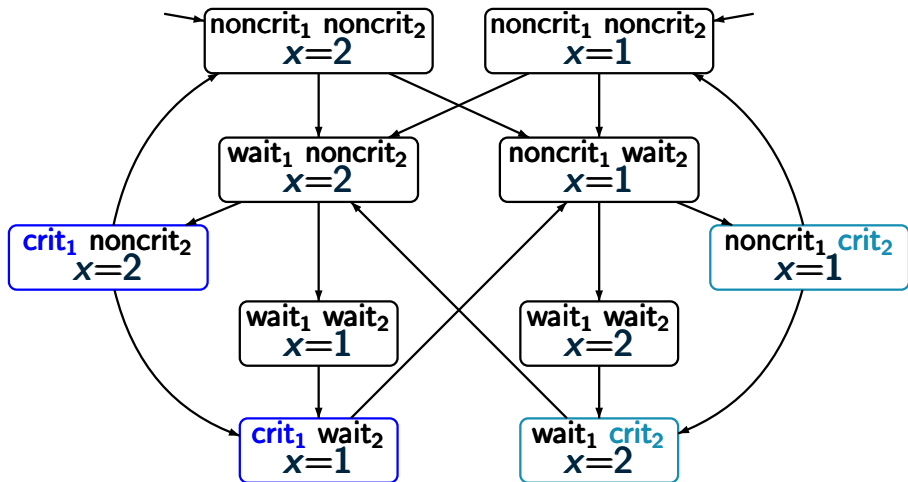


value of  $b_1$  is given by  $\text{wait}_1 \vee \text{crit}_1$

value of  $b_2$  is given by  $\text{wait}_2 \vee \text{crit}_2$

# TS for the Peterson algorithm

PC2.2-14



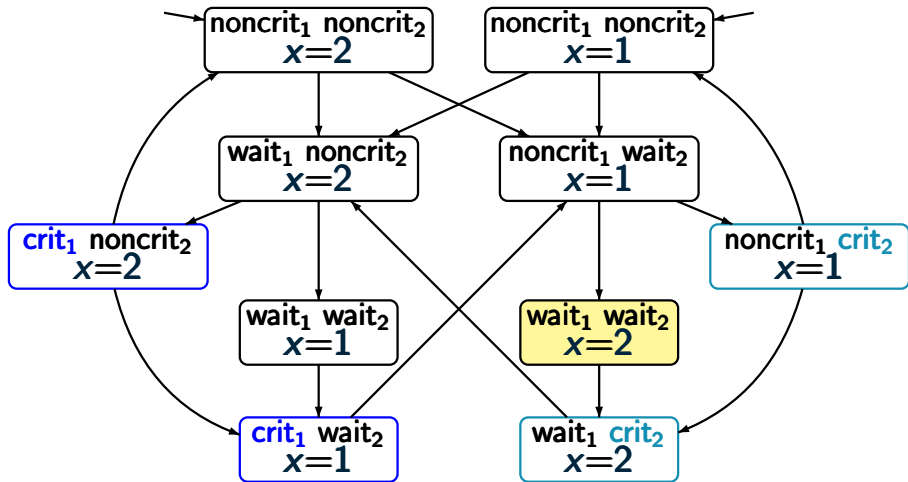
value of  $b_1$  is given by  $\text{wait}_1 \vee \text{crit}_1$

value of  $b_2$  is given by  $\text{wait}_2 \vee \text{crit}_2$

+ unreachable  
states

# TS for the Peterson algorithm

PC2.2-14



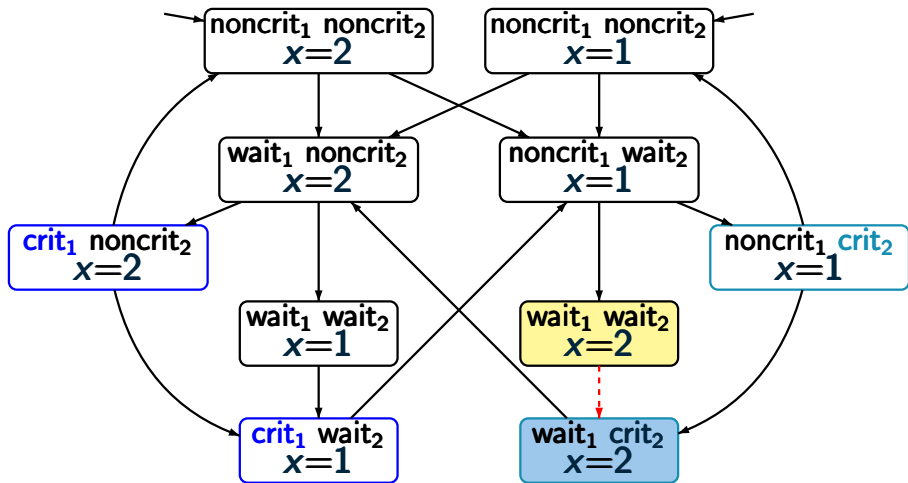
value of  $b_1$  is given by **wait<sub>1</sub>**  $\vee$  **crit<sub>1</sub>**

value of  $b_2$  is given by **wait<sub>2</sub>**  $\vee$  **crit<sub>2</sub>**

+ unreachable  
states

# TS for the Peterson algorithm

PC2.2-14



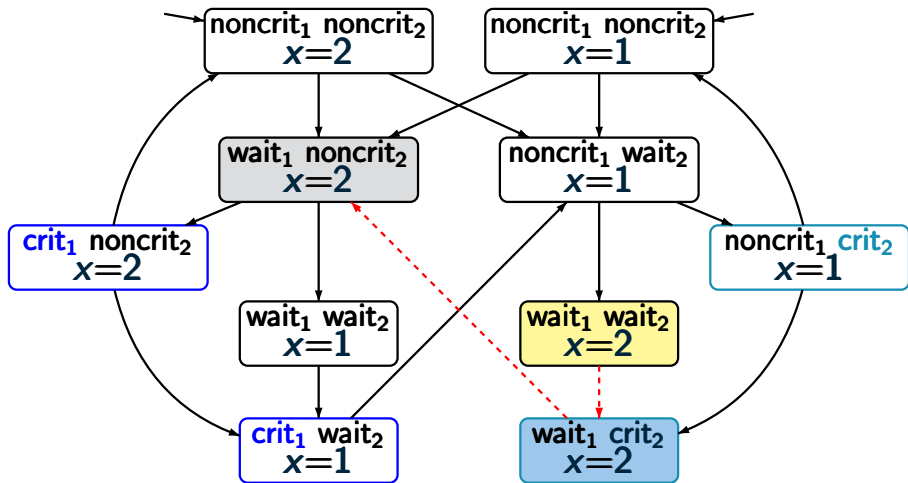
value of  $b_1$  is given by  $\text{wait}_1 \vee \text{crit}_1$

value of  $b_2$  is given by  $\text{wait}_2 \vee \text{crit}_2$

+ unreachable  
states

# TS for the Peterson algorithm

PC2.2-14



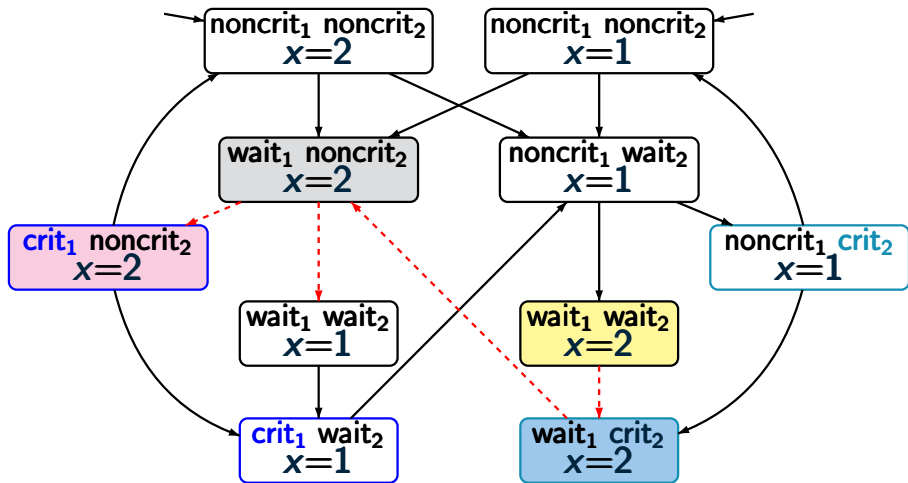
value of  $b_1$  is given by  $\text{wait}_1 \vee \text{crit}_1$

value of  $b_2$  is given by  $\text{wait}_2 \vee \text{crit}_2$

+ unreachable  
states

# TS for the Peterson algorithm

PC2.2-14



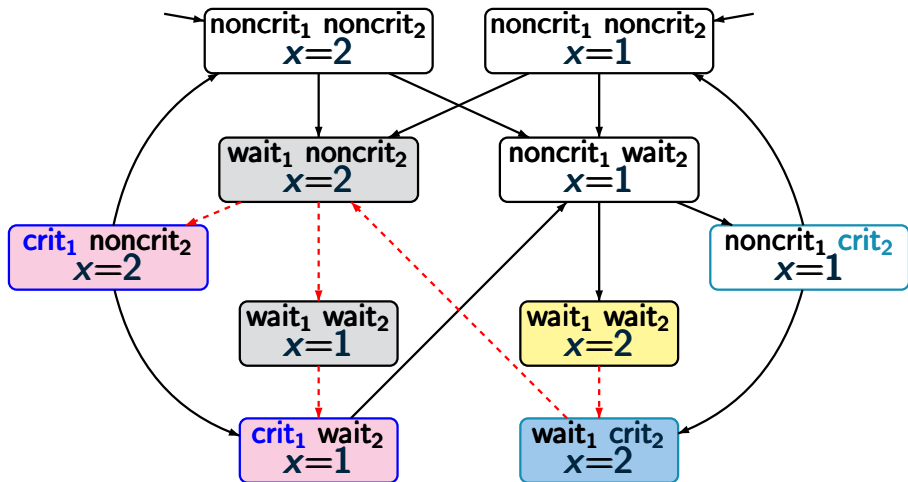
value of  $b_1$  is given by  $wait_1 \vee crit_1$

value of  $b_2$  is given by  $wait_2 \vee crit_2$

+ unreachable  
states

# TS for the Peterson algorithm

PC2.2-14



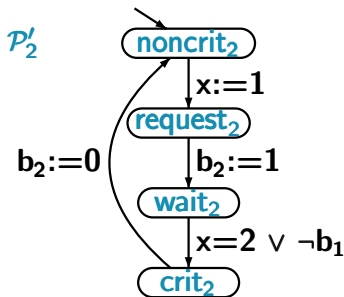
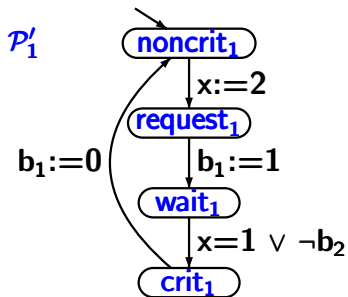
value of  $b_1$  is given by  $\text{wait}_1 \vee \text{crit}_1$

value of  $b_2$  is given by  $\text{wait}_2 \vee \text{crit}_2$

+ unreachable  
states

# Variant of Peterson algorithm

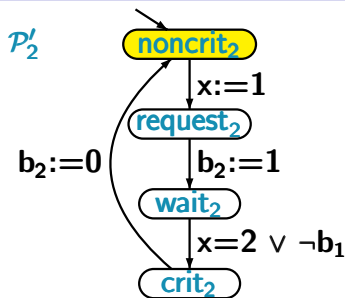
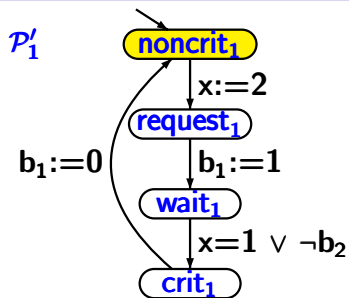
PC2.2-15





# Variant of Peterson algorithm

PC2.2-15

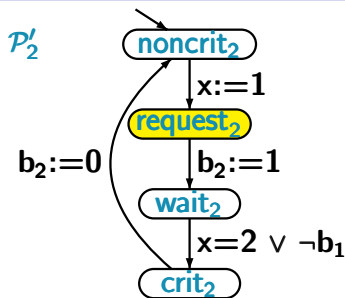
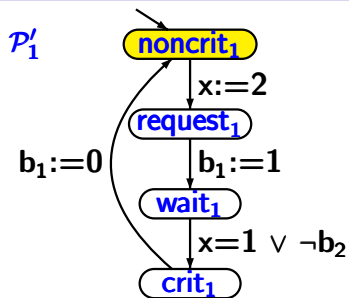


possible executions

**noncrit<sub>1</sub>**   **noncrit<sub>2</sub>**    $x=1$     $\neg b_1$     $\neg b_2$

# Variant of Peterson algorithm

PC2.2-15

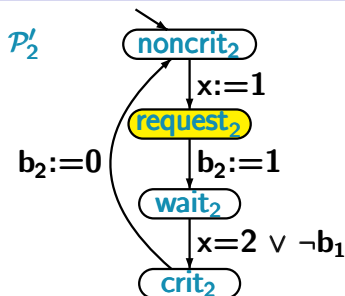
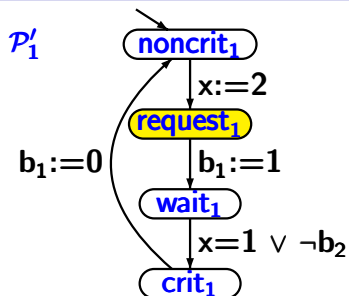


possible executions

noncrit <sub>1</sub>	noncrit <sub>2</sub>	x=1	¬b <sub>1</sub>	¬b <sub>2</sub>
noncrit <sub>1</sub>	request <sub>2</sub>	x=1	¬b <sub>1</sub>	¬b <sub>2</sub>

# Variant of Peterson algorithm

PC2.2-15

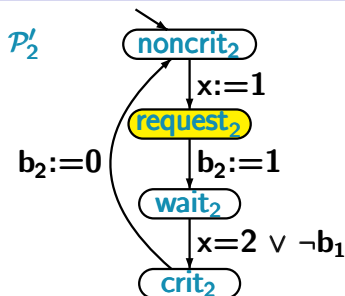
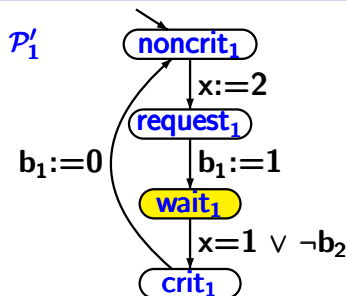


possible executions

noncrit <sub>1</sub>	noncrit <sub>2</sub>	x=1	¬b <sub>1</sub>	¬b <sub>2</sub>
noncrit <sub>1</sub>	request <sub>2</sub>	x=1	¬b <sub>1</sub>	¬b <sub>2</sub>
request <sub>1</sub>	request <sub>2</sub>	x=2	¬b <sub>1</sub>	¬b <sub>2</sub>

# Variant of Peterson algorithm

PC2.2-15

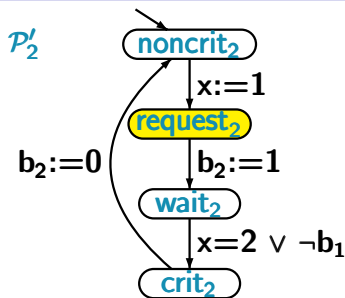
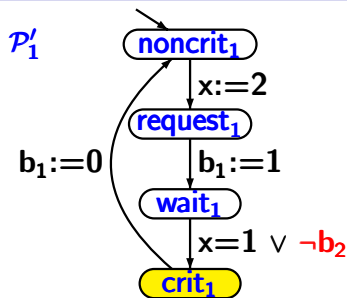


possible executions

noncrit <sub>1</sub>	noncrit <sub>2</sub>	x=1	¬b <sub>1</sub>	¬b <sub>2</sub>
noncrit <sub>1</sub>	request <sub>2</sub>	x=1	¬b <sub>1</sub>	¬b <sub>2</sub>
request <sub>1</sub>	request <sub>2</sub>	x=2	¬b <sub>1</sub>	¬b <sub>2</sub>
wait <sub>1</sub>	request <sub>2</sub>	x=2	b <sub>1</sub>	¬b <sub>2</sub>

# Variant of Peterson algorithm

PC2.2-15

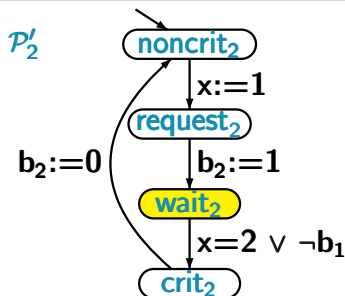
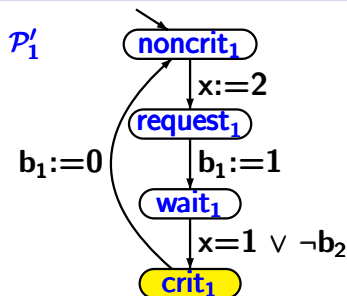


possible executions

noncrit <sub>1</sub>	noncrit <sub>2</sub>	x=1	¬b <sub>1</sub>	¬b <sub>2</sub>
noncrit <sub>1</sub>	request <sub>2</sub>	x=1	¬b <sub>1</sub>	¬b <sub>2</sub>
request <sub>1</sub>	request <sub>2</sub>	x=2	¬b <sub>1</sub>	¬b <sub>2</sub>
wait <sub>1</sub>	request <sub>2</sub>	x=2	b <sub>1</sub>	¬b <sub>2</sub>
crit <sub>1</sub>	request <sub>2</sub>	x=2	b <sub>1</sub>	¬b <sub>2</sub>

# Variant of Peterson algorithm

PC2.2-15

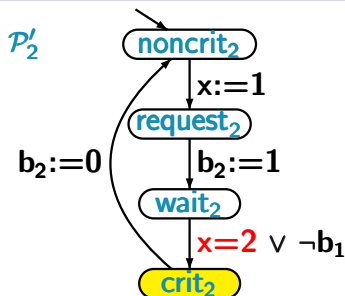
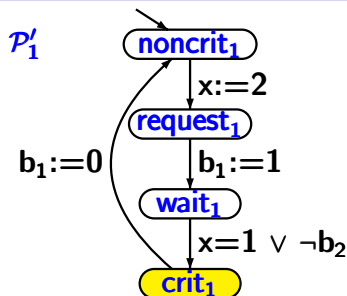


possible executions

noncrit <sub>1</sub>	noncrit <sub>2</sub>	x=1	¬b <sub>1</sub>	¬b <sub>2</sub>
noncrit <sub>1</sub>	request <sub>2</sub>	x=1	¬b <sub>1</sub>	¬b <sub>2</sub>
request <sub>1</sub>	request <sub>2</sub>	x=2	¬b <sub>1</sub>	¬b <sub>2</sub>
wait <sub>1</sub>	request <sub>2</sub>	x=2	b <sub>1</sub>	¬b <sub>2</sub>
crit <sub>1</sub>	request <sub>2</sub>	x=2	b <sub>1</sub>	¬b <sub>2</sub>
crit <sub>1</sub>	wait <sub>2</sub>	x=2	b <sub>1</sub>	b <sub>2</sub>

# Variant of Peterson algorithm

PC2.2-15

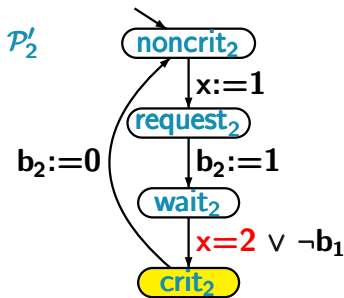
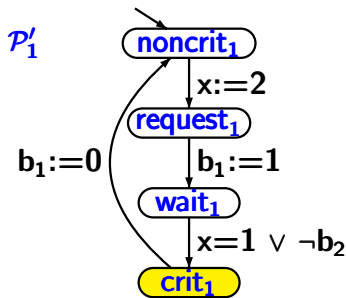


possible executions

noncrit <sub>1</sub>	noncrit <sub>2</sub>	x=1	¬b <sub>1</sub>	¬b <sub>2</sub>
noncrit <sub>1</sub>	request <sub>2</sub>	x=1	¬b <sub>1</sub>	¬b <sub>2</sub>
request <sub>1</sub>	request <sub>2</sub>	x=2	¬b <sub>1</sub>	¬b <sub>2</sub>
wait <sub>1</sub>	request <sub>2</sub>	x=2	b <sub>1</sub>	¬b <sub>2</sub>
crit <sub>1</sub>	request <sub>2</sub>	x=2	b <sub>1</sub>	¬b <sub>2</sub>
crit <sub>1</sub>	wait <sub>2</sub>	x=2	b <sub>1</sub>	b <sub>2</sub>
crit <sub>1</sub>	crit <sub>2</sub>	x=2	b <sub>1</sub>	b <sub>2</sub>

# Variant of Peterson algorithm **incorrect!**

PC2.2-15



possible executions

noncrit <sub>1</sub>	noncrit <sub>2</sub>	x=1	¬b <sub>1</sub>	¬b <sub>2</sub>
noncrit <sub>1</sub>	request <sub>2</sub>	x=1	¬b <sub>1</sub>	¬b <sub>2</sub>
request <sub>1</sub>	request <sub>2</sub>	x=2	¬b <sub>1</sub>	¬b <sub>2</sub>
wait <sub>1</sub>	request <sub>2</sub>	x=2	b <sub>1</sub>	¬b <sub>2</sub>
crit <sub>1</sub>	request <sub>2</sub>	x=2	b <sub>1</sub>	¬b <sub>2</sub>
crit <sub>1</sub>	wait <sub>2</sub>	x=2	b <sub>1</sub>	b <sub>2</sub>
crit <sub>1</sub>	crit <sub>2</sub>	x=2	b <sub>1</sub>	b <sub>2</sub>





- **true concurrency**: interleaving operator `|||` for TS  
(no communication, no dependencies)

- **true concurrency**: interleaving operator  $|||$  for **TS**  
(no communication, no dependencies)
- **communication via shared variables**
  - \* description of subsystems by **program graphs**
  - \* interleaving  $|||$  for program graphs
  - \* **TS** is obtained by “**unfolding**”

- **true concurrency**: interleaving operator  $|||$  for **TS**  
(no communication, no dependencies)
- **communication via shared variables**
  - \* description of subsystems by **program graphs**
  - \* interleaving  $|||$  for program graphs
  - \* **TS** is obtained by “**unfolding**”
- **synchronous message passing**

- **true concurrency**: interleaving operator  $|||$  for **TS**  
(no communication, no dependencies)
- **communication via shared variables**
  - \* description of subsystems by **program graphs**
  - \* interleaving  $|||$  for program graphs
  - \* **TS** is obtained by “**unfolding**”
- **synchronous message passing**
  - \* operator  $||_{\text{Syn}}$  for TS
  - \* interleaving for independent actions
  - \* synchronization over actions in **Syn**

- **true concurrency**: interleaving operator  $|||$  for **TS**  
(no communication, no dependencies)
- **communication via shared variables**
  - \* description of subsystems by **program graphs**
  - \* interleaving  $|||$  for program graphs
  - \* **TS** is obtained by “**unfolding**”
- **synchronous message passing**  $\longleftarrow$  data abstract
  - \* operator  $||_{\text{Syn}}$  for TS
  - \* interleaving for independent actions
  - \* synchronization over actions in **Syn**

- **true concurrency**: interleaving operator  $|||$  for **TS**  
(no communication, no dependencies)
- **communication via shared variables**
  - \* description of subsystems by **program graphs**
  - \* interleaving  $|||$  for program graphs
  - \* **TS** is obtained by “**unfolding**”
- **synchronous message passing**  $\leftarrow$  data abstract
  - \* operator  $||_{\text{Syn}}$  for TS
  - \* interleaving for independent actions
  - \* synchronization over actions in **Syn**
- **channel systems**  
communication via shared variables + via channels
- **synchronous product**





$\mathcal{T}_1 = (S_1, Act_1, \rightarrow_1, \dots)$ ,  $\mathcal{T}_2 = (S_2, Act_2, \rightarrow_2, \dots)$  TS

$Syn \subseteq Act_1 \cap Act_2$  set of synchronization actions

$\mathcal{T}_1 = (\mathcal{S}_1, \mathcal{A}ct_1, \rightarrow_1, \dots)$ ,  $\mathcal{T}_2 = (\mathcal{S}_2, \mathcal{A}ct_2, \rightarrow_2, \dots)$  TS

$Syn \subseteq \mathcal{A}ct_1 \cap \mathcal{A}ct_2$  set of synchronization actions

composite transition system:

$$\mathcal{T}_1 \parallel_{Syn} \mathcal{T}_2 = (\mathcal{S}_1 \times \mathcal{S}_2, \mathcal{A}ct_1 \cup \mathcal{A}ct_2, \rightarrow, \dots)$$

for modeling the concurrent execution of  $\mathcal{T}_1$  and  $\mathcal{T}_2$   
with **synchronization** over all actions in  $Syn$

$\mathcal{T}_1 = (\mathcal{S}_1, \text{Act}_1, \rightarrow_1, \dots)$ ,  $\mathcal{T}_2 = (\mathcal{S}_2, \text{Act}_2, \rightarrow_2, \dots)$  TS

$\text{Syn} \subseteq \text{Act}_1 \cap \text{Act}_2$  set of synchronization actions

composite transition system:

$$\mathcal{T}_1 \parallel_{\text{Syn}} \mathcal{T}_2 = (\mathcal{S}_1 \times \mathcal{S}_2, \text{Act}_1 \cup \text{Act}_2, \rightarrow, \dots)$$

interleaving for all actions  $\alpha \in \text{Act}_i \setminus \text{Syn}$ :

$$\frac{s_1 \xrightarrow{\alpha}_1 s'_1}{\langle s_1, s_2 \rangle \xrightarrow{\alpha} \langle s'_1, s_2 \rangle} \qquad \frac{s_2 \xrightarrow{\alpha}_2 s'_2}{\langle s_1, s_2 \rangle \xrightarrow{\alpha} \langle s_1, s'_2 \rangle}$$

$\mathcal{T}_1 = (\mathcal{S}_1, \text{Act}_1, \rightarrow_1, \dots)$ ,  $\mathcal{T}_2 = (\mathcal{S}_2, \text{Act}_2, \rightarrow_2, \dots)$  TS

$\text{Syn} \subseteq \text{Act}_1 \cap \text{Act}_2$  set of synchronization actions

composite transition system:

$$\mathcal{T}_1 \parallel_{\text{Syn}} \mathcal{T}_2 = (\mathcal{S}_1 \times \mathcal{S}_2, \text{Act}_1 \cup \text{Act}_2, \rightarrow, \dots)$$

interleaving for all actions  $\alpha \in \text{Act}_i \setminus \text{Syn}$ :

$$\frac{s_1 \xrightarrow{\alpha}_1 s'_1}{\langle s_1, s_2 \rangle \xrightarrow{\alpha} \langle s'_1, s_2 \rangle} \qquad \frac{s_2 \xrightarrow{\alpha}_2 s'_2}{\langle s_1, s_2 \rangle \xrightarrow{\alpha} \langle s_1, s'_2 \rangle}$$

handshaking (rendezvous) for all  $\alpha \in \text{Syn}$ :

$\mathcal{T}_1 = (\mathcal{S}_1, \text{Act}_1, \rightarrow_1, \dots)$ ,  $\mathcal{T}_2 = (\mathcal{S}_2, \text{Act}_2, \rightarrow_2, \dots)$  TS

$\text{Syn} \subseteq \text{Act}_1 \cap \text{Act}_2$  set of synchronization actions

composite transition system:

$$\mathcal{T}_1 \parallel_{\text{Syn}} \mathcal{T}_2 = (\mathcal{S}_1 \times \mathcal{S}_2, \text{Act}_1 \cup \text{Act}_2, \rightarrow, \dots)$$

interleaving for all actions  $\alpha \in \text{Act}_i \setminus \text{Syn}$ :

$$\frac{s_1 \xrightarrow{\alpha}_1 s'_1}{\langle s_1, s_2 \rangle \xrightarrow{\alpha} \langle s'_1, s_2 \rangle} \qquad \frac{s_2 \xrightarrow{\alpha}_2 s'_2}{\langle s_1, s_2 \rangle \xrightarrow{\alpha} \langle s_1, s'_2 \rangle}$$

handshaking (rendezvous) for all  $\alpha \in \text{Syn}$ :

$$\frac{s_1 \xrightarrow{\alpha}_1 s'_1 \quad \wedge \quad s_2 \xrightarrow{\alpha}_2 s'_2}{\langle s_1, s_2 \rangle \xrightarrow{\alpha} \langle s'_1, s'_2 \rangle}$$

by synchronous message passing

by synchronous message passing using an arbiter

protocol for process  $P_i$

```
LOOP FOREVER DO
    noncritical actions
    request
    critical section
    release
    noncritical actions
OD
```



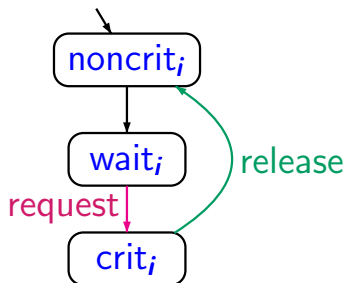
# Mutual exclusion with an arbiter

PC2.2-18

protocol for process  $P_i$

```
LOOP FOREVER DO
  noncritical actions
  request
  critical section
  release
  noncritical actions
OD
```

transition system  $\mathcal{T}_i$



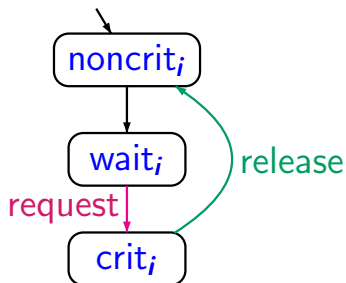
# Mutual exclusion with an arbiter

PC2.2-18

protocol for process  $P_i$

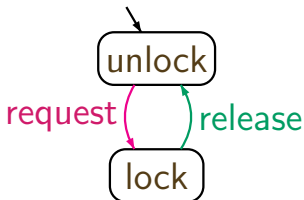
```
LOOP FOREVER DO
  noncritical actions
  request
  critical section
  release
  noncritical actions
OD
```

transition system  $\mathcal{T}_i$

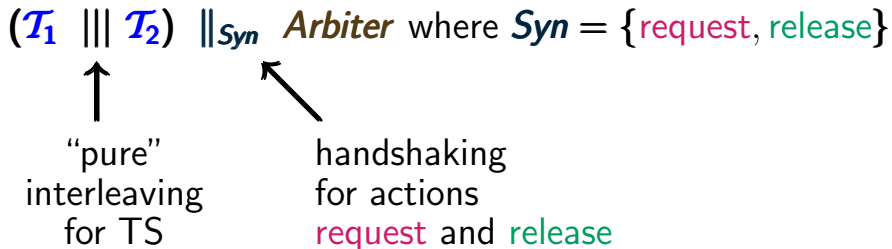


**Arbiter:**

selects nondeterministically  
a synchronization partner  
 $\mathcal{T}_1$  or  $\mathcal{T}_2$



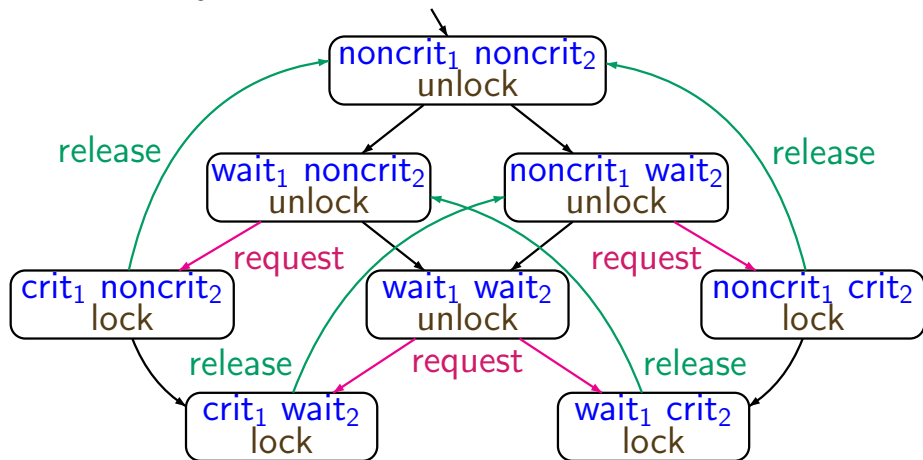
$(\mathcal{T}_1 \parallel \mathcal{T}_2) \parallel_{Syn} \textit{Arbiter}$  where  $Syn = \{\text{request}, \text{release}\}$



# Mutual exclusion with an arbiter

PC2.2-19

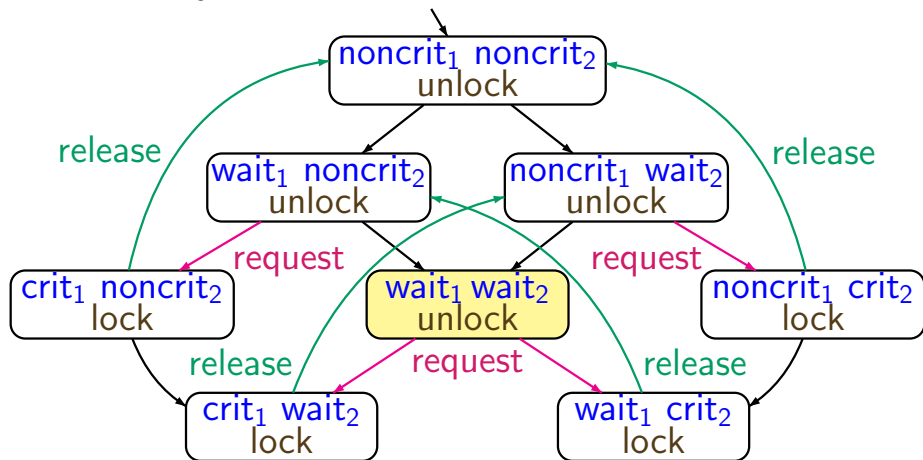
$(\mathcal{T}_1 ||| \mathcal{T}_2) ||_{\text{Syn}} \text{Arbiter}$  where  $\text{Syn} = \{\text{request}, \text{release}\}$



# Mutual exclusion with an arbiter

PC2.2-19

$(\mathcal{T}_1 ||| \mathcal{T}_2) ||_{\text{Syn}} \text{Arbiter}$  where  $\text{Syn} = \{\text{request}, \text{release}\}$



*nondeterministic choice: who enters the critical section?*

synchronization operator  $\parallel_{\text{Syn}}$  for  
three or more processes

$$\left. \begin{array}{l} \mathcal{T}_1 = (S_1, Act_1, \rightarrow_1, \dots) \\ \mathcal{T}_2 = (S_2, Act_2, \rightarrow_2, \dots) \\ \mathcal{T}_3 = (S_3, Act_3, \rightarrow_3, \dots) \\ \mathcal{T}_4 = (S_4, Act_4, \rightarrow_4, \dots) \\ \vdots \qquad \qquad \qquad \vdots \end{array} \right\} \text{transition systems}$$



$$\left. \begin{array}{l} \mathcal{T}_1 = (S_1, \text{Act}_1, \rightarrow_1, \dots) \\ \mathcal{T}_2 = (S_2, \text{Act}_2, \rightarrow_2, \dots) \\ \mathcal{T}_3 = (S_3, \text{Act}_3, \rightarrow_3, \dots) \\ \mathcal{T}_4 = (S_4, \text{Act}_4, \rightarrow_4, \dots) \\ \vdots \end{array} \right\} \text{ transition systems}$$

for  $\text{Syn} \subseteq \text{Act}_1 \cup \text{Act}_2 \cup \text{Act}_3 \cup \text{Act}_4 \cup \dots$

$$\begin{aligned} \mathcal{T}_1 \parallel_{\text{Syn}} \mathcal{T}_2 \parallel_{\text{Syn}} \mathcal{T}_3 \parallel_{\text{Syn}} \mathcal{T}_4 \parallel_{\text{Syn}} \dots &\stackrel{\text{def}}{=} \\ \left( \left( \left( \mathcal{T}_1 \parallel_{\text{Syn}} \mathcal{T}_2 \right) \parallel_{\text{Syn}} \mathcal{T}_3 \right) \parallel_{\text{Syn}} \mathcal{T}_4 \right) \parallel_{\text{Syn}} \dots \end{aligned}$$

$$\left. \begin{array}{l} \mathcal{T}_1 = (S_1, \text{Act}_1, \rightarrow_1, \dots) \\ \mathcal{T}_2 = (S_2, \text{Act}_2, \rightarrow_2, \dots) \\ \mathcal{T}_3 = (S_3, \text{Act}_3, \rightarrow_3, \dots) \\ \mathcal{T}_4 = (S_4, \text{Act}_4, \rightarrow_4, \dots) \\ \vdots \end{array} \right\} \text{ transition systems}$$

for  $\text{Syn} \subseteq \text{Act}_1 \cup \text{Act}_2 \cup \text{Act}_3 \cup \text{Act}_4 \cup \dots$

$$\boxed{\begin{array}{l} \mathcal{T}_1 \parallel_{\text{Syn}} \mathcal{T}_2 \parallel_{\text{Syn}} \mathcal{T}_3 \parallel_{\text{Syn}} \mathcal{T}_4 \parallel_{\text{Syn}} \dots \stackrel{\text{def}}{=} \\ \left( \left( \left( \mathcal{T}_1 \parallel_{\text{Syn}} \mathcal{T}_2 \right) \parallel_{\text{Syn}} \mathcal{T}_3 \right) \parallel_{\text{Syn}} \mathcal{T}_4 \right) \parallel_{\text{Syn}} \dots \end{array}}$$

or any other order of paranthesis

$$\left. \begin{array}{l} \mathcal{T}_1 = (S_1, \text{Act}_1, \rightarrow_1, \dots) \\ \mathcal{T}_2 = (S_2, \text{Act}_2, \rightarrow_2, \dots) \\ \mathcal{T}_3 = (S_3, \text{Act}_3, \rightarrow_3, \dots) \\ \mathcal{T}_4 = (S_4, \text{Act}_4, \rightarrow_4, \dots) \\ \vdots \end{array} \right\} \text{ transition systems}$$

for  $\text{Syn} \subseteq \text{Act}_1 \cup \text{Act}_2 \cup \text{Act}_3 \cup \text{Act}_4 \cup \dots$

$$\boxed{\begin{array}{l} \mathcal{T}_1 \parallel_{\text{Syn}} \mathcal{T}_2 \parallel_{\text{Syn}} \mathcal{T}_3 \parallel_{\text{Syn}} \mathcal{T}_4 \parallel_{\text{Syn}} \dots \stackrel{\text{def}}{=} \\ \left( ((\mathcal{T}_1 \parallel_{\text{Syn}} \mathcal{T}_2) \parallel_{\text{Syn}} \mathcal{T}_3) \parallel_{\text{Syn}} \mathcal{T}_4 \right) \parallel_{\text{Syn}} \dots \end{array}}$$

where, e.g.,  $\mathcal{T}_1 \parallel_{\text{Syn}} \mathcal{T}_2 \stackrel{\text{def}}{=} \mathcal{T}_1 \parallel_H \mathcal{T}_2$

with  $H = \text{Syn} \cap \text{Act}_1 \cap \text{Act}_2$

# Parallel operator $\parallel$

PC2.2-OP-PAR

$$\begin{aligned}\mathcal{T}_1 &= (S_1, \text{Act}_1, \rightarrow_1, \dots) \\ \mathcal{T}_2 &= (S_2, \text{Act}_2, \rightarrow_2, \dots) \\ \mathcal{T}_3 &= (S_3, \text{Act}_3, \rightarrow_3, \dots) \\ \mathcal{T}_4 &= (S_4, \text{Act}_4, \rightarrow_4, \dots) \\ &\vdots\end{aligned}$$

transition systems s.t.

$$\text{Act}_i \cap \text{Act}_j \cap \text{Act}_k = \emptyset$$

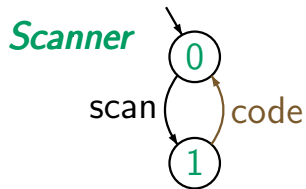
if  $i, j, k$  are pairwise  
distinct

$$\begin{aligned}\mathcal{T}_1 \parallel \mathcal{T}_2 \parallel \mathcal{T}_3 \parallel \mathcal{T}_4 \parallel \dots &\stackrel{\text{def}}{=} \\ (((\mathcal{T}_1 \parallel_{\text{Syn}_{1,2}} \mathcal{T}_2) \parallel_{\text{Syn}_{1,2,3}} \mathcal{T}_3) \parallel_{\text{Syn}_{1,2,3,4}} \mathcal{T}_4) \dots\end{aligned}$$

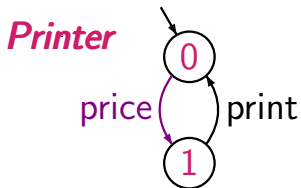
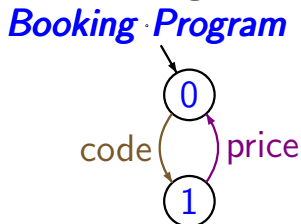
$$\begin{aligned}\text{where } \text{Syn}_{1,2} &= \text{Act}_1 \cap \text{Act}_2 \\ \text{Syn}_{1,2,3} &= (\text{Act}_1 \cup \text{Act}_2) \cap \text{Act}_3 \\ \text{Syn}_{1,2,3,4} &= (\text{Act}_1 \cup \text{Act}_2 \cup \text{Act}_3) \cap \text{Act}_4 \\ &\vdots\end{aligned}$$

# Booking system in supermarket

PC2.2-21A

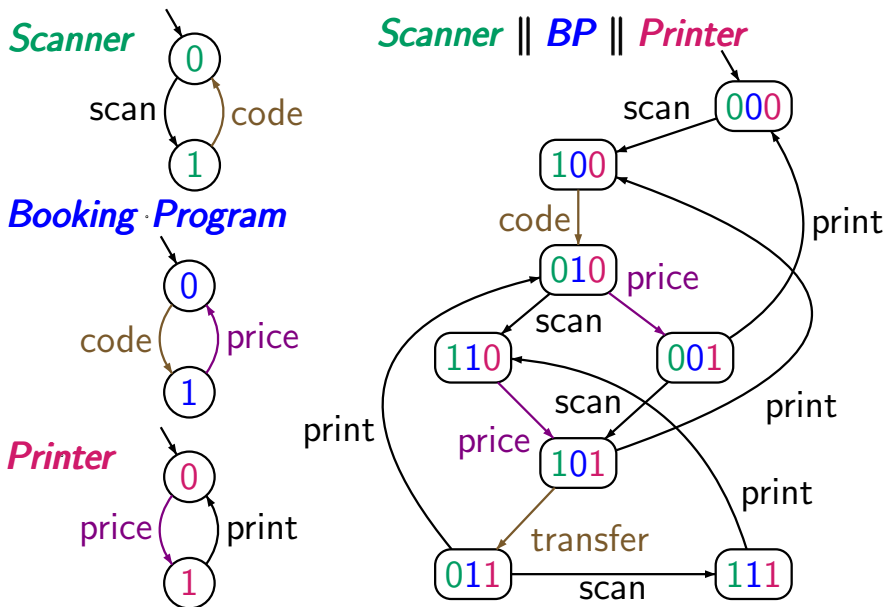


*Scanner* || *BP* || *Printer*



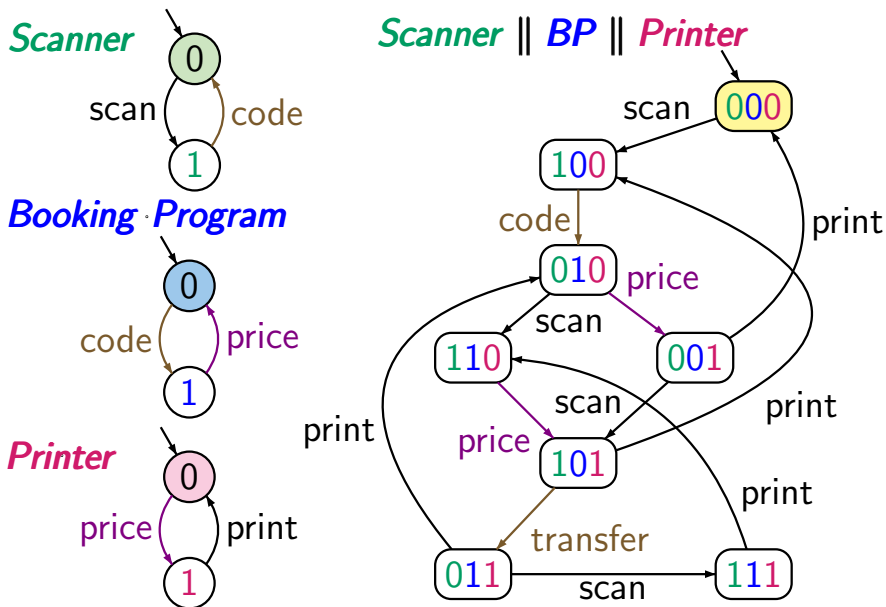
# Booking system in supermarket

PC2.2-21A



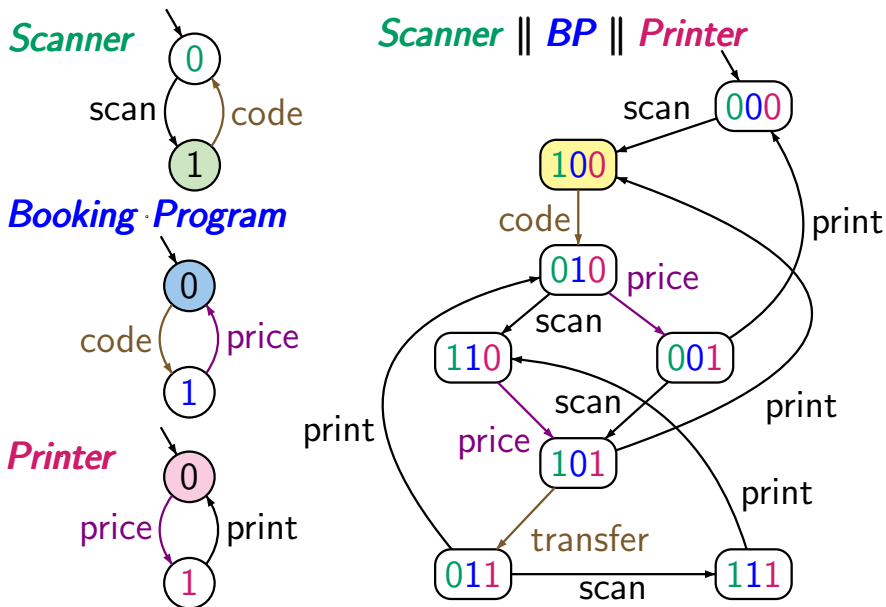
# Booking system in supermarket

PC2.2-21A



# Booking system in supermarket

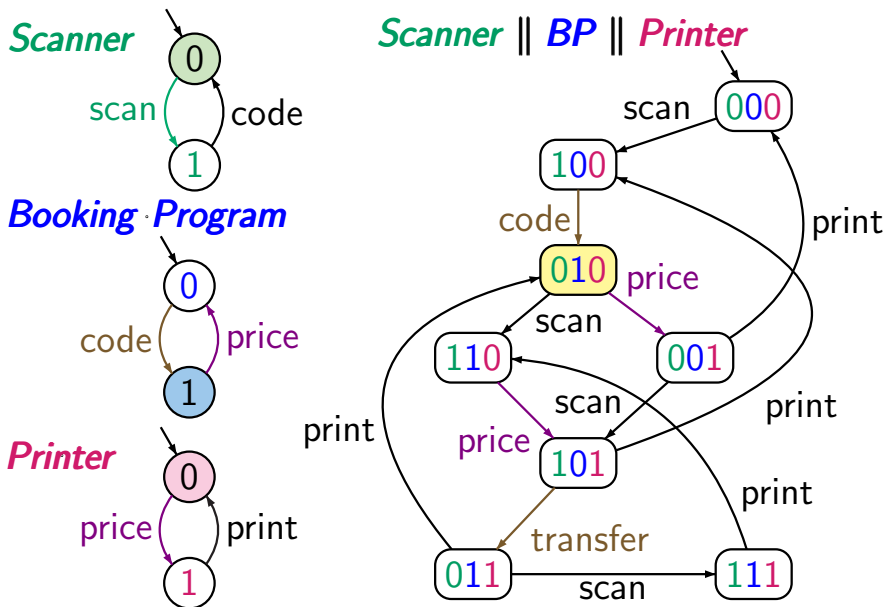
PC2.2-21A



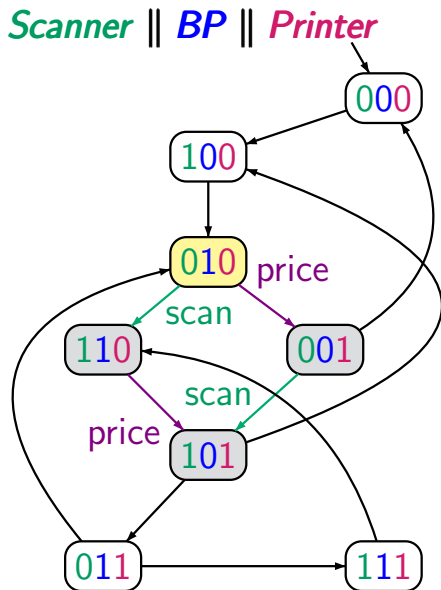


# Booking system in supermarket

PC2.2-21A

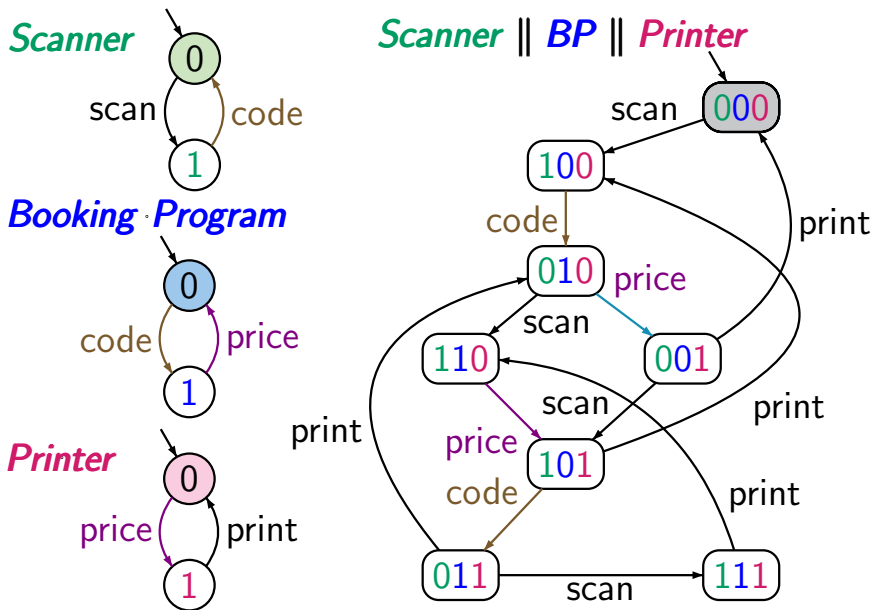


## PC2.2-21A



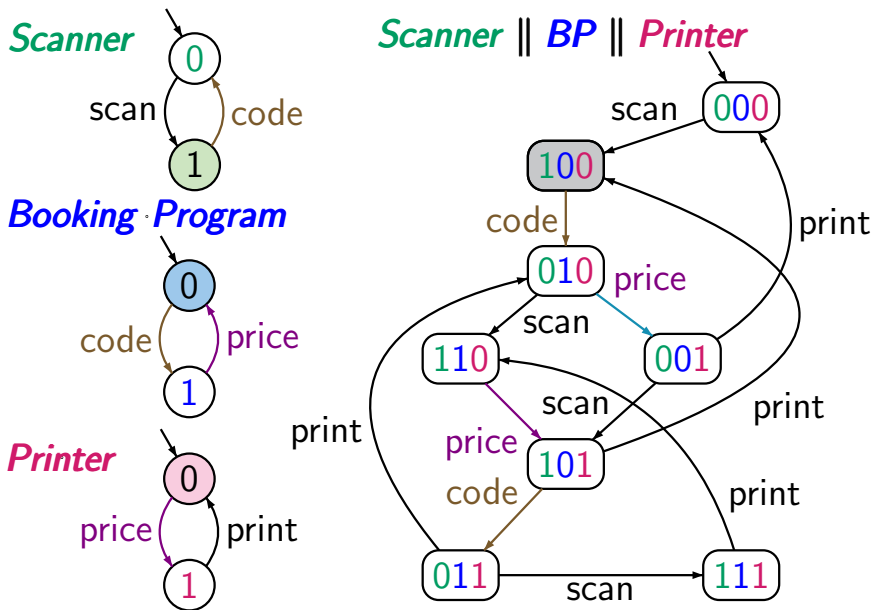
## Booking system in supermarket

PC2.2-21



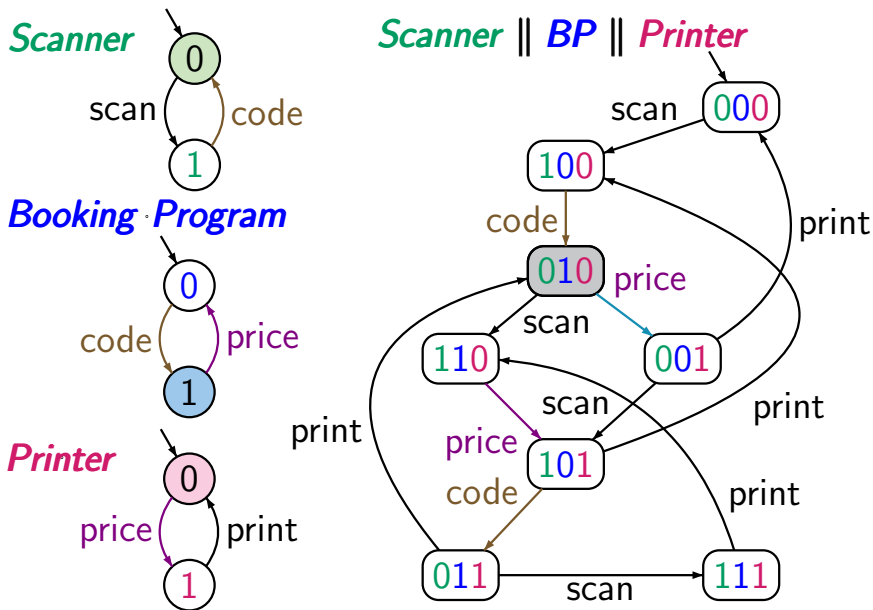
# Booking system in supermarket

PC2.2-21



# Booking system in supermarket

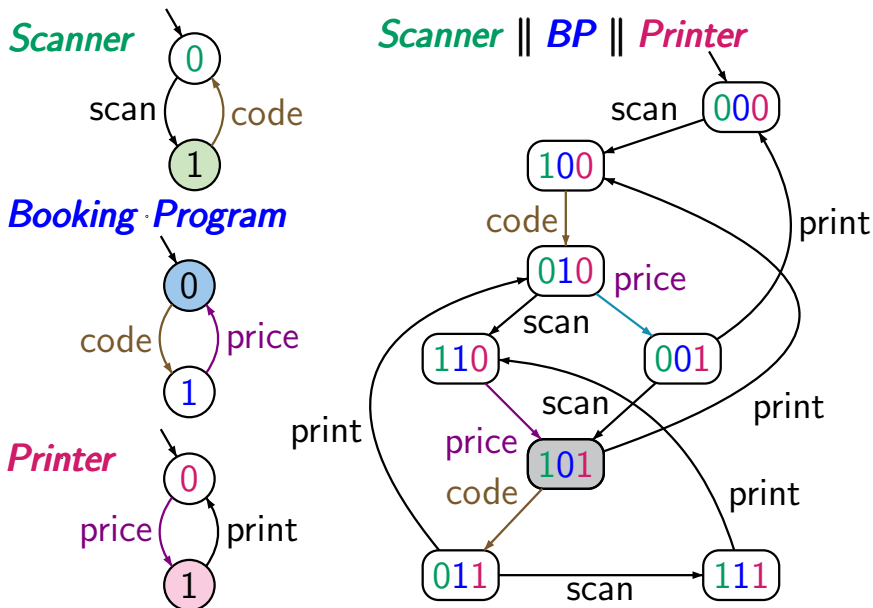
PC2.2-21





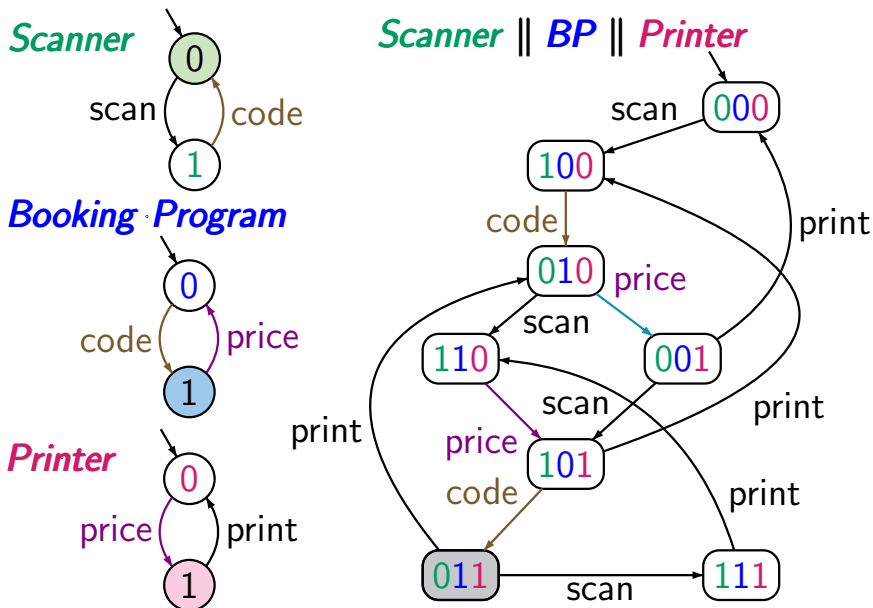
# Booking system in supermarket

PC2.2-21



# Booking system in supermarket

PC2.2-21





# Interleaving

PC2.2-21

