Decomposition of transition systems into sets of synchronizing Free-choice Petri Nets

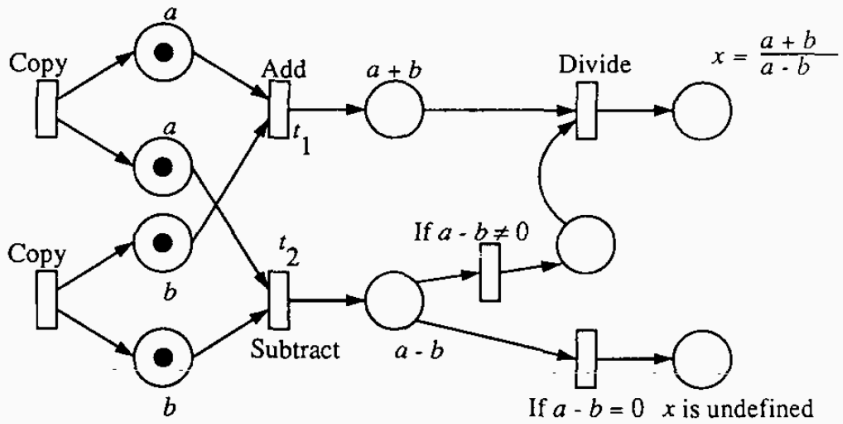
Viktor Teren[1], Jordi Cortadella[2] and Tiziano Villa[1]

[1]Università degli Studi di Verona, Verona, Italy
[2]Universitat Politècnica de Catalunya, Barcelona, Spain

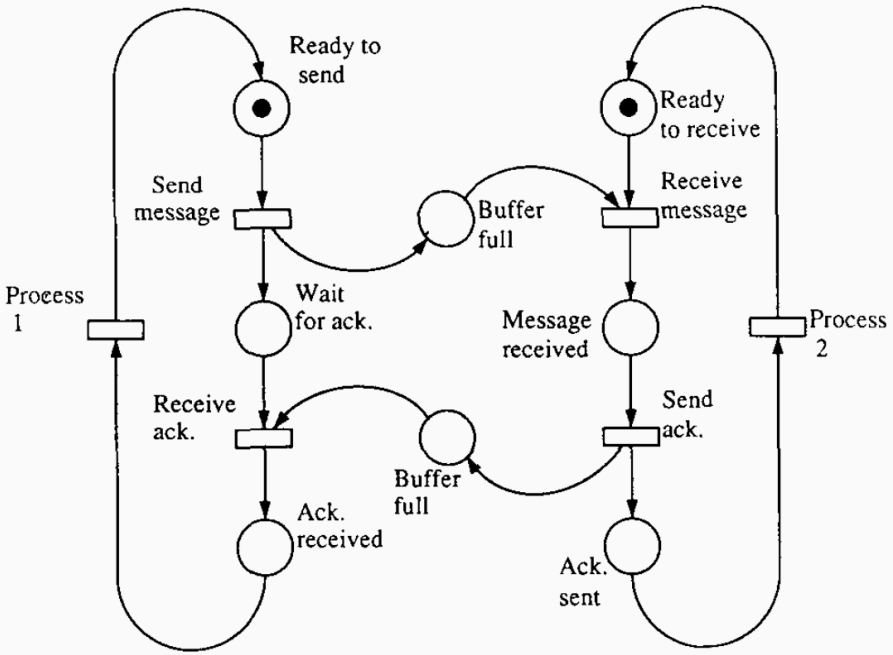
Outline

- Motivation
- Background theory
- Decomposition of Transition Systems
- Results
- Conclusions and future work

Motivation



Data flow

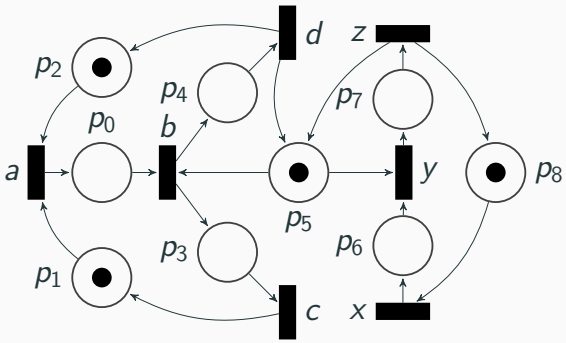


Communication protocol

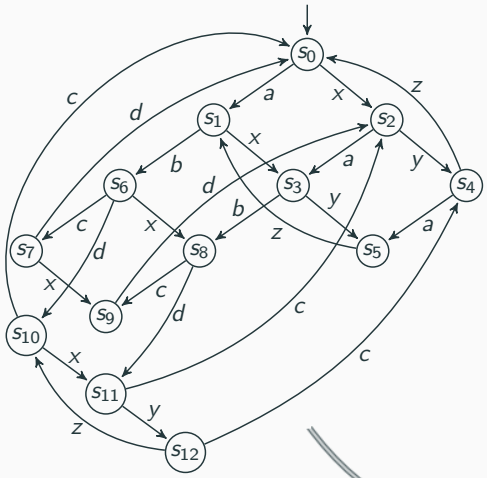
Murata, Tadao. "Petri nets: Properties, analysis and applications." Proceedings of the IEEE 77.4 (1989): 541-580.

Motivation

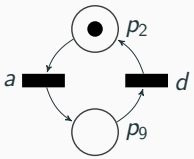
Monolithic Petri Net



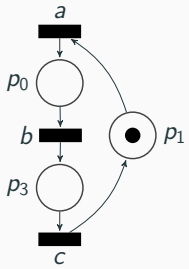
Transition System



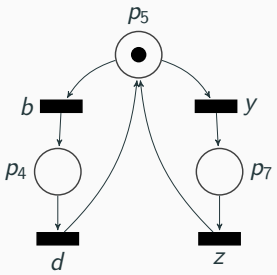
State Machines



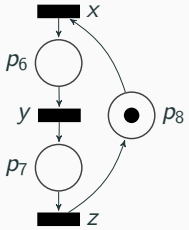
SM_1



SM_2



SM_3

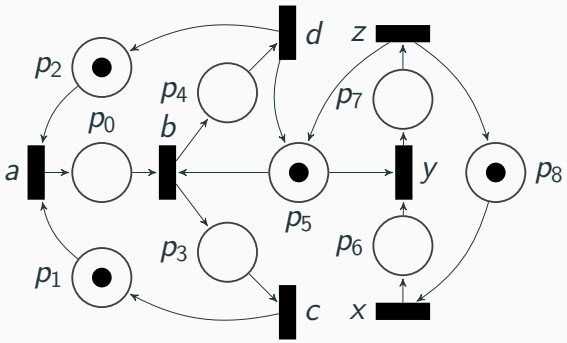


SM_4

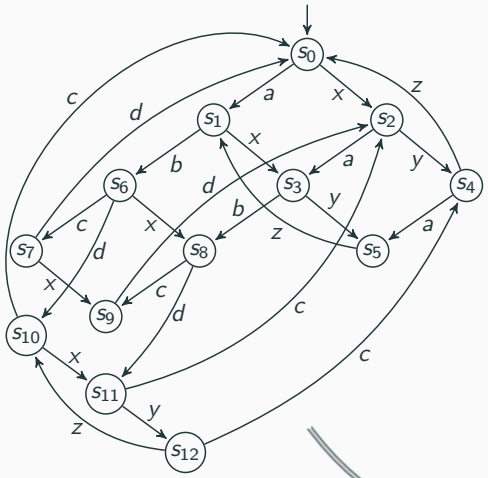
Synchronous product

Motivation

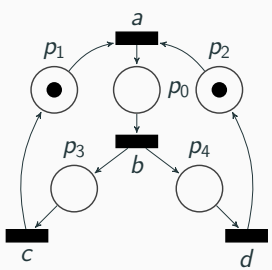
Monolithic Petri Net



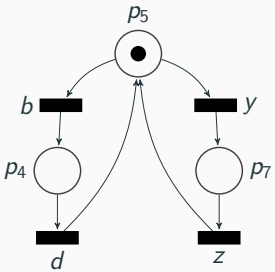
Transition System



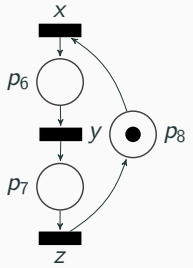
Free-choice Petri nets



$FCPN_1$



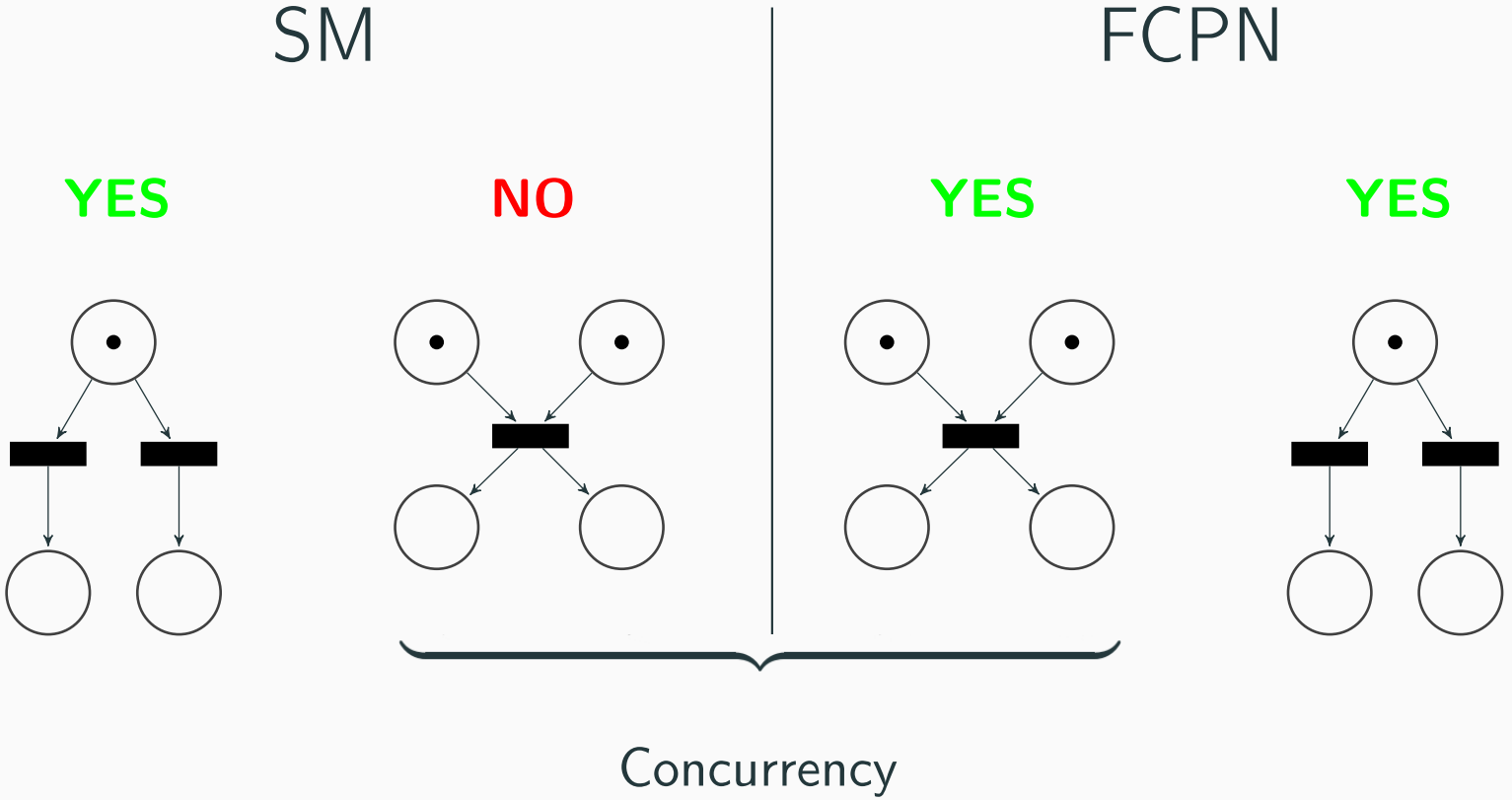
$FCPN_2$



$FCPN_3$

Synchronous product

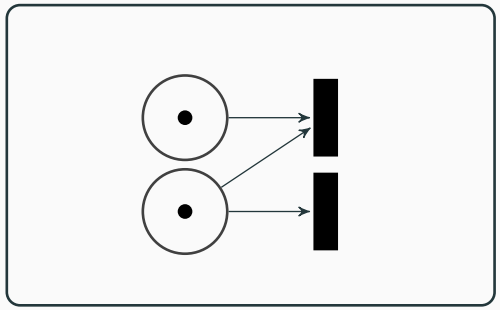
Why Free-choice Petri nets?



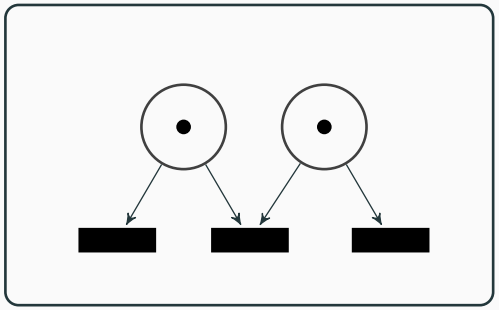
The **sequential** structure of State Machines is a **limitation**!!!

Why FCPNs and not any other subclass of PNs?

Structures **forbidden** in Free-choice Petri nets:



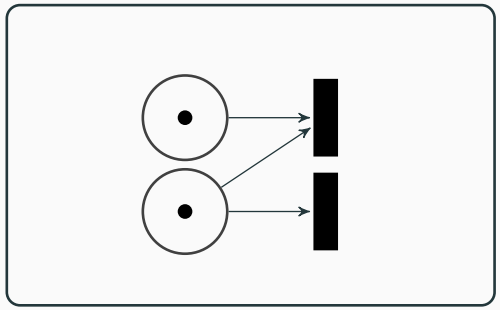
Asymmetric choice



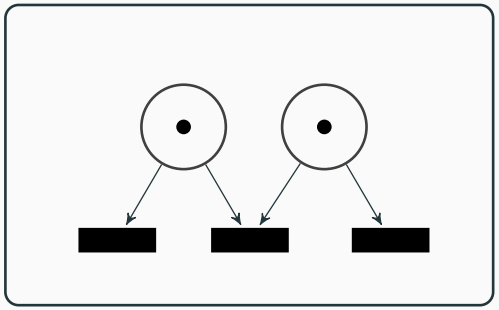
Confusion

Why FCPNs and not any other subclass of PNs?

Structures **forbidden** in Free-choice Petri nets:



Asymmetric choice



Confusion

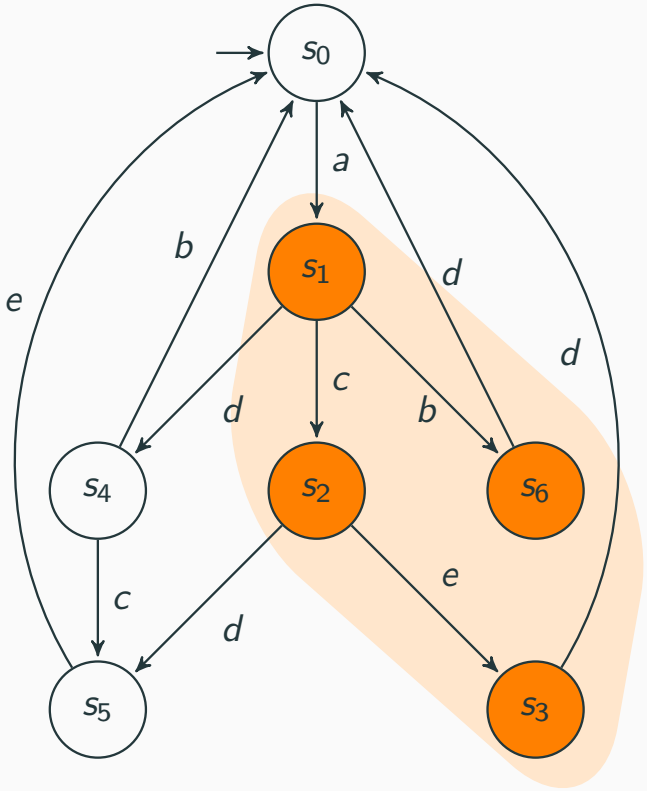
Why FCPNs:

- Simple structures
- Good visual representation
- Explicit concurrency
- Reduced complexity for some PN problems

Background theory

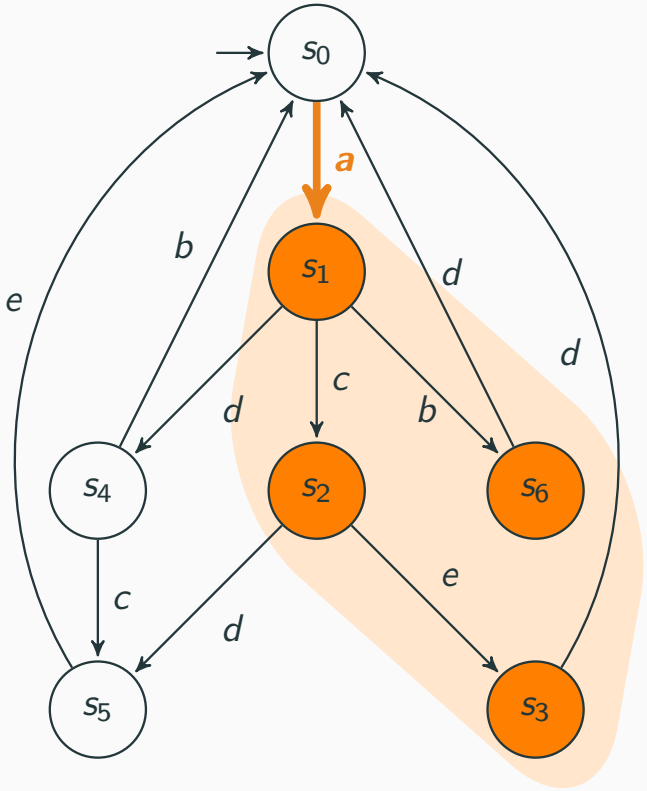
Regions

Region: subset of states in which events have the same *enter*/*exit*/*no_cross* behavior.



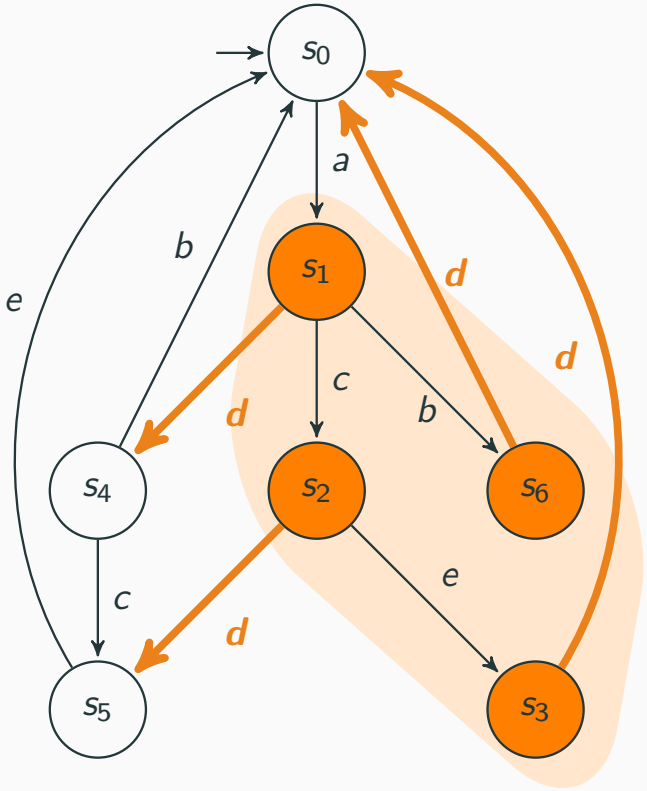
Regions

Region: subset of states in which events have the same *enter*/*exit*/*no_cross* behavior.



Regions

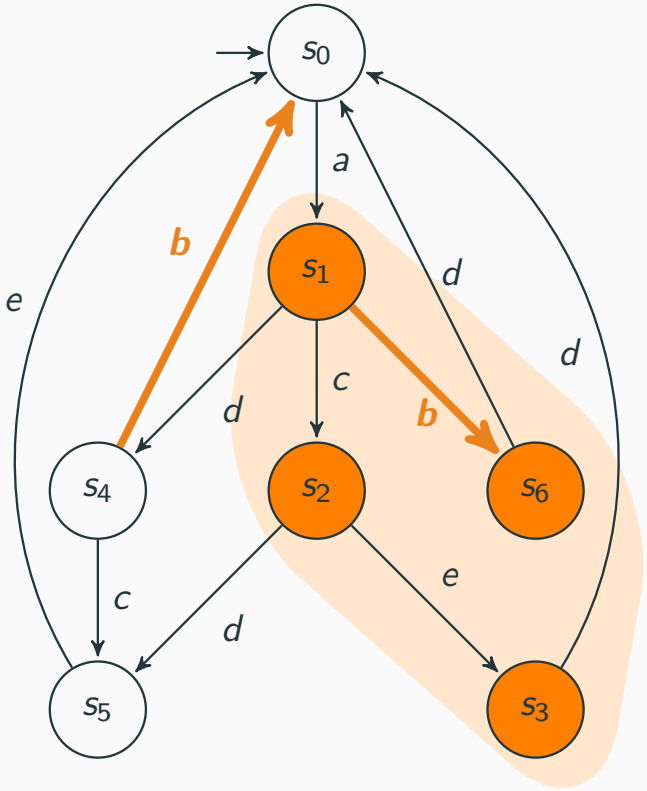
Region: subset of states in which events have the same *enter*/*exit*/*no_cross* behavior.



If a region r has the **exit** property with e we can say that r is a **pre-region** of e i.e. $r \in {}^\circ e$.

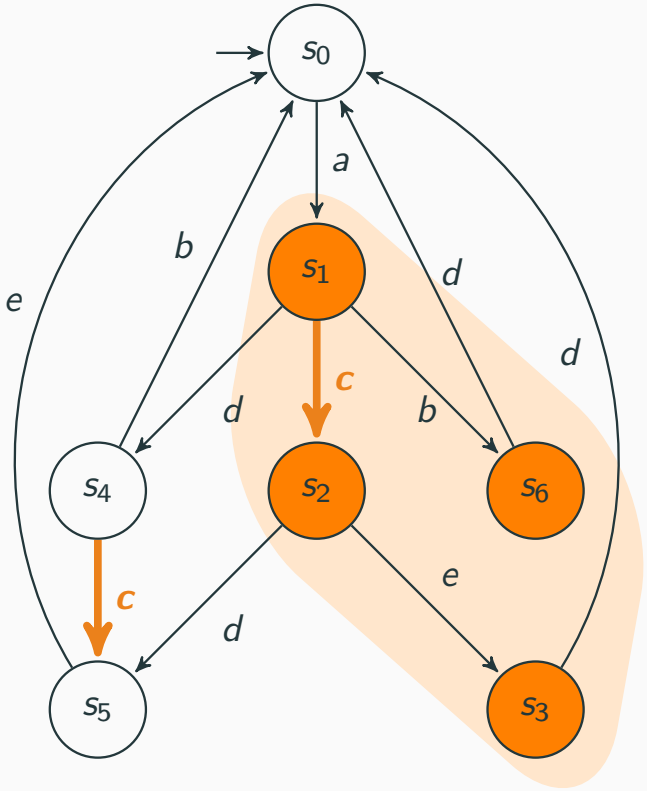
Regions

Region: subset of states in which events have the same *enter*/*exit*/*no_cross* behavior.



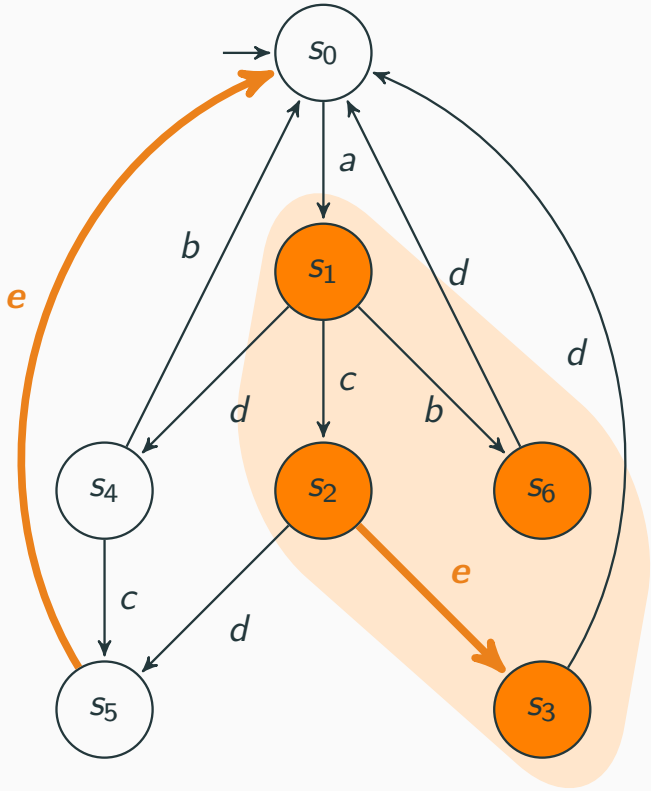
Regions

Region: subset of states in which events have the same *enter*/*exit*/*no_cross* behavior.



Regions

Region: subset of states in which events have the same *enter*/*exit*/*no_cross* behavior.



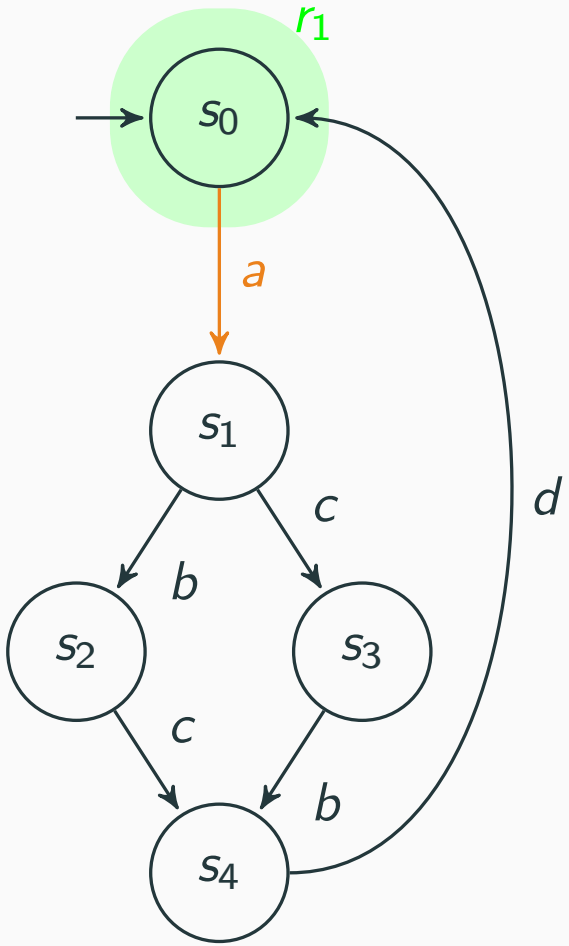
Excitation-closed transition system (ECTS)

Excitation region of an event: set of states in which the event is activated.

If for each event:

$$\bigcap \text{pre-regions} = \text{excitation region}$$

then we have an **Excitation-closed transition system**.



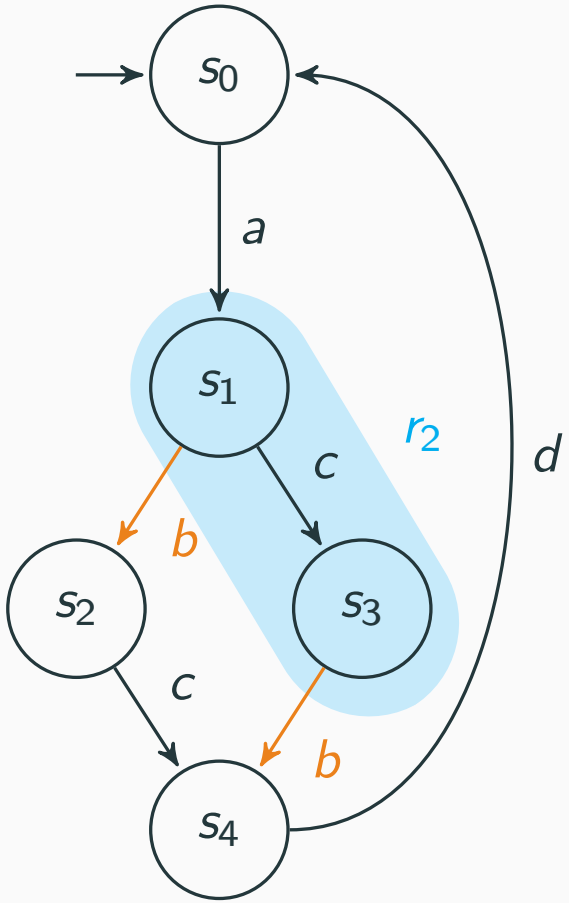
Excitation-closed transition system (ECTS)

Excitation region of an event: set of states in which the event is activated.

If for each event:

$$\bigcap \text{pre-regions} = \text{excitation region}$$

then we have an **Excitation-closed transition system**.



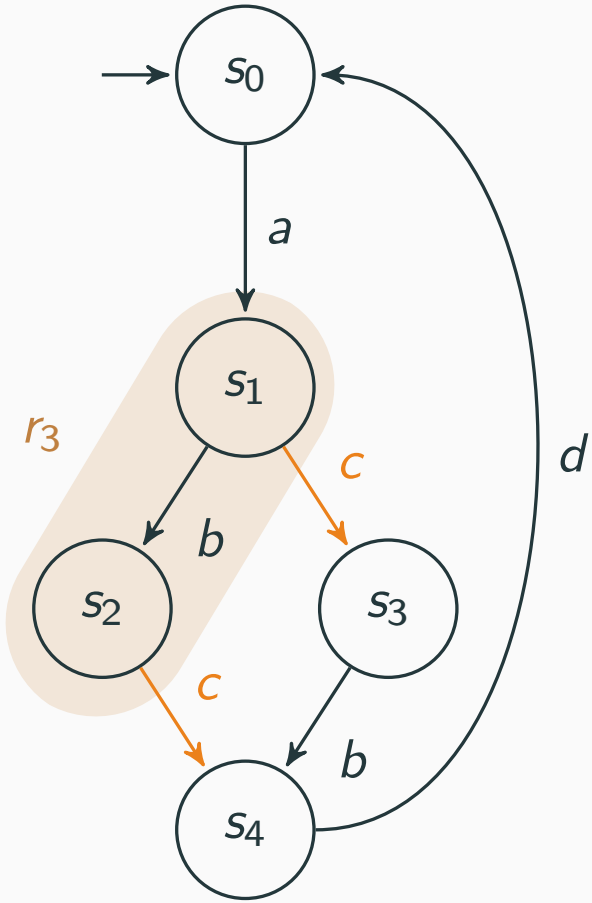
Excitation-closed transition system (ECTS)

Excitation region of an event: set of states in which the event is activated.

If for each event:

$$\bigcap \text{pre-regions} = \text{excitation region}$$

then we have an **Excitation-closed transition system**.



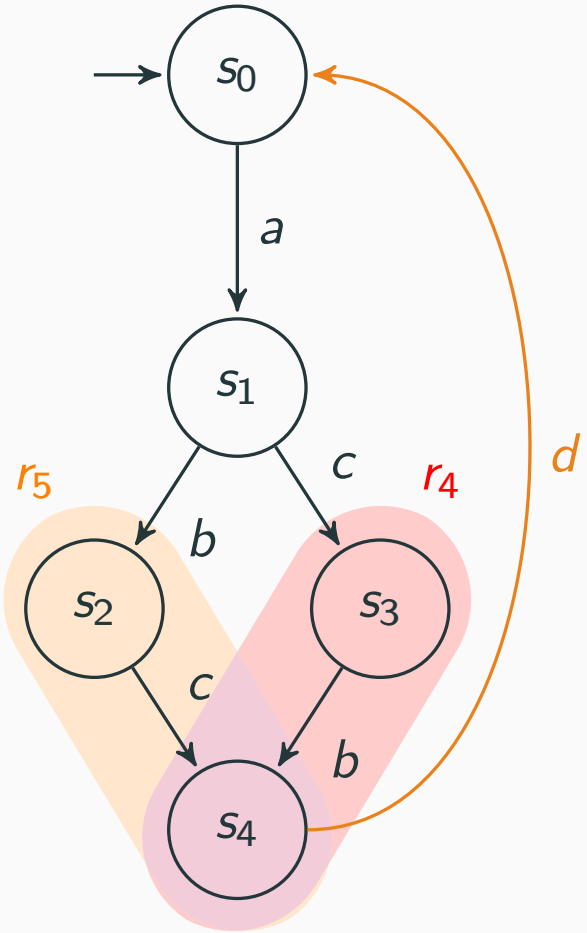
Excitation-closed transition system (ECTS)

Excitation region of an event: set of states in which the event is activated.

If for each event:

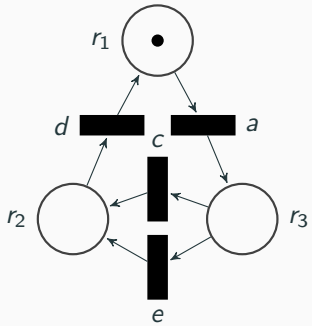
$$\bigcap \text{pre-regions} = \text{excitation region}$$

then we have an **Excitation-closed transition system**.

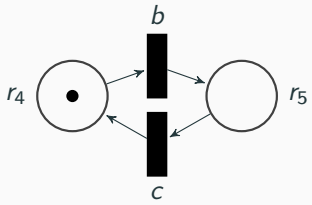


How does the synchronization between PNs work?

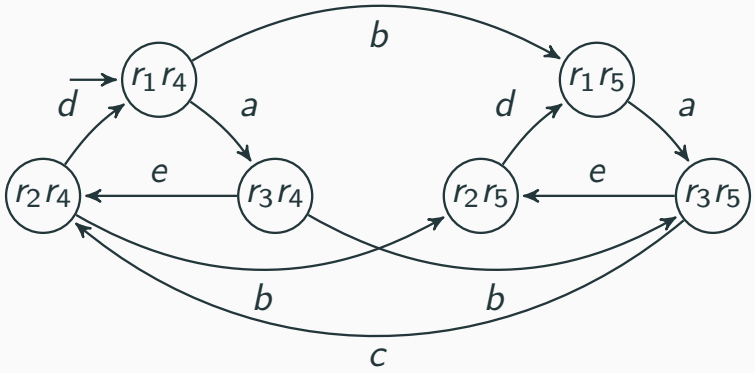
Intuitively, the PNs **cooperate** with the same rules of the **synchronous product** of transition systems.



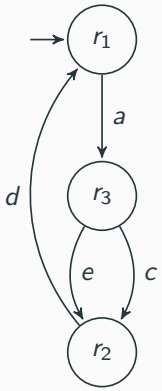
PN_1



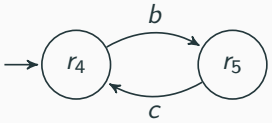
PN_2



$TS_1 || TS_2$



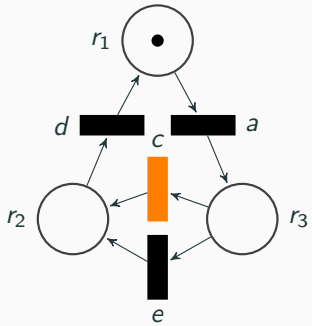
TS_1



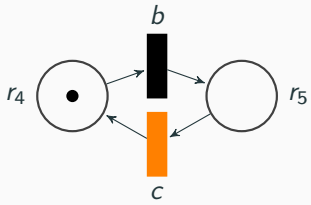
TS_2

How does the synchronization between PNs work?

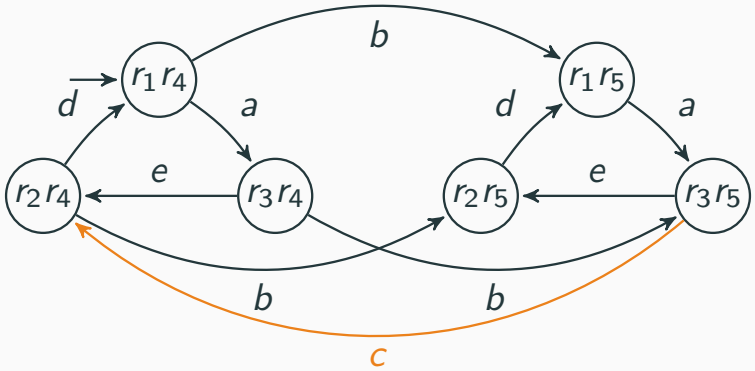
Intuitively, the PNs **cooperate** with the same rules of the **synchronous product** of transition systems.



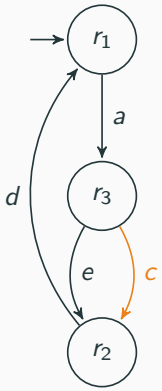
PN_1



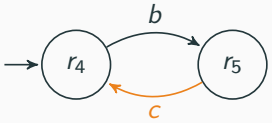
PN_2



$TS_1 || TS_2$



TS_1

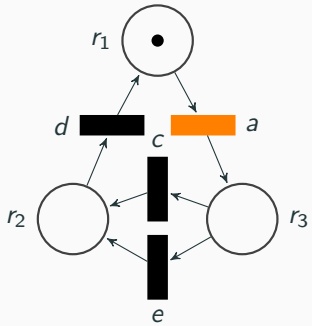


TS_2

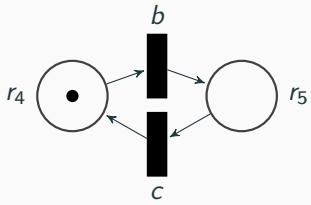
Case 1: common event

How does the synchronization between PNs work?

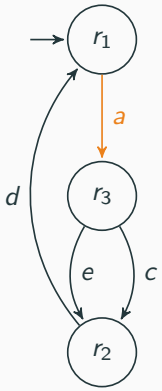
Intuitively, the PNs **cooperate** with the same rules of the **synchronous product** of transition systems.



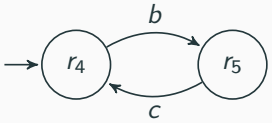
PN_1



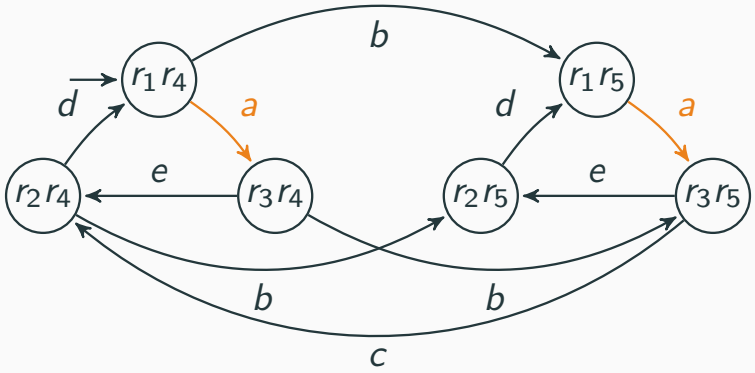
PN_2



TS_1



TS_2



$TS_1 || TS_2$

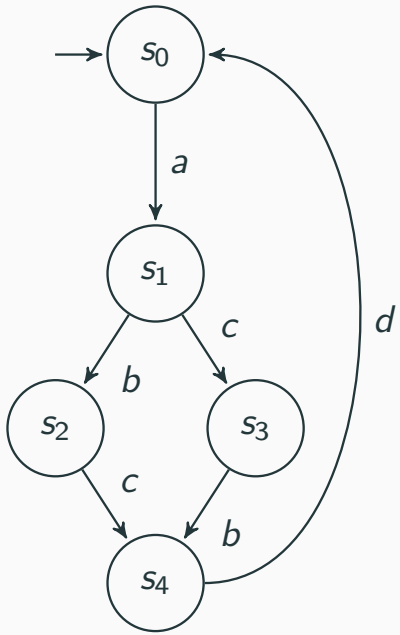
Case 1: common event

Case 2: the event appears in one of the PNs

How to derive a PN from a set of regions?

For each region:

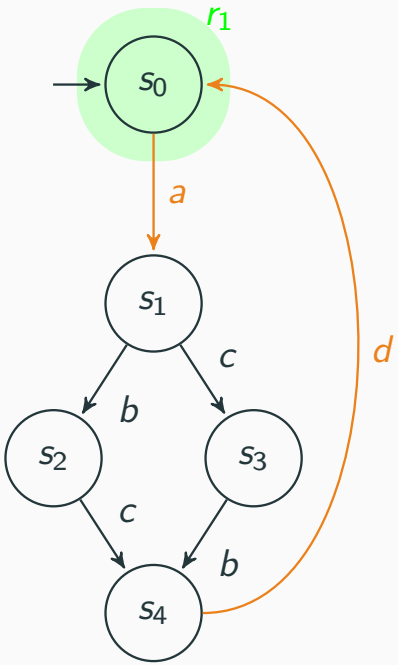
1. Create the place representing the region
   1.1 If the region contains the initial state the place will take part of the initial marking
2. Connect the place to transitions representing $enter/exit$ events with respect to the region



How to derive a PN from a set of regions?

For each region:

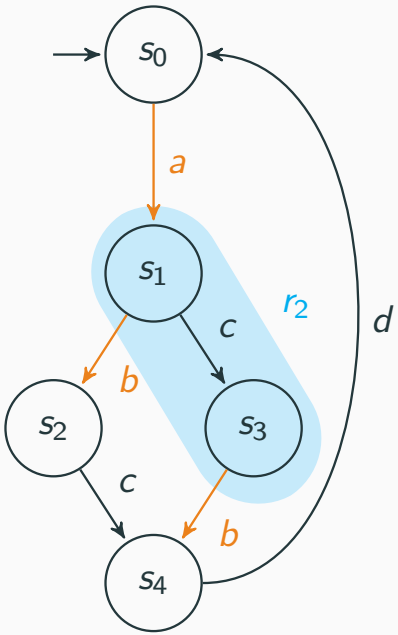
1. Create the place representing the region
   1.1 If the region contains the initial state the place will take part of the initial marking
2. Connect the place to transitions representing $enter/exit$ events with respect to the region

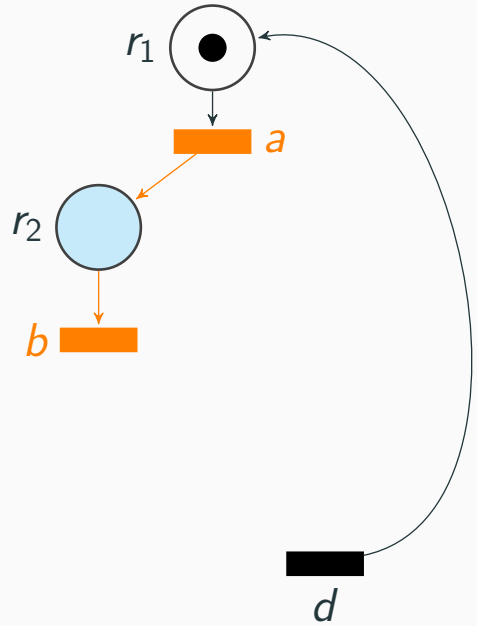


How to derive a PN from a set of regions?

For each region:

1. Create the place representing the region
   1.1 If the region contains the initial state the place will take part of the initial marking
2. Connect the place to transitions representing $enter/exit$ events with respect to the region

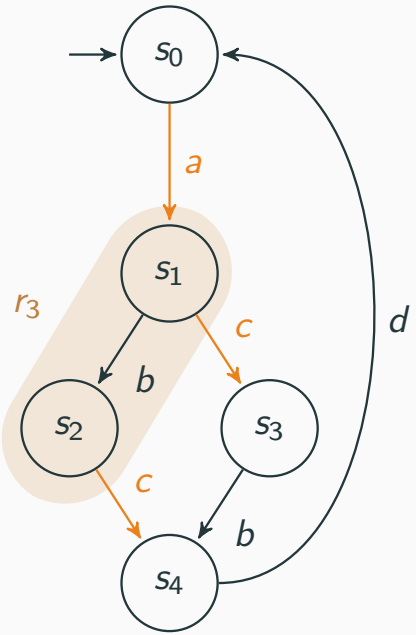


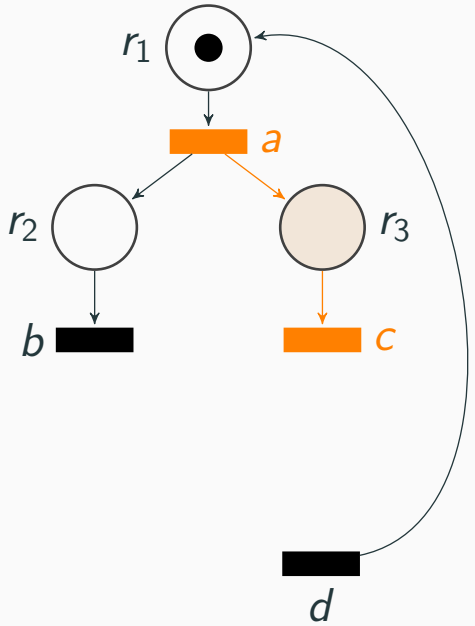


How to derive a PN from a set of regions?

For each region:

1. Create the place representing the region
   1.1 If the region contains the initial state the place will take part of the initial marking
2. Connect the place to transitions representing $enter/exit$ events with respect to the region

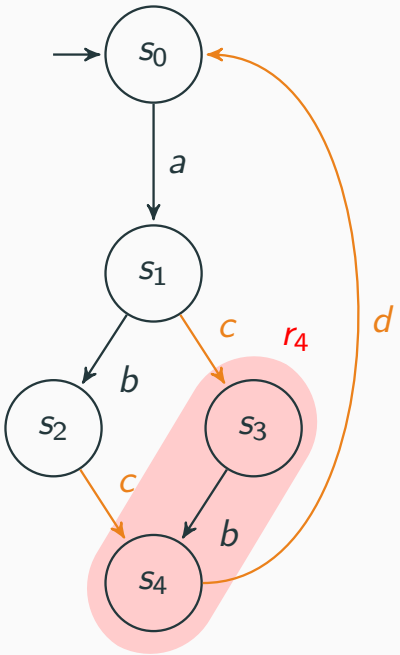


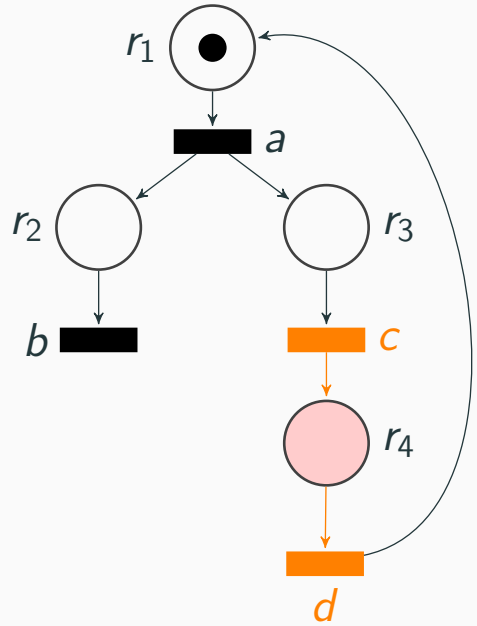


How to derive a PN from a set of regions?

For each region:

1. Create the place representing the region
   1.1 If the region contains the initial state the place will take part of the initial marking
2. Connect the place to transitions representing $enter/exit$ events with respect to the region

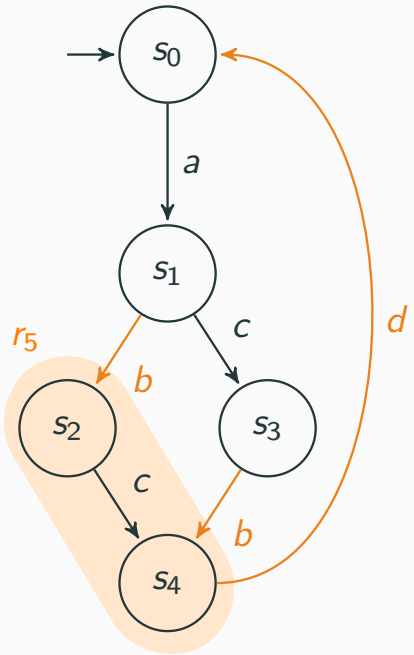


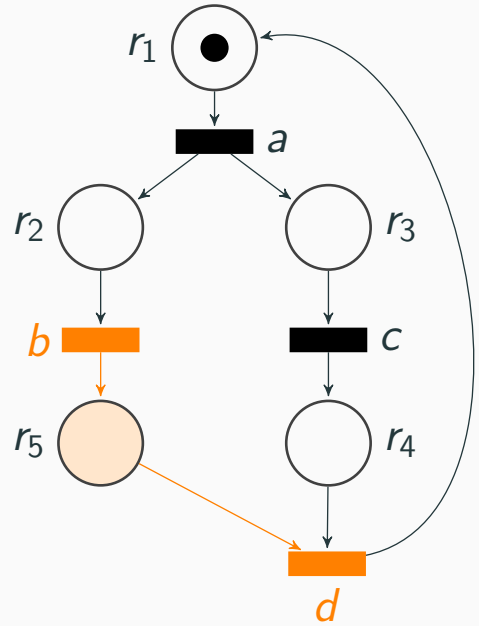


How to derive a PN from a set of regions?

For each region:

1. Create the place representing the region
   1.1 If the region contains the initial state the place will take part of the initial marking
2. Connect the place to transitions representing $enter/exit$ events with respect to the region

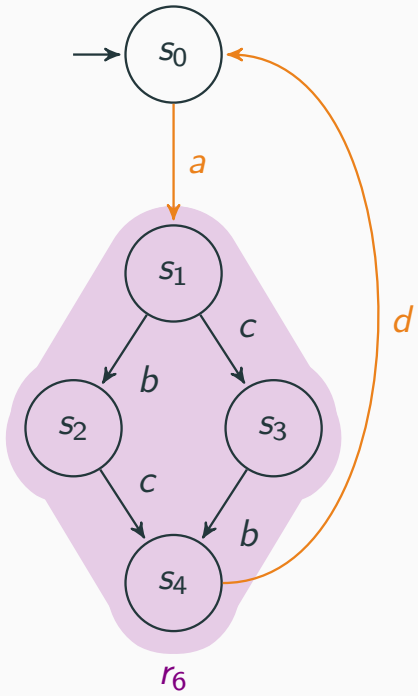


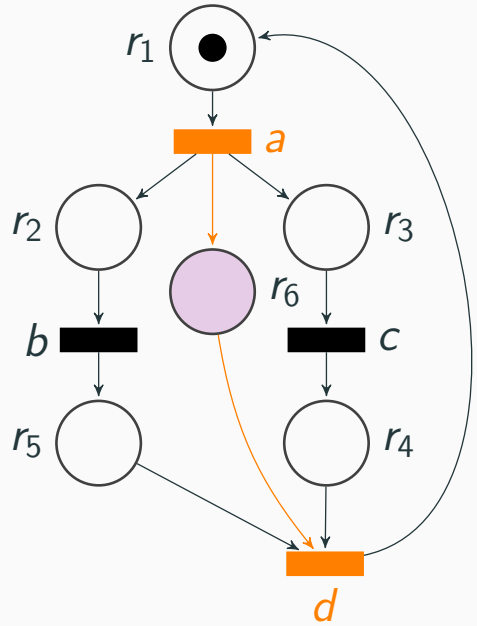


How to derive a PN from a set of regions?

For each region:

1. Create the place representing the region
   1.1 If the region contains the initial state the place will take part of the initial marking
2. Connect the place to transitions representing $enter/exit$ events with respect to the region

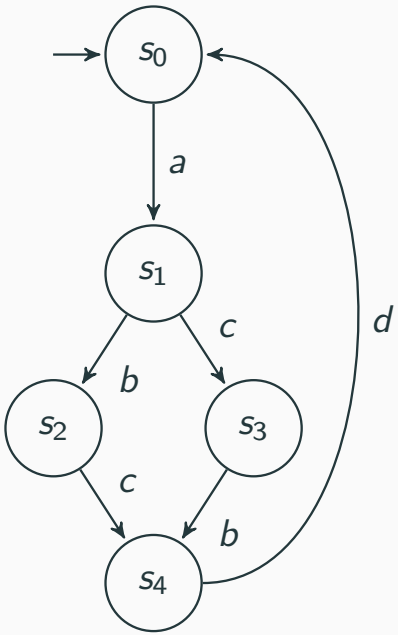


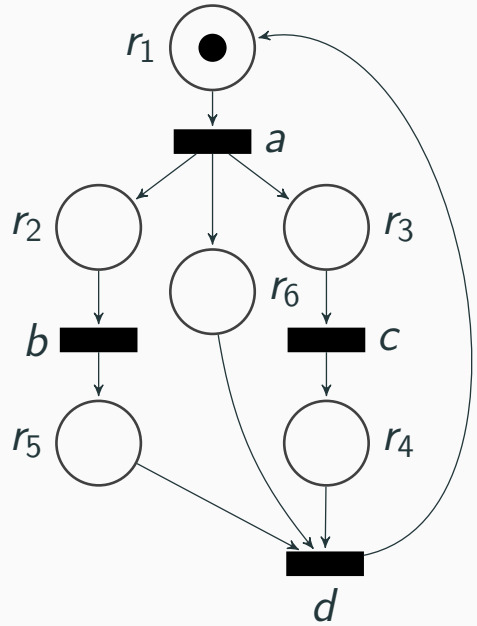


How to derive a PN from a set of regions?

For each region:

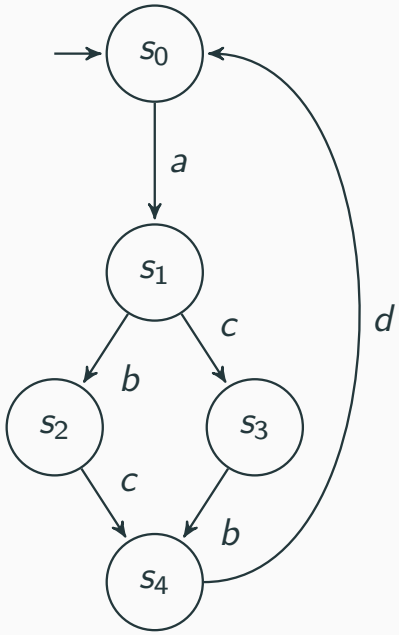
1. Create the place representing the region
   1.1 If the region contains the initial state the place will take part of the initial marking
2. Connect the place to transitions representing $enter/exit$ events with respect to the region

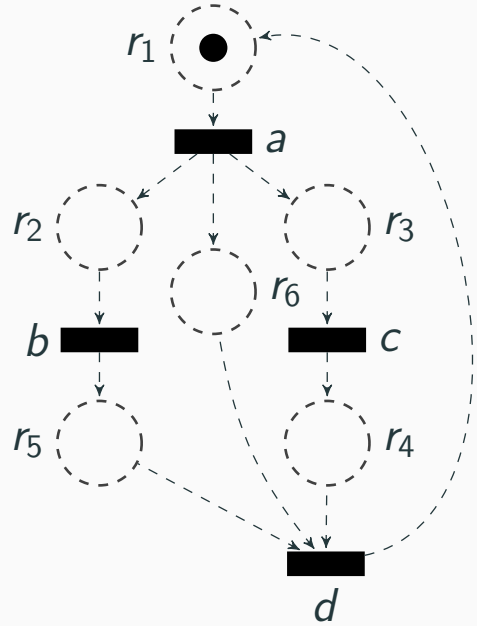




When do the regions represent an eligible PN?

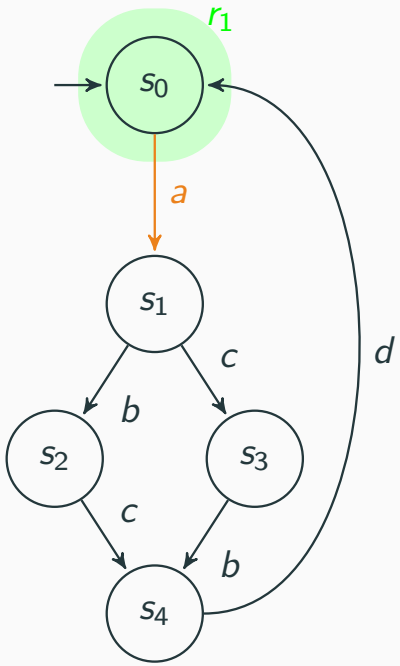
Excitation-closure

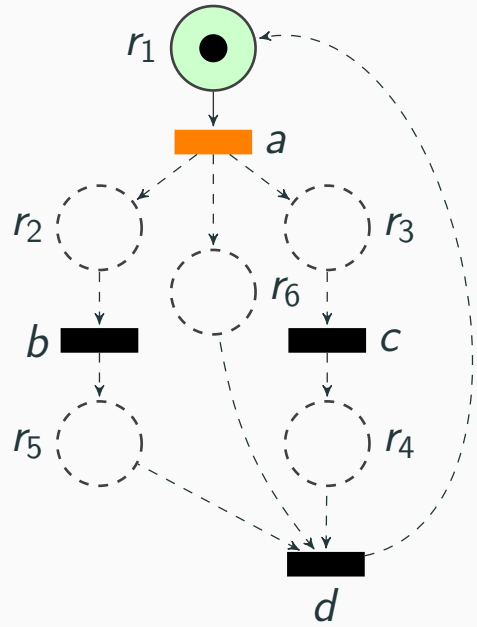




When do the regions represent an eligible PN?

Excitation-closure

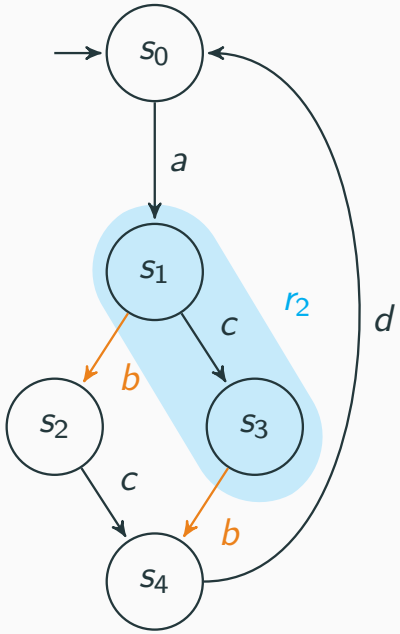


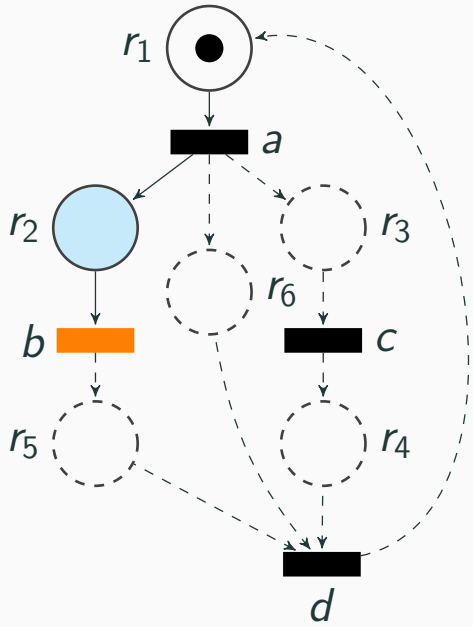


$$a : \{s_0\} = r_1$$

When do the regions represent an eligible PN?

Excitation-closure

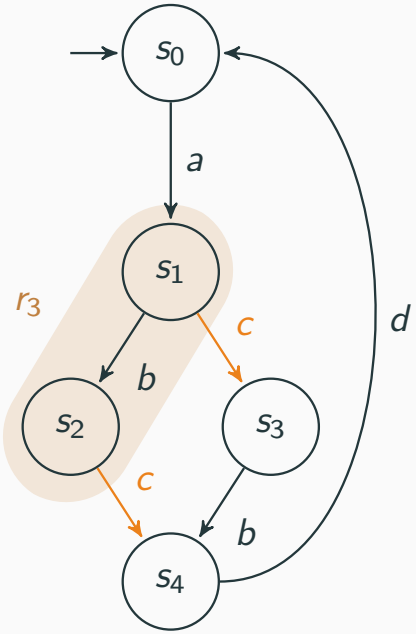


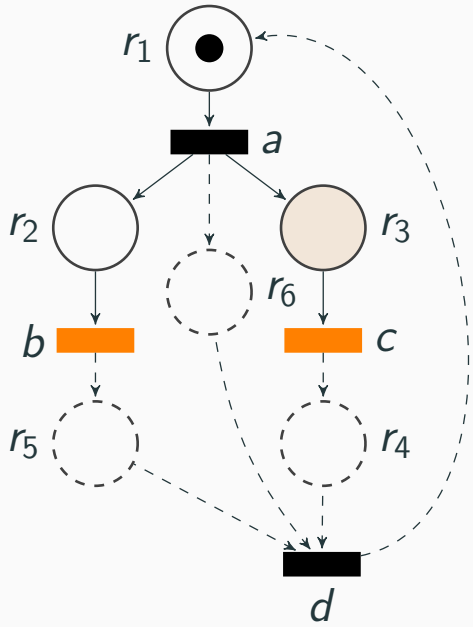


$$b : \{s_1, s_3\} = r_2$$

When do the regions represent an eligible PN?

Excitation-closure

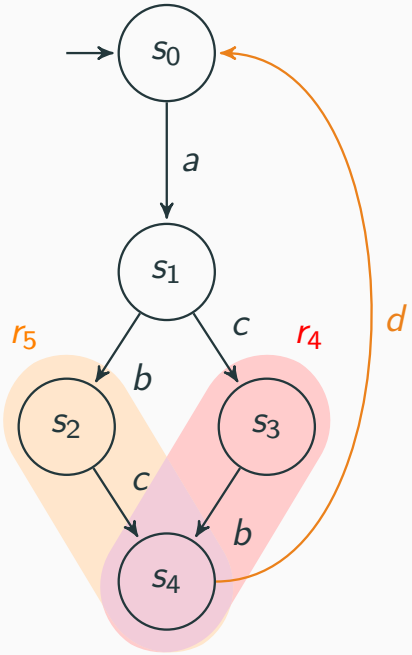


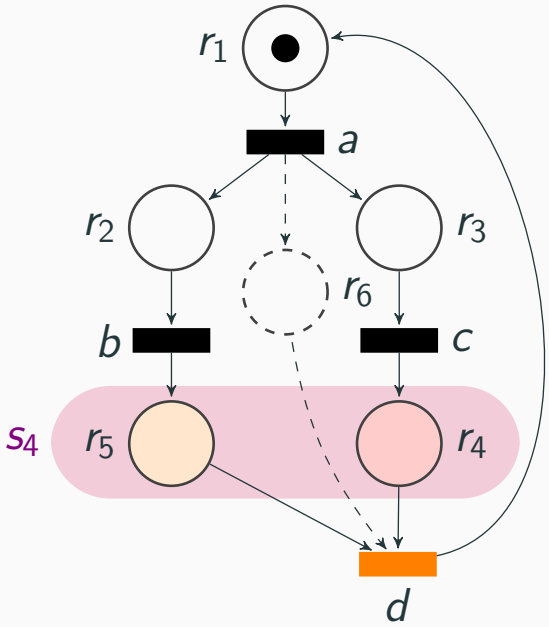


$$c : \{s_1, s_2\} = r_3$$

When do the regions represent an eligible PN?

Excitation-closure

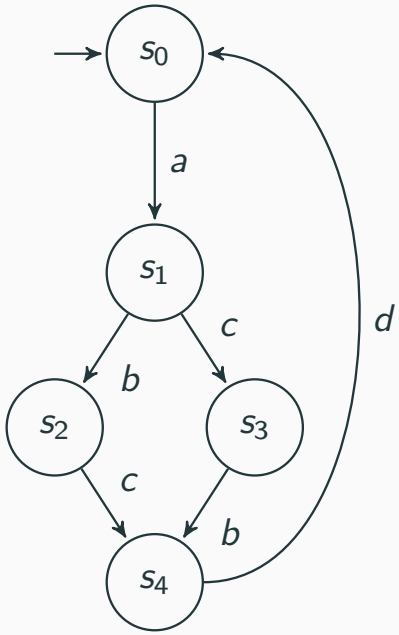


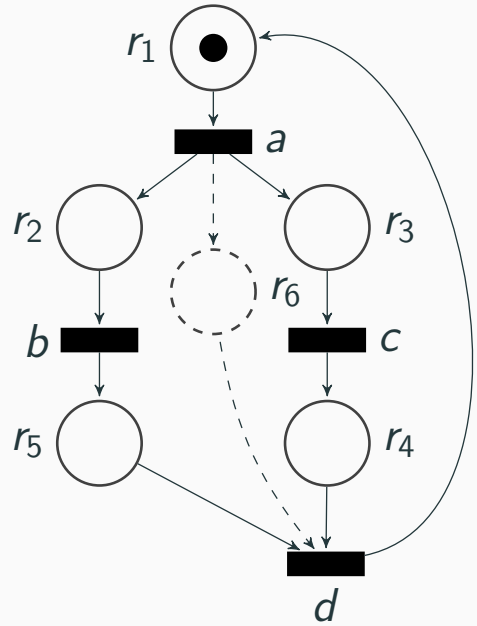


$$d : \{s_4\} = r_4 \cap r_5$$

When do the regions represent an eligible PN?

Excitation-closure

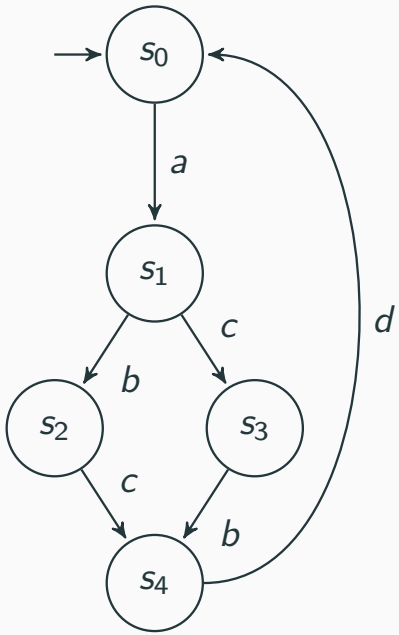


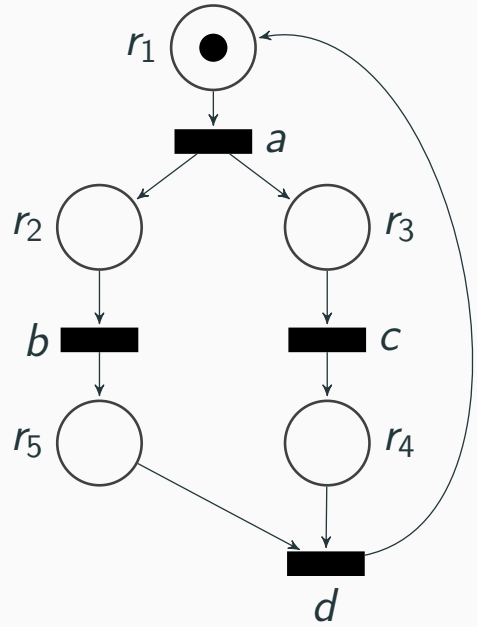


r_6 is not necessary!!!

When do the regions represent an eligible PN?

Excitation-closure



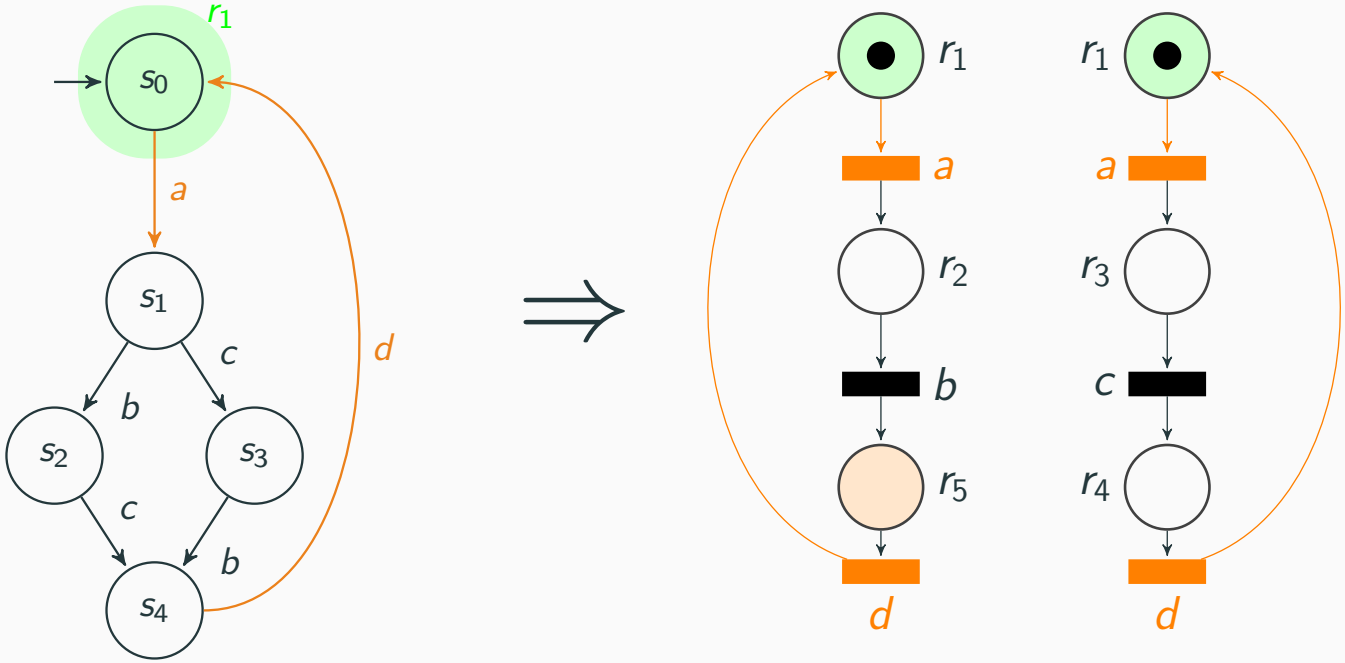


When do the regions represent a set of eligible PNs?

For a set of PNs excitation-closure is not enough, we also need:

Place connectedness

Given a region, all its incoming and outgoing events are also included.

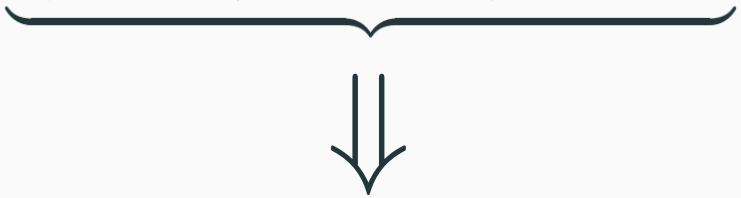


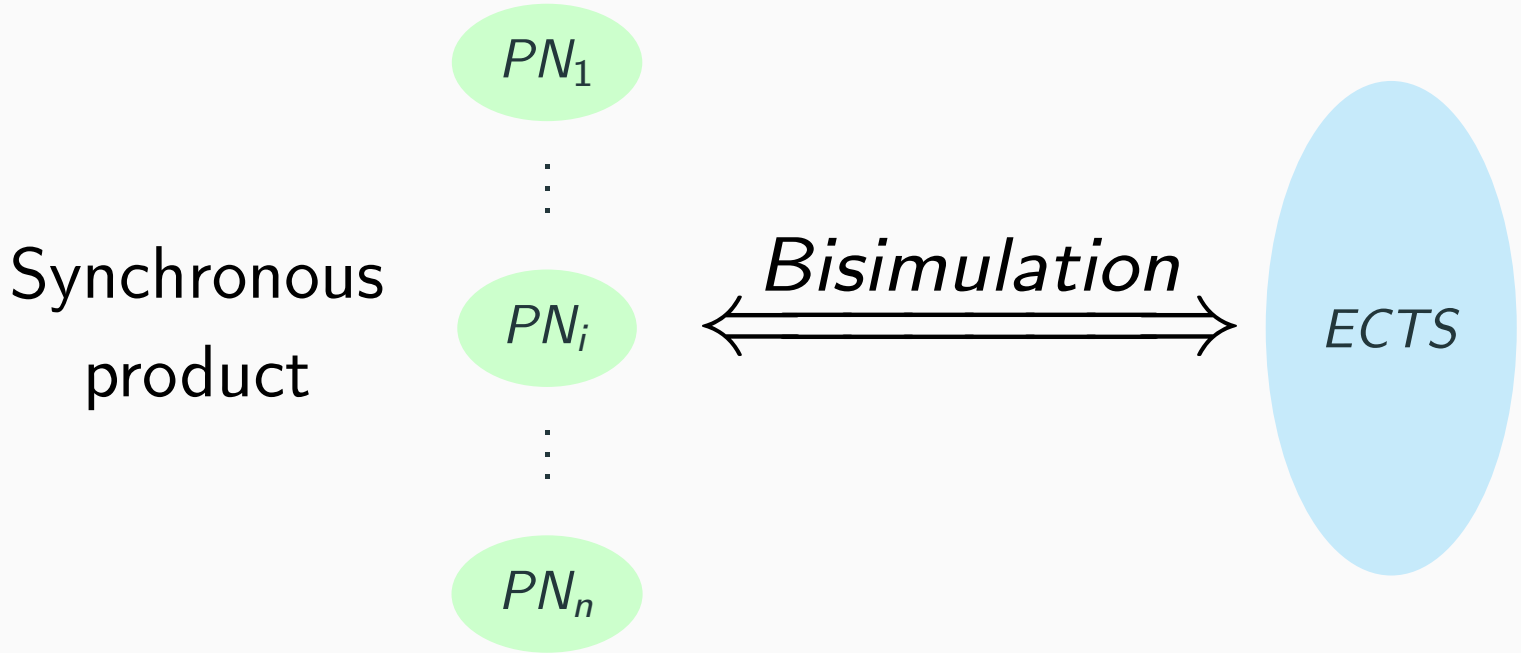
Place connectedness guarantees **marking consistency** across all Petri nets.

E.g., the TS on the left is decomposed into the two SMs on the right.

Theorem of equivalence (by bisimulation)

1. **Excitation-closure**
2. **Place connectedness**



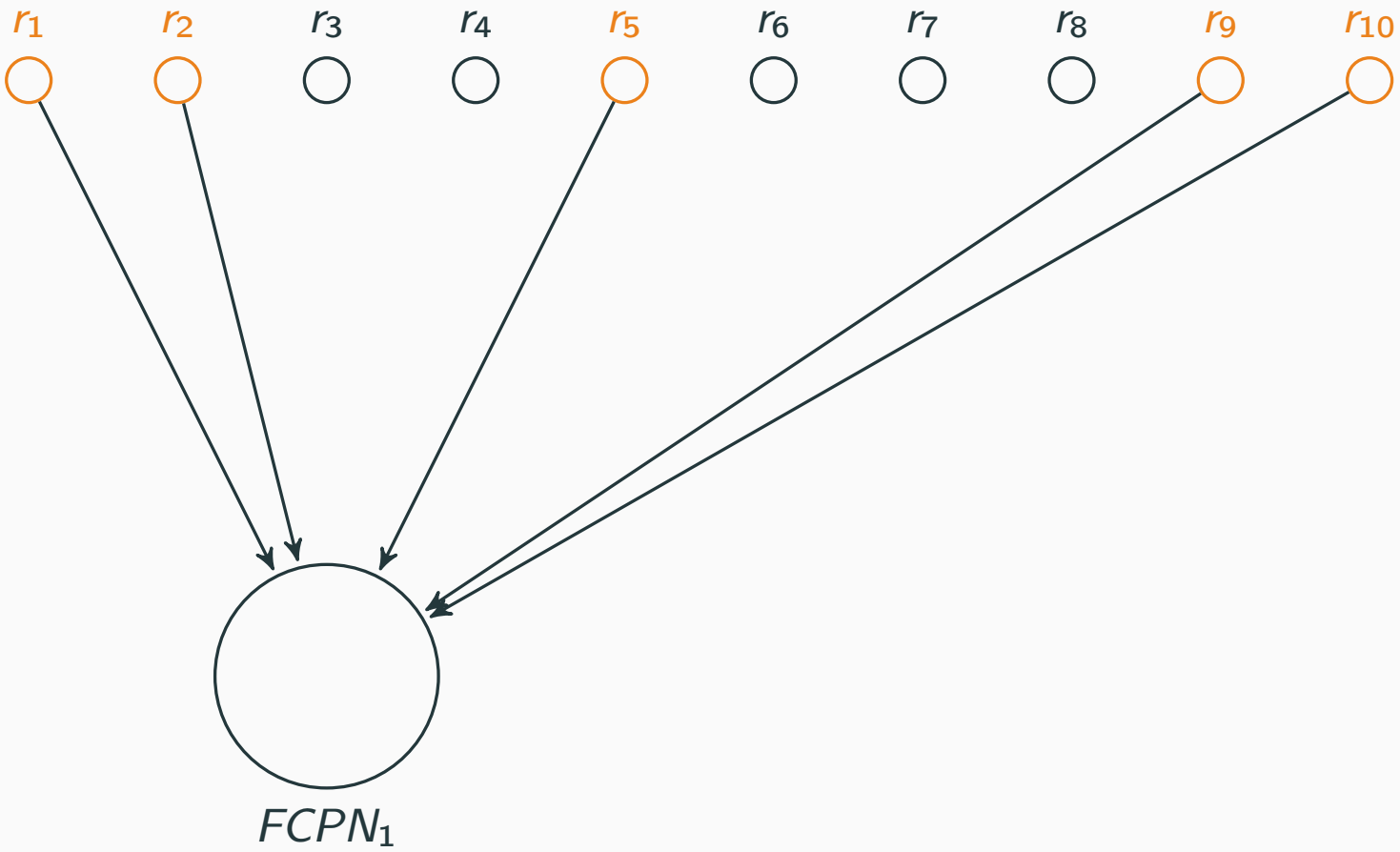


Decomposition of Transition Systems

Iterative extraction

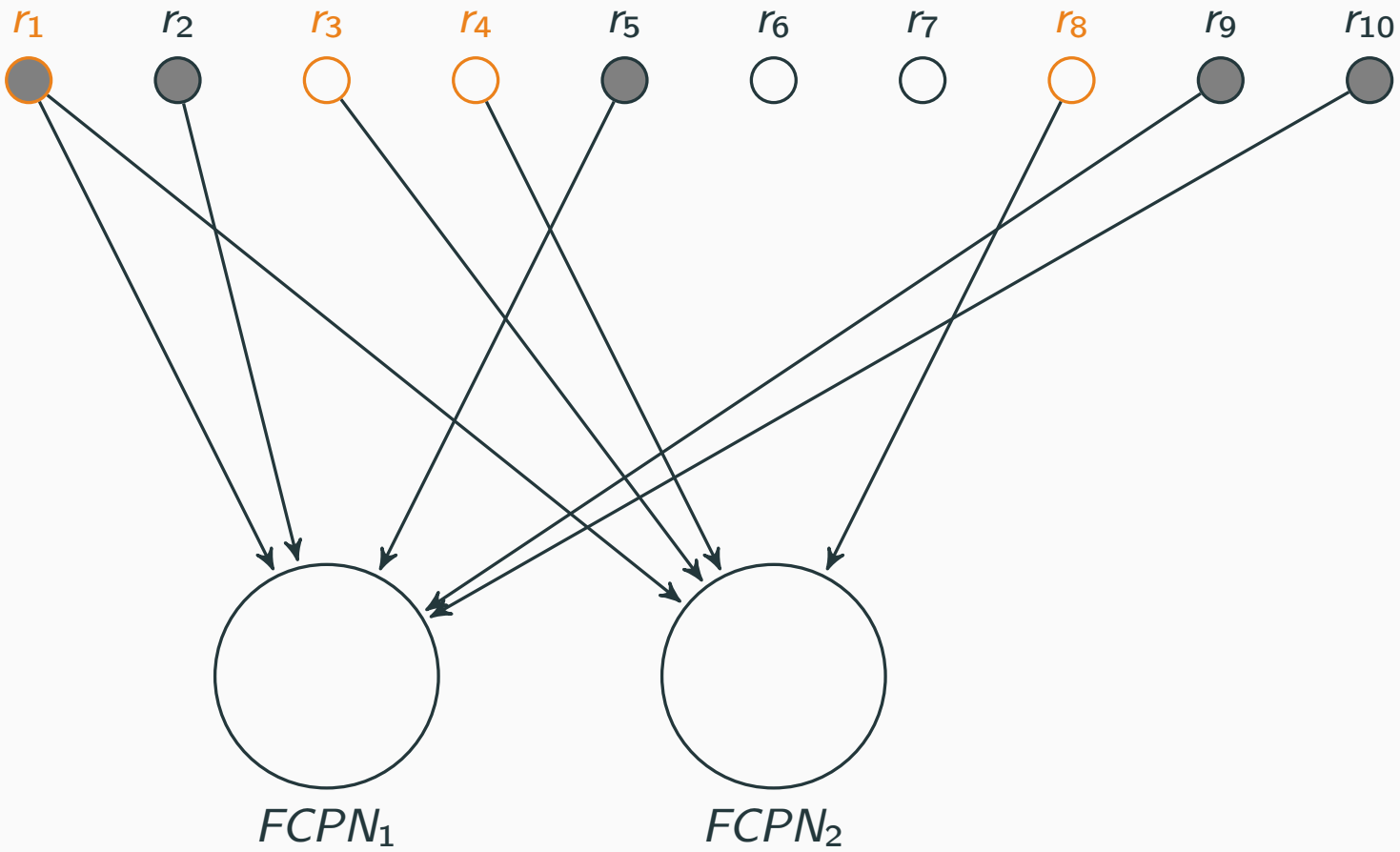
Iterative extraction

SAT formula F: constraints (Place connectedness and maximization of new regions) and FCPN property



Iterative extraction

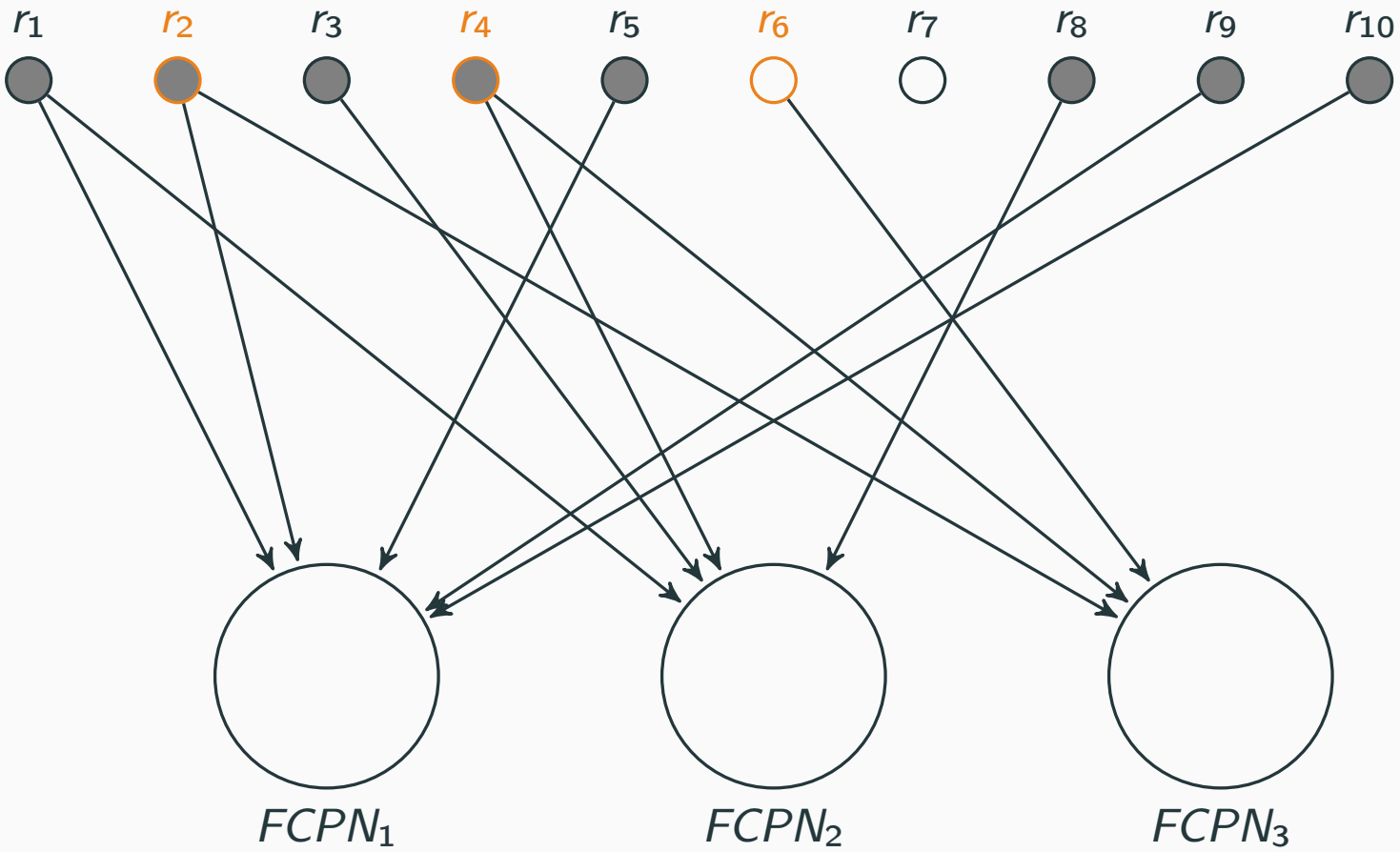
SAT formula F: constraints (Place connectedness and maximization of new regions) and FCPN property



Each component maximizes the number of new regions (**Pseudo-boolean optimization**).

Iterative extraction

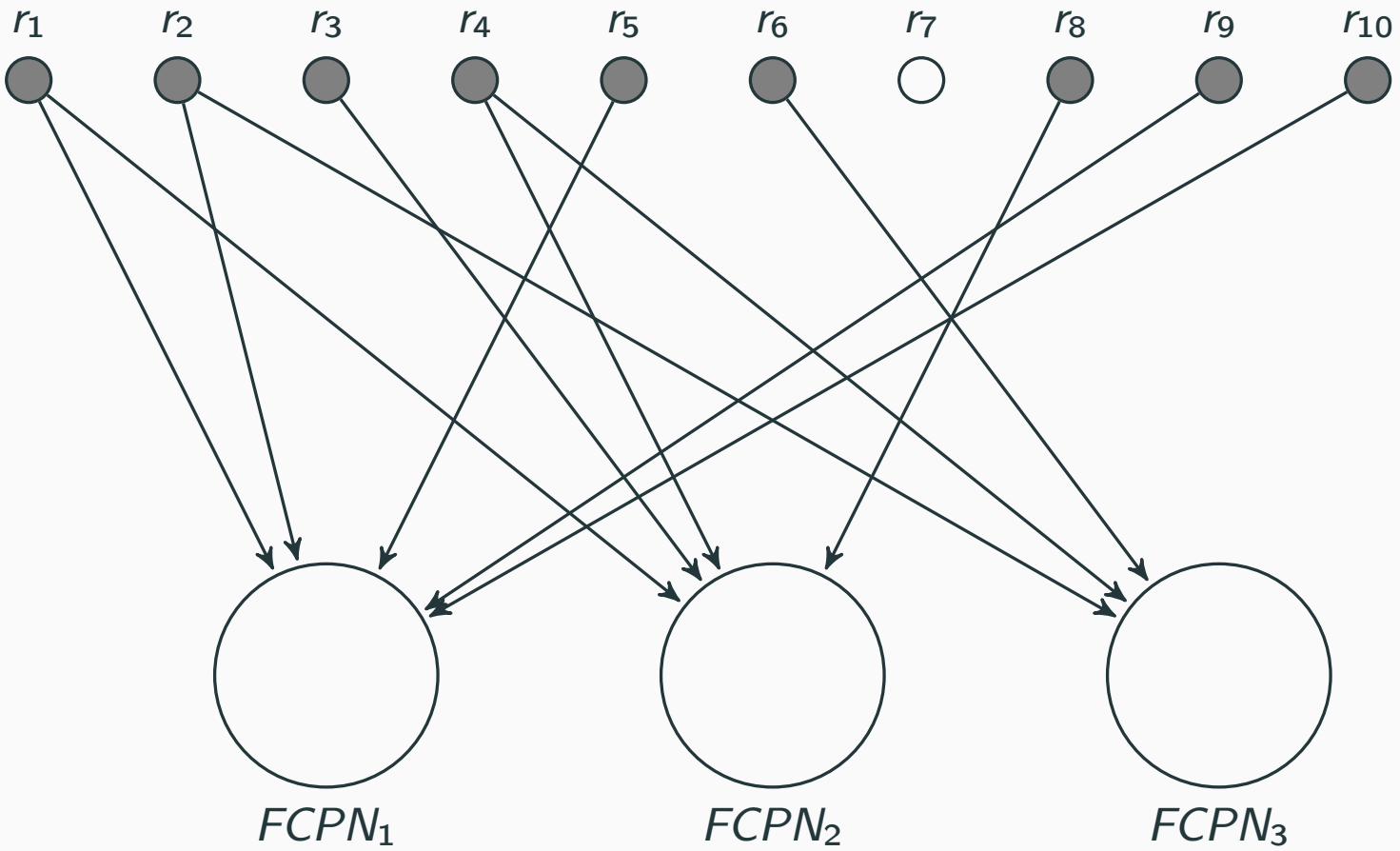
SAT formula F: constraints (Place connectedness and maximization of new regions) and FCPN property



Each component maximizes the number of new regions (**Pseudo-boolean optimization**).

Iterative extraction

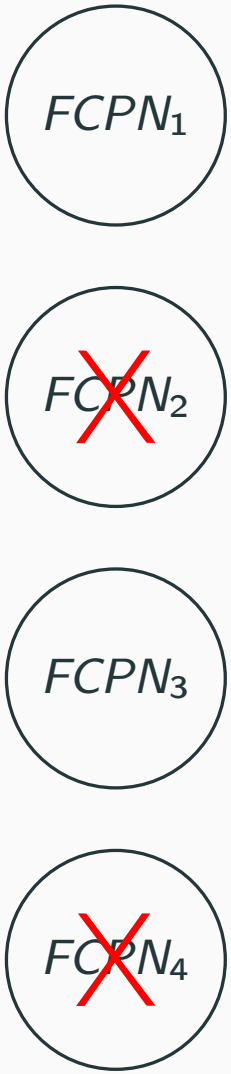
SAT formula F: constraints (Place connectedness and maximization of new regions) and FCPN property



Not all regions are necessary!!!

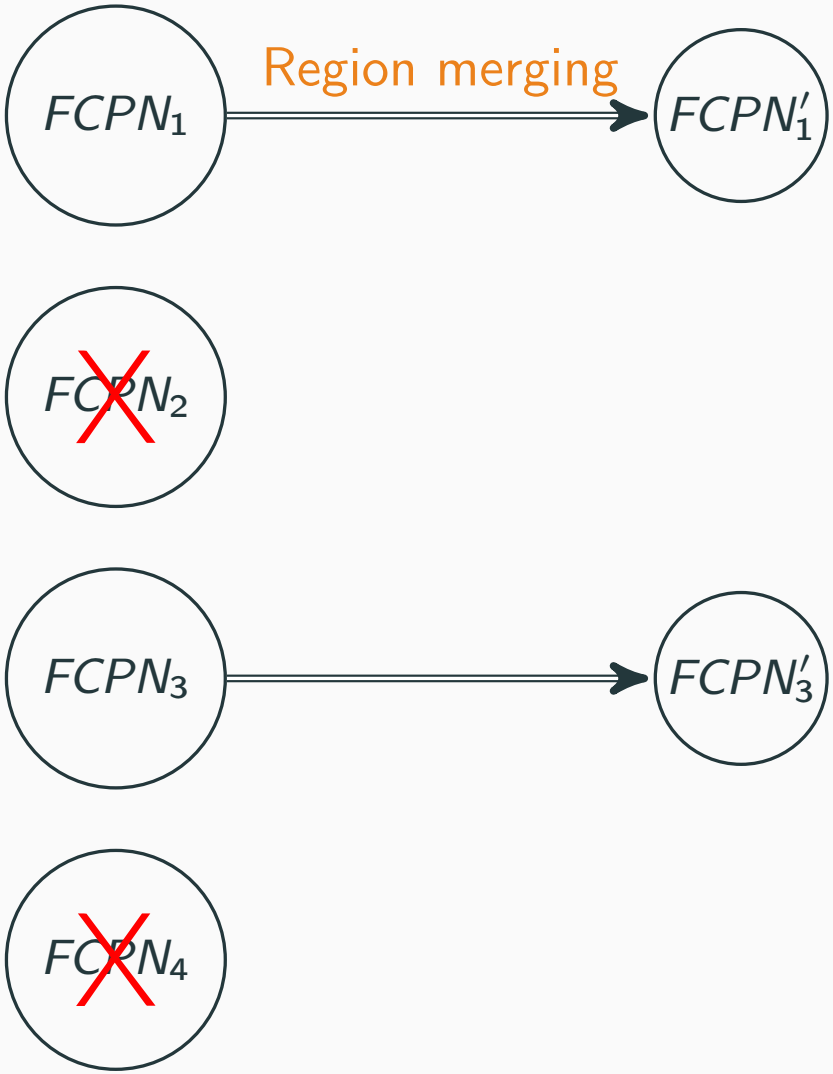
Post-optimizations

Post-optimizations



Greedy removal of FCPNs

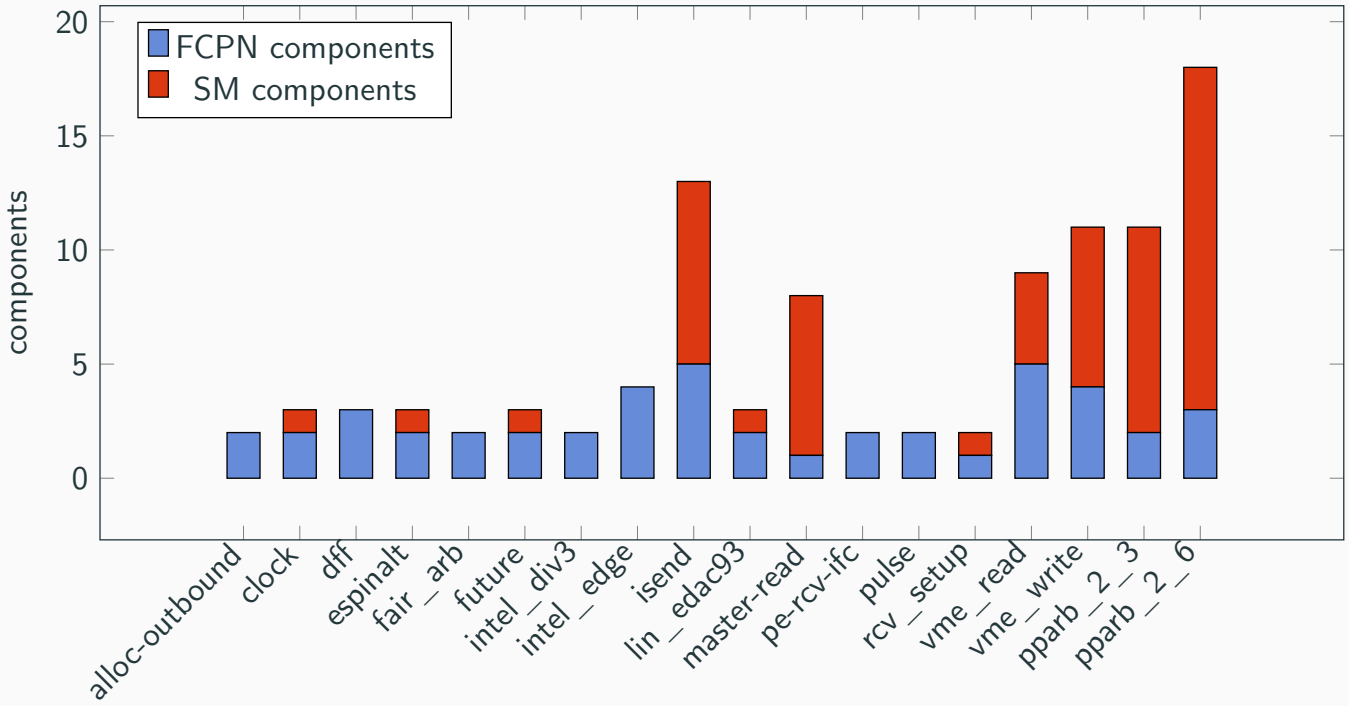
Post-optimizations



Greedy removal of FCPNs

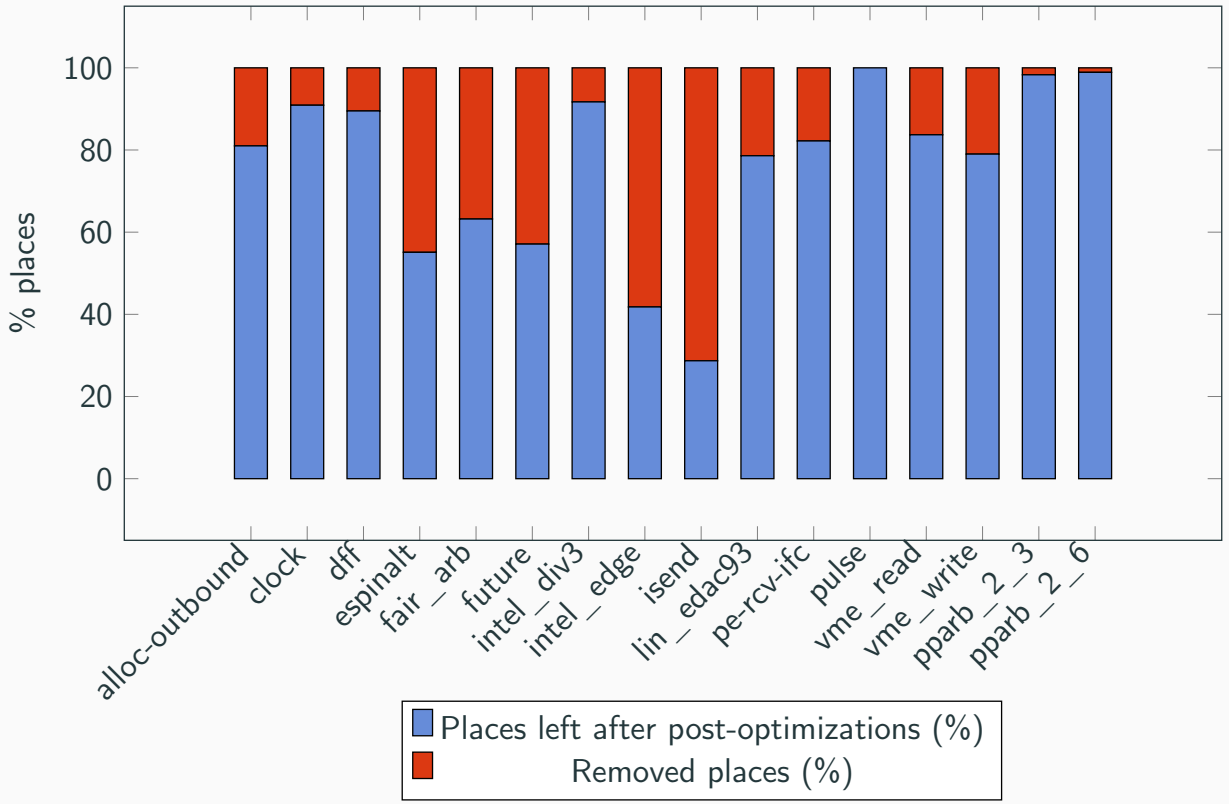
Results

Reduction in number of components (SMs vs FCPNs)



- Average decrease from **5.6** components to **2.6**
- The new step did not require more computational effort
- Bottleneck: more of **90%** of time spent in generation of regions
- Maximum TS size: **90k** state and **320k** transitions
- Computational times do not exceed 10 mins

Post-optimizations are important



39% of places removed on average

Conclusion and Future work

We extended the decomposition flow to include concurrent components.

Future work:

- Extension to other types of Petri nets
- **Parallel** computation to mitigate the cost of region generation
- Application to **process mining**