

Background

Parameterized Systems

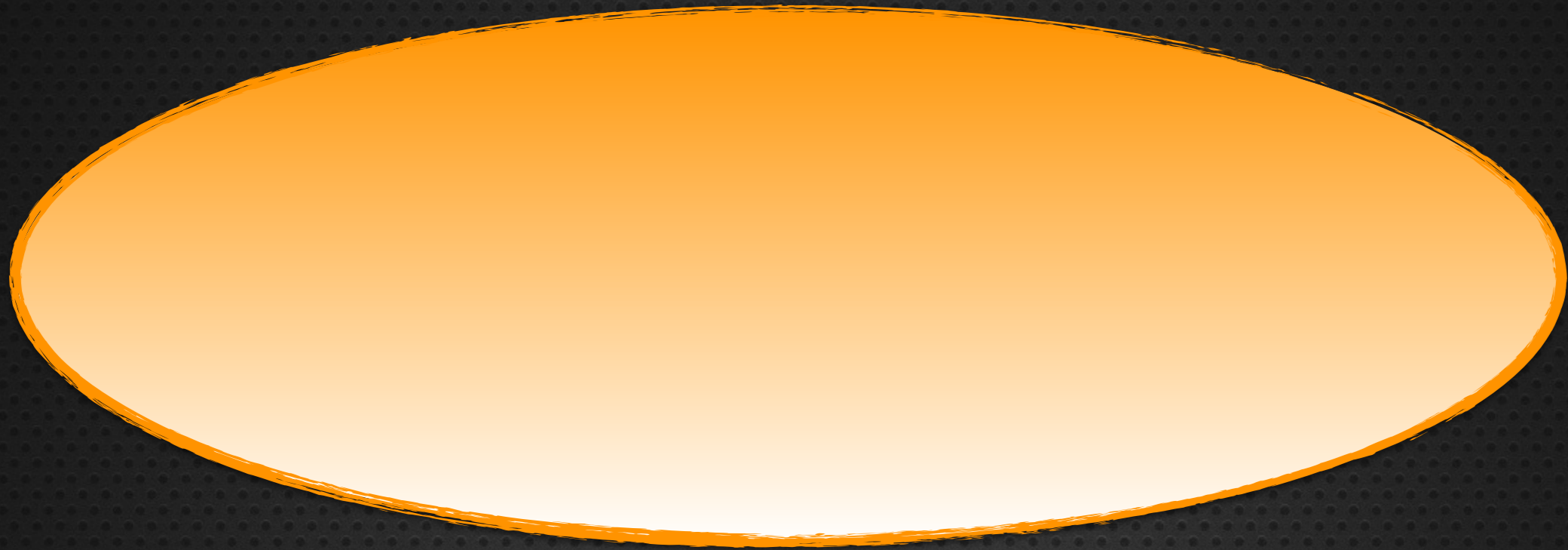
Petri Nets

Lossy Channel Systems

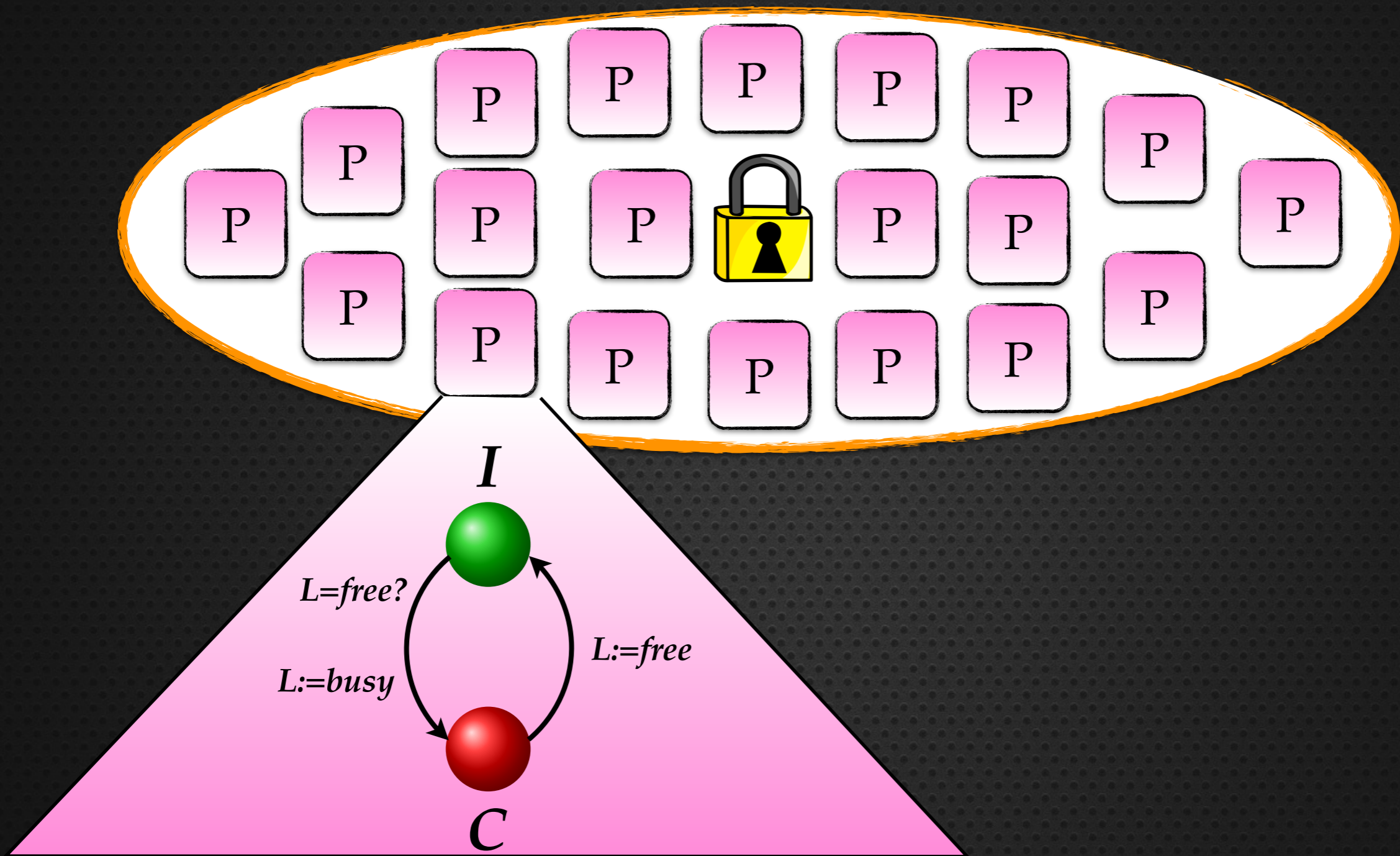
Timed Petri Nets

Parameterized Systems

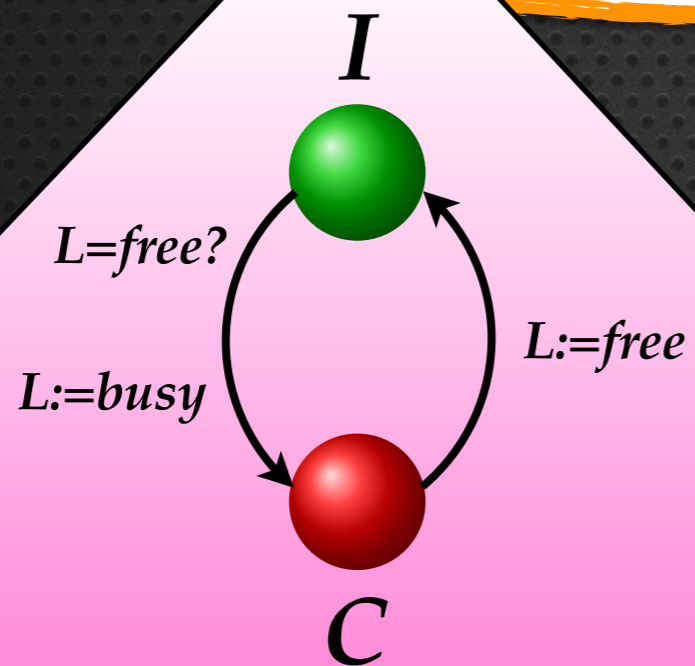
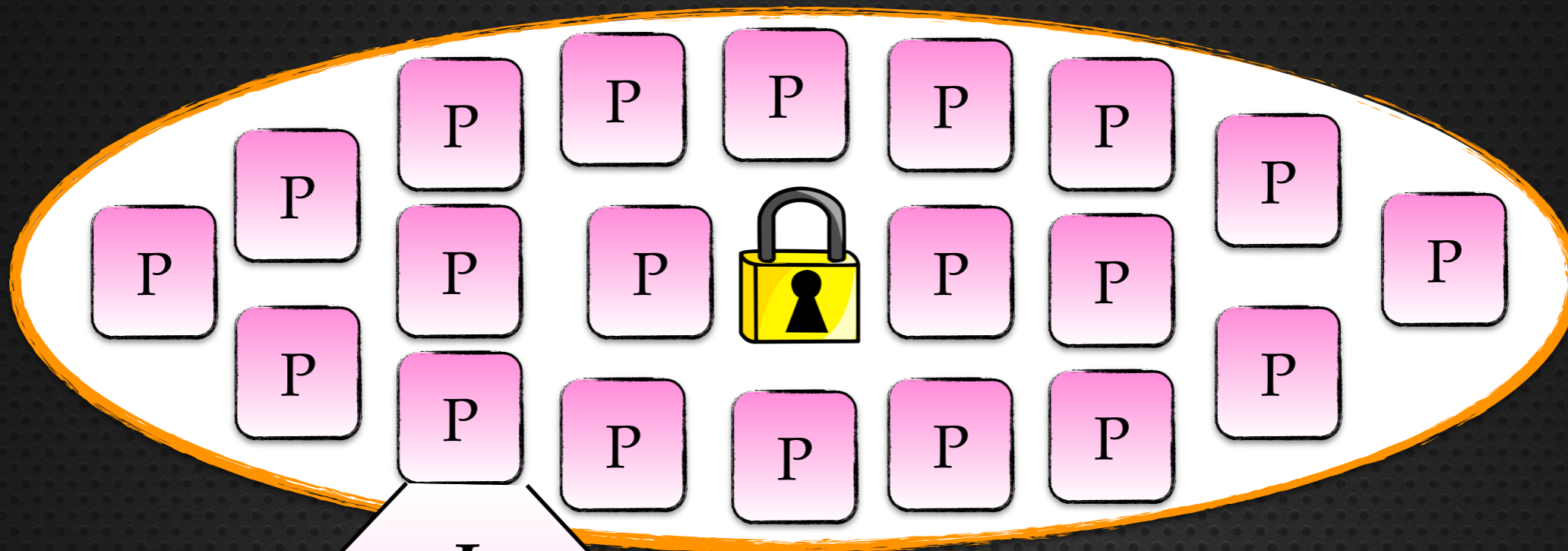
Parameterized Systems



Parameterized Systems



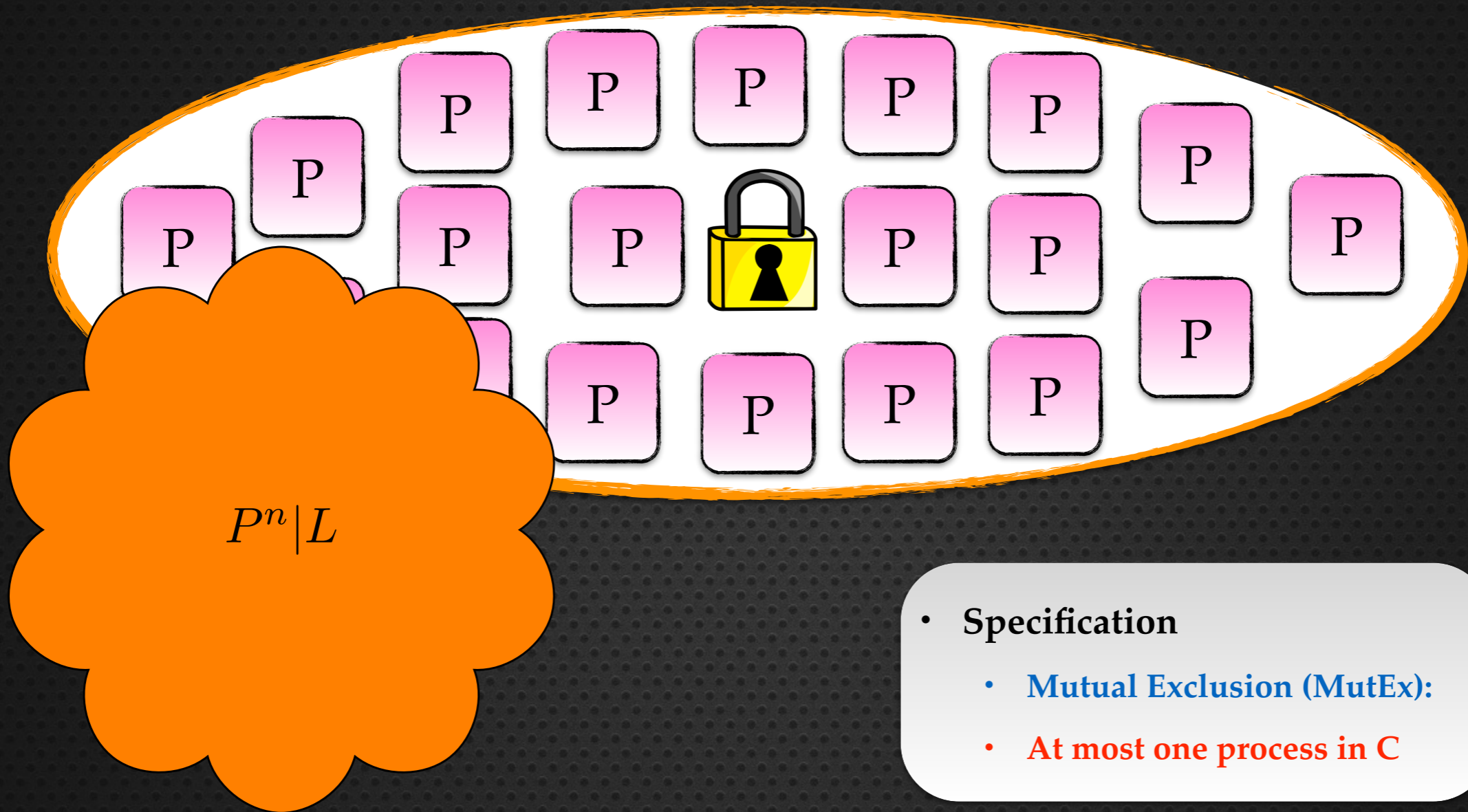
Parameterized Systems



- Specification

- **Mutual Exclusion (MutEx):**
- **At most one process in C**

Parameterized Systems



$P^n | L$

- Specification
 - Mutual Exclusion (MutEx):
 - At most one process in C

Parameterized Systems



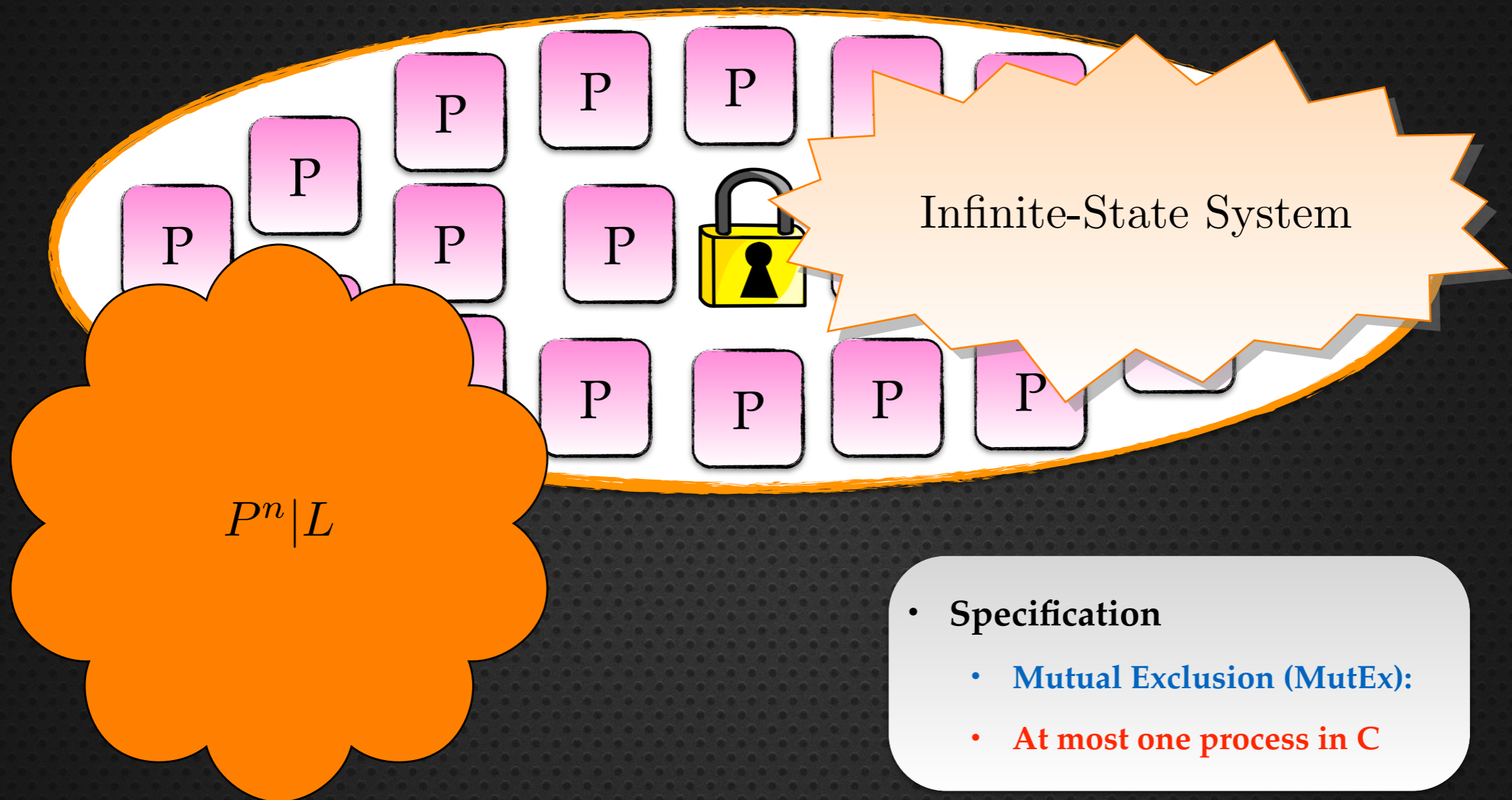
- Specification

- Mutual Exclusion (MutEx):
- At most one process in C

- Task = Parameterized Verification

- Verify correctness regardless of the number of processes
- $\forall n. (P^n | L) \models MutEx$

Parameterized Systems



- Specification

- Mutual Exclusion (MutEx):
- At most one process in C

- Task = Parameterized Verification

- Verify correctness regardless of the number of processes
- $\forall n. (P^n | L) \models MutEx$

Background

Parameterized Systems

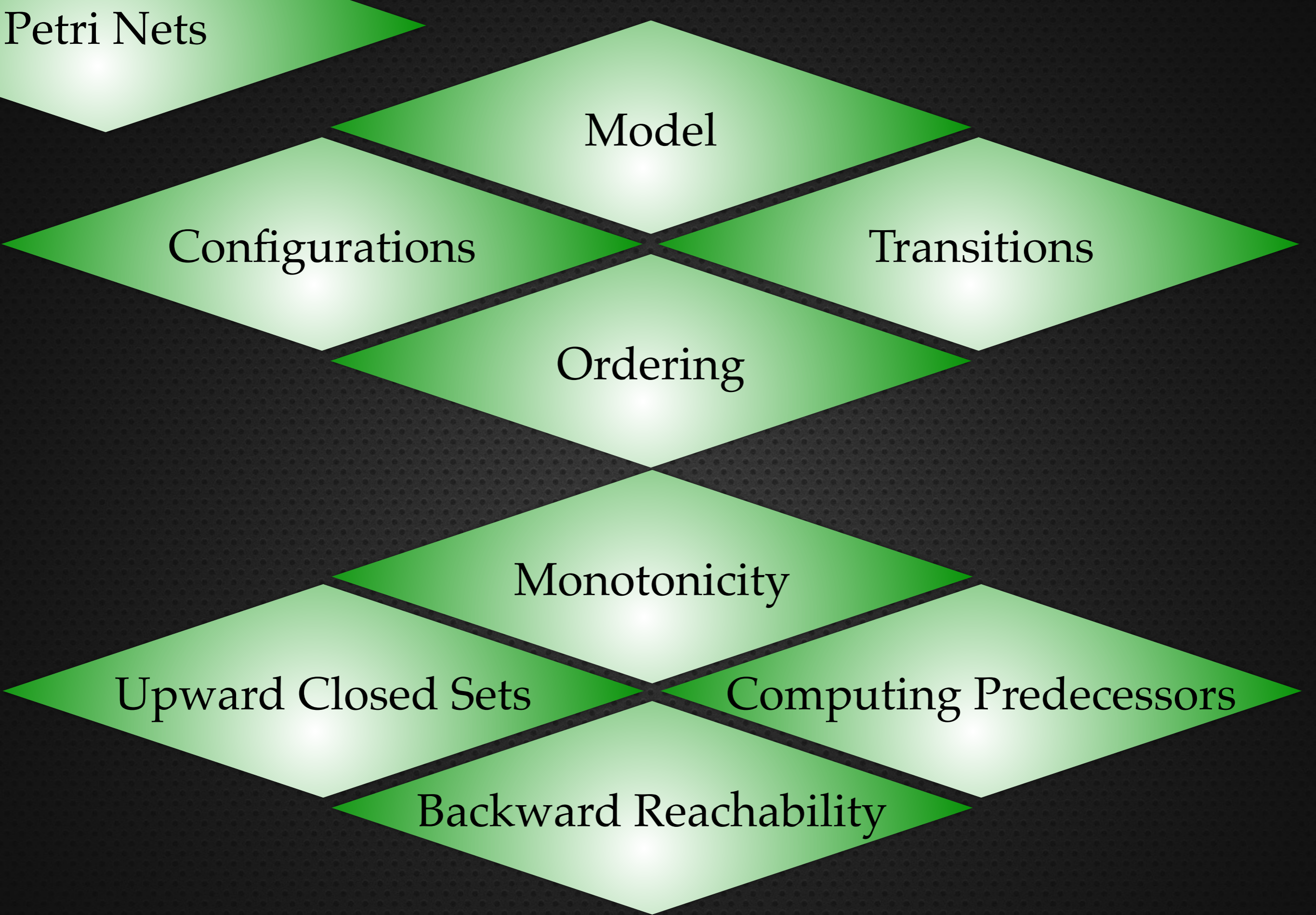
Petri Nets

Lossy Channel Systems

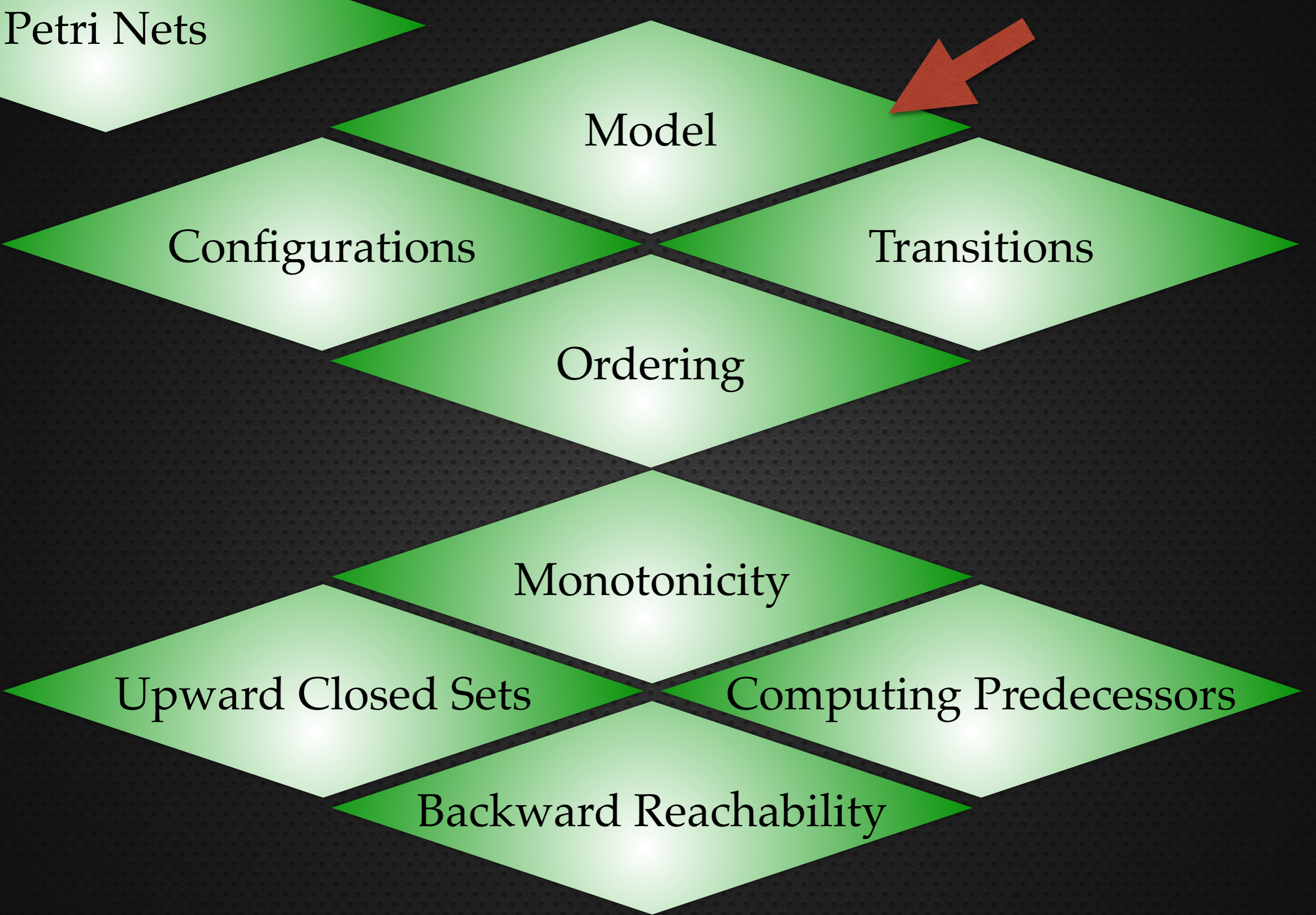
Timed Petri Nets

Petri Nets

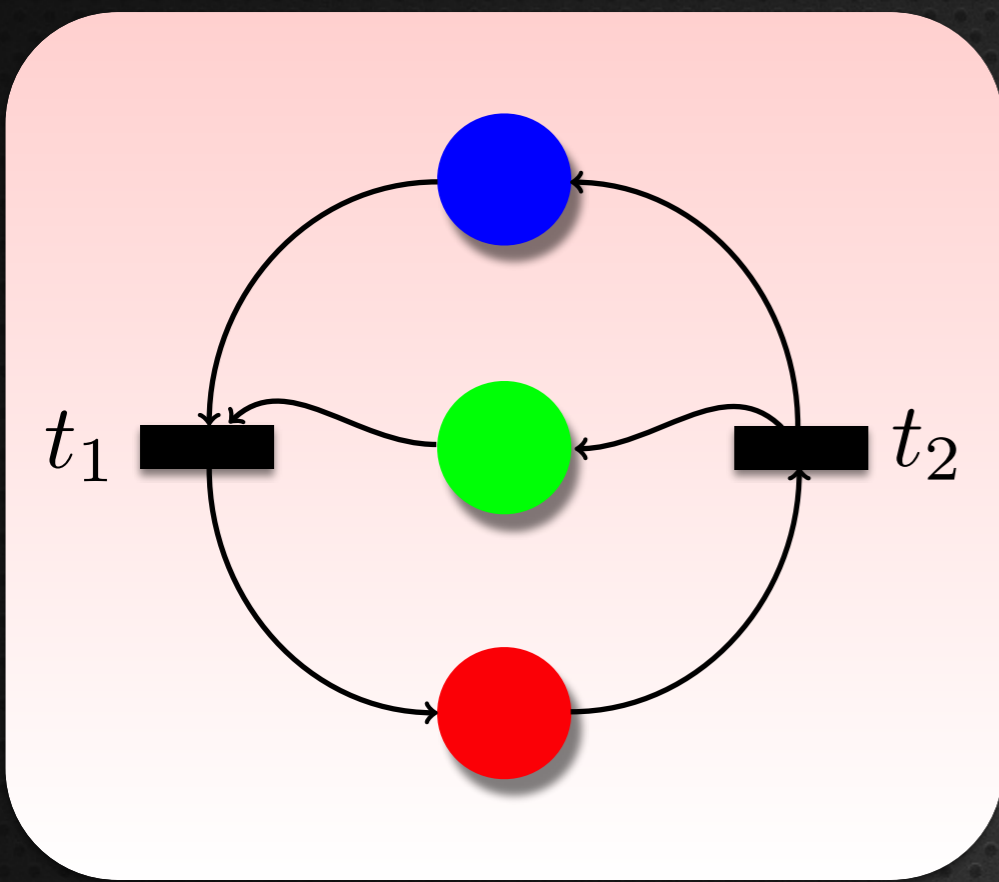
Petri Nets



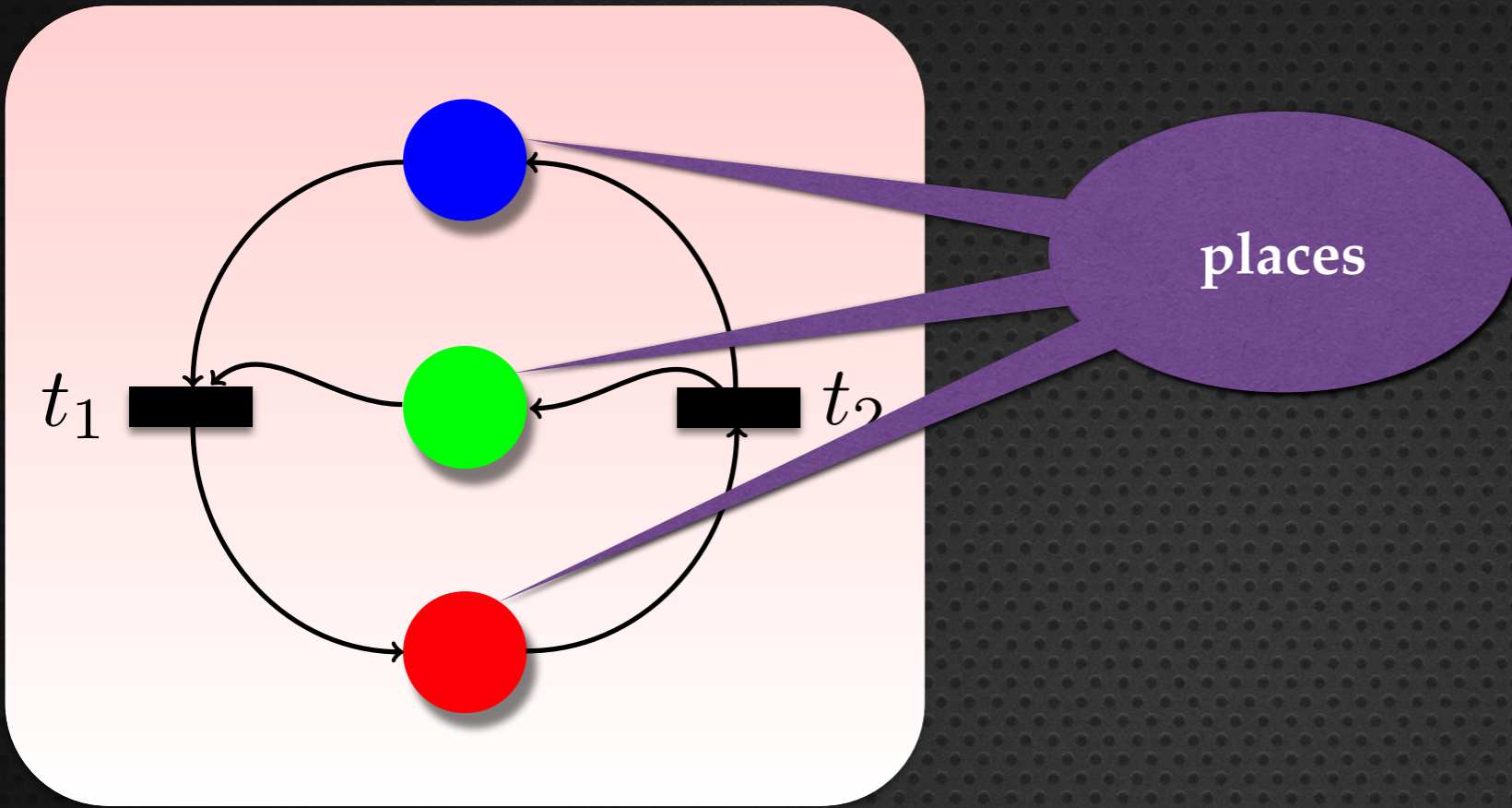
Petri Nets



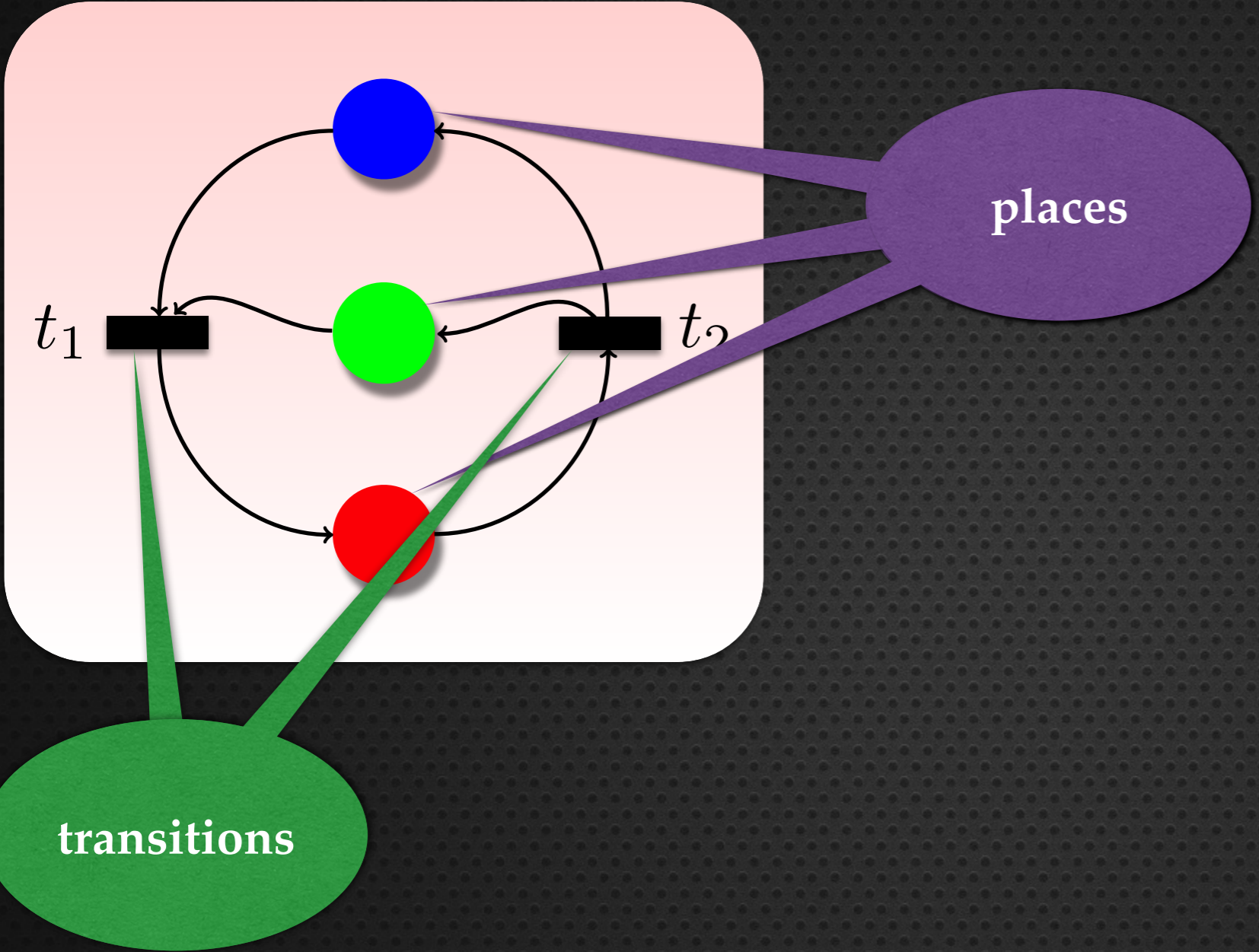
Petri Nets



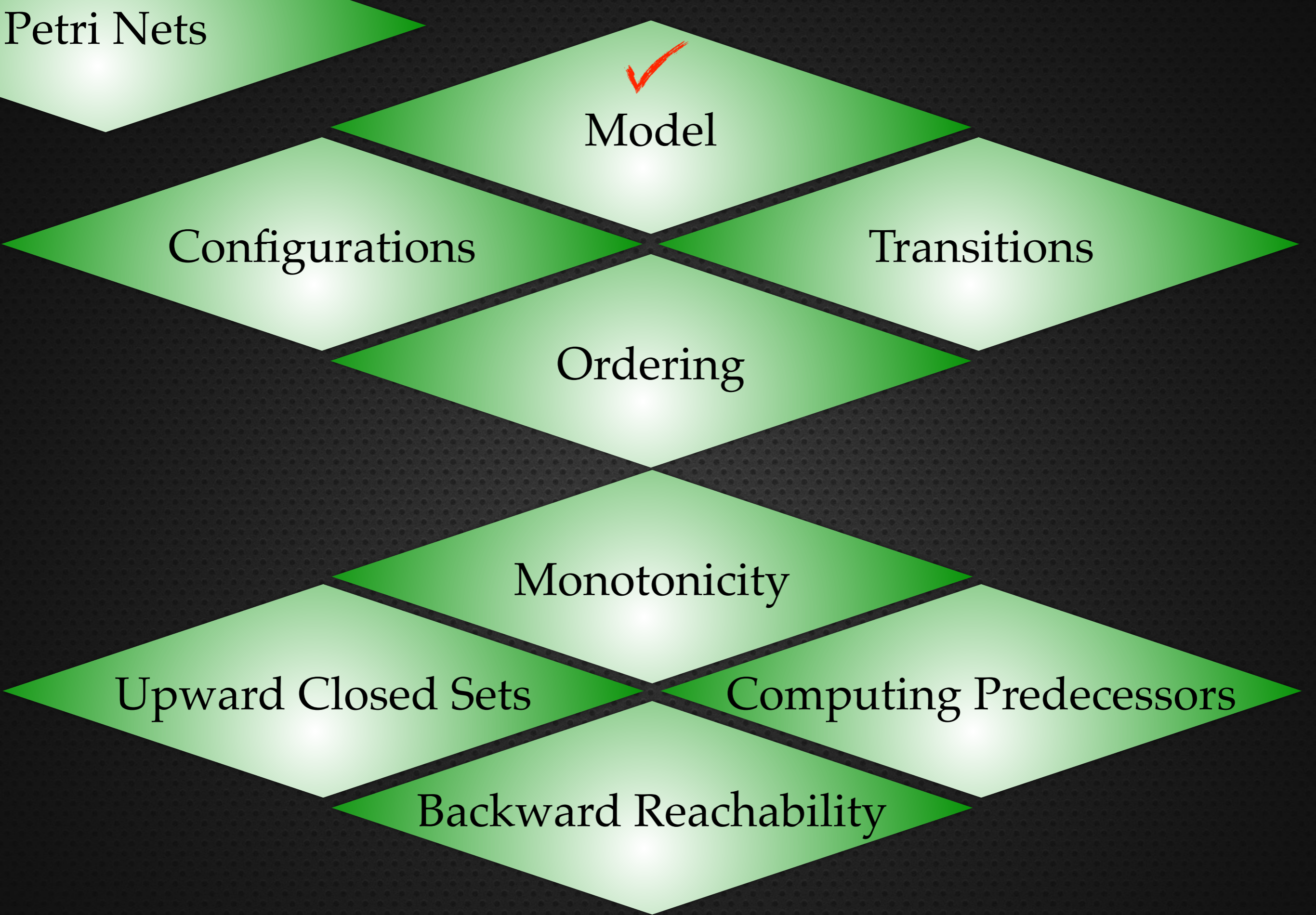
Petri Nets



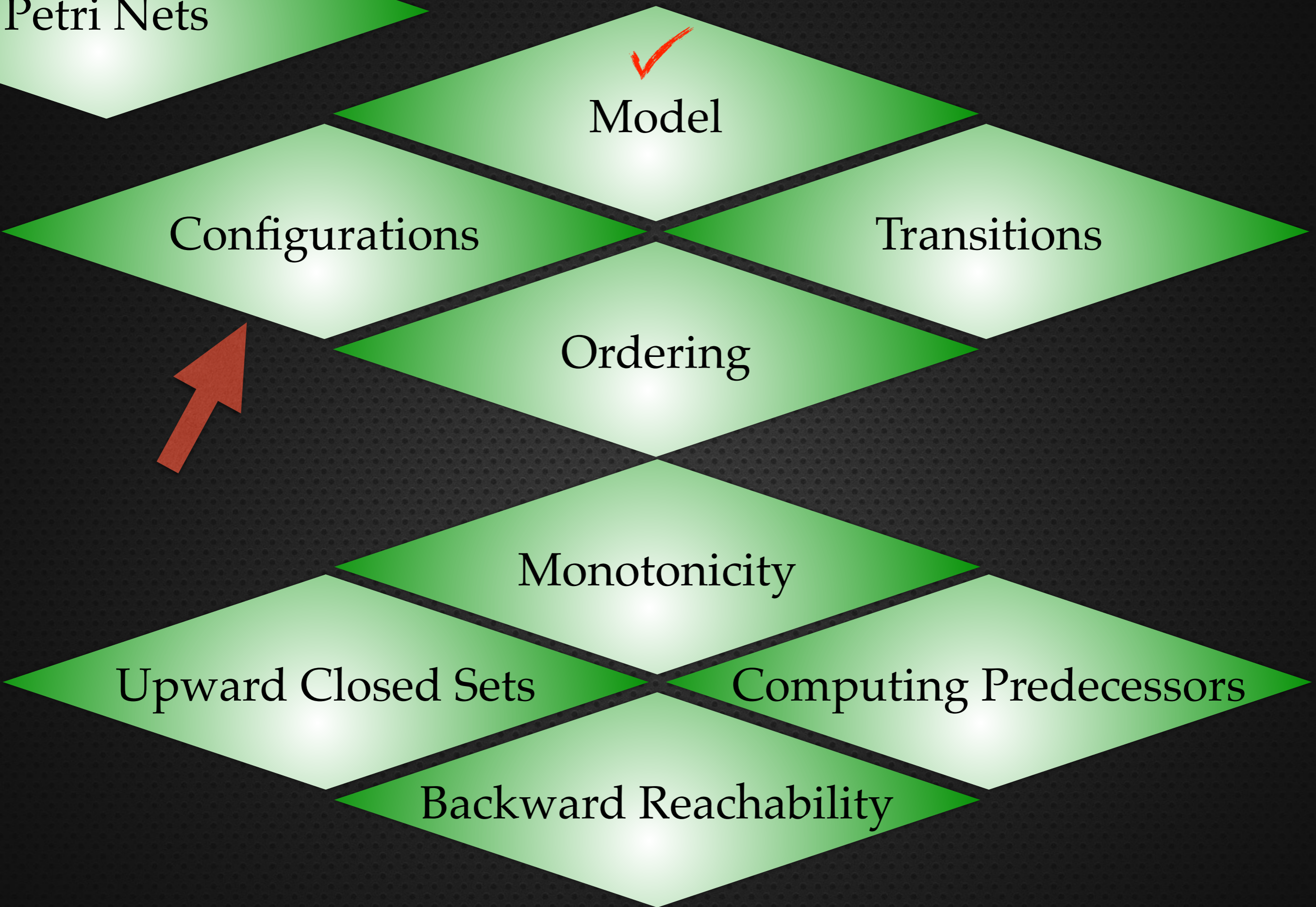
Petri Nets



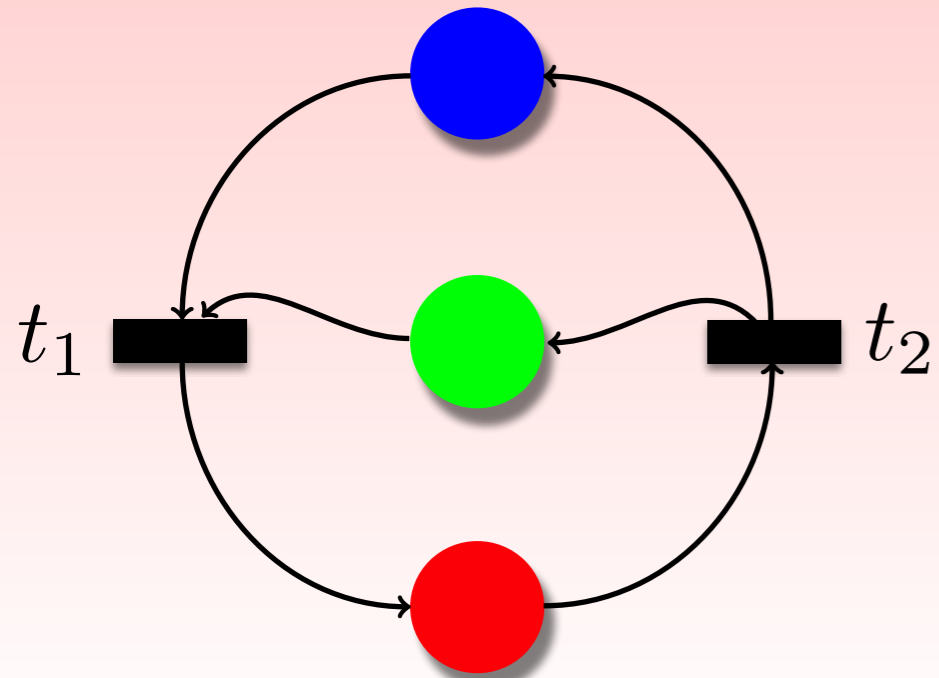
Petri Nets



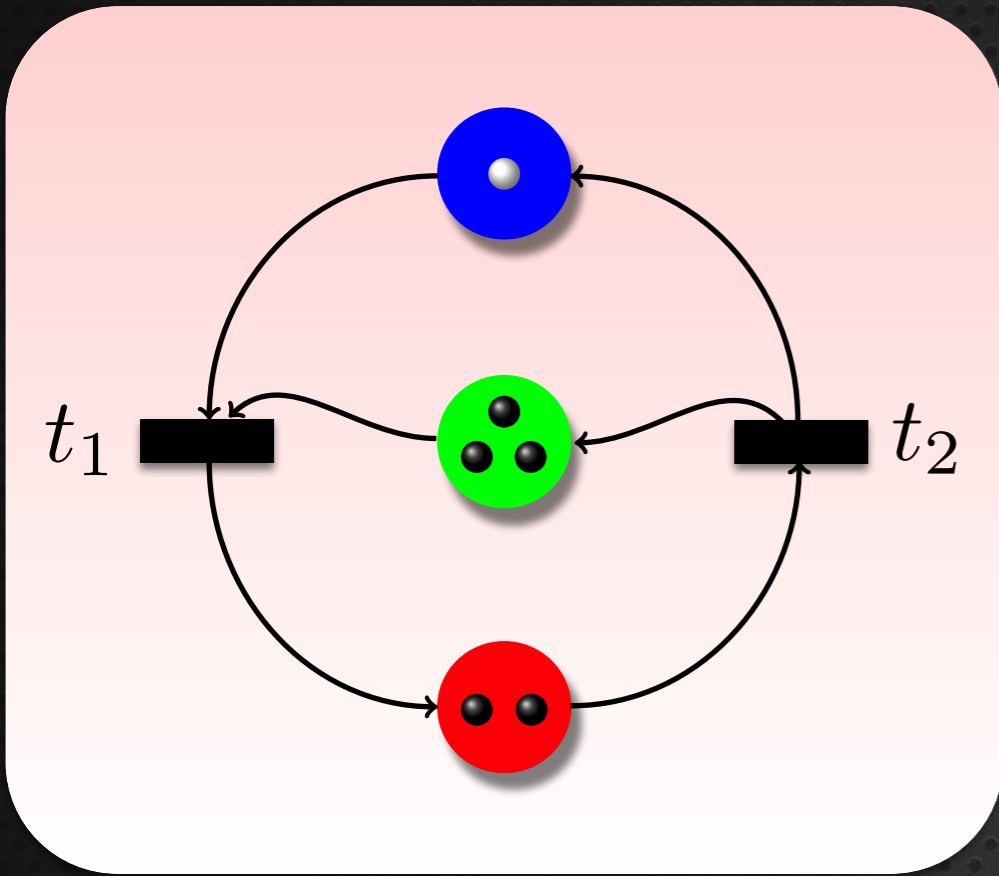
Petri Nets



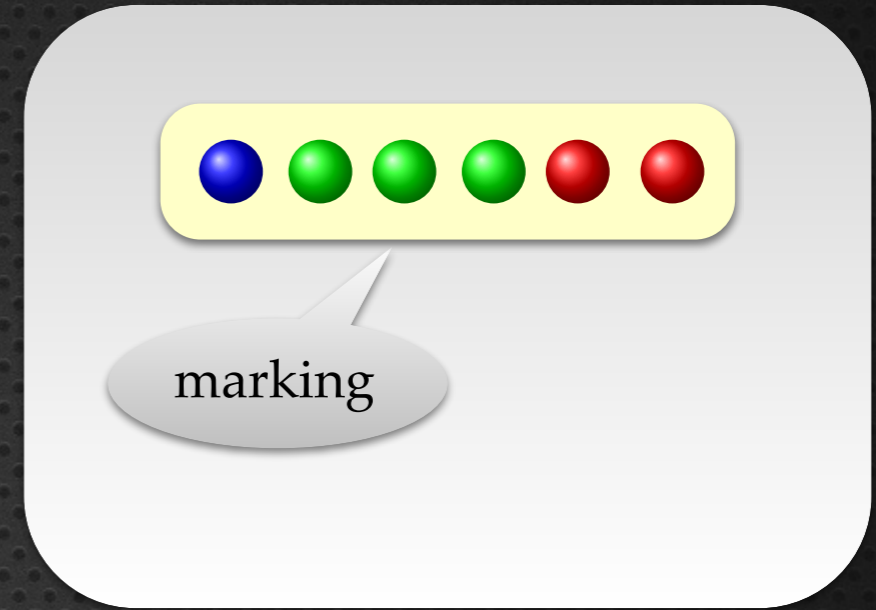
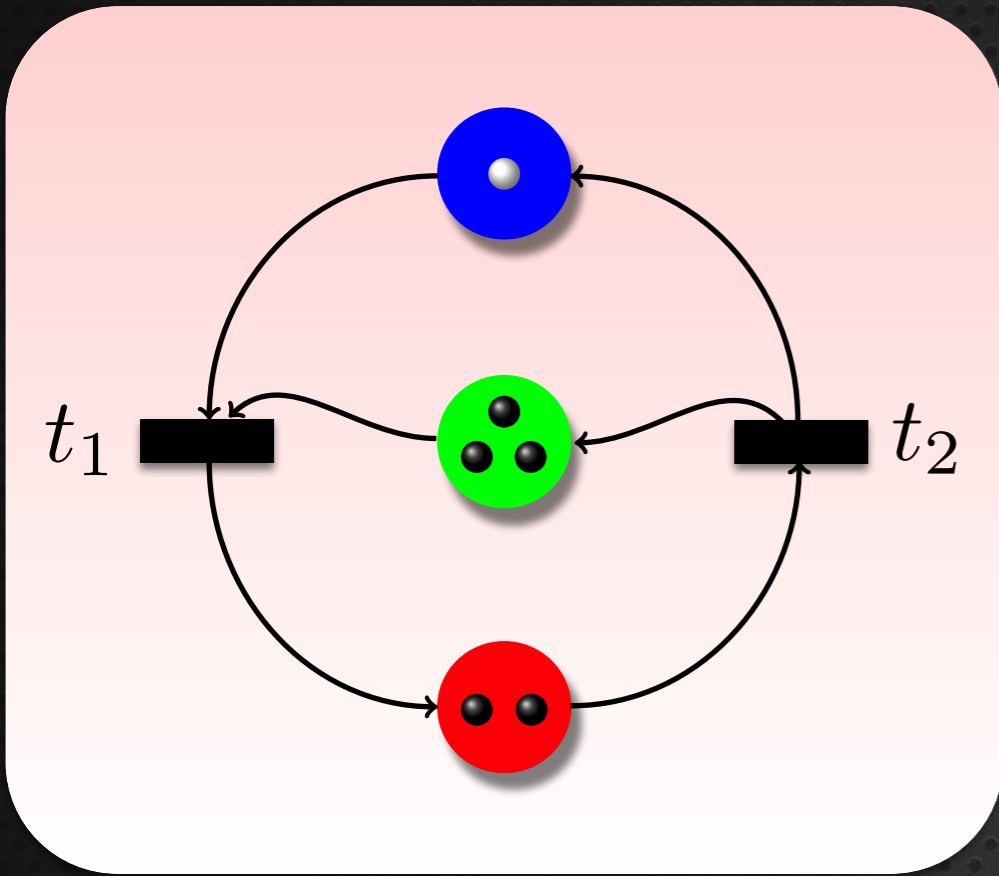
Markings



Markings

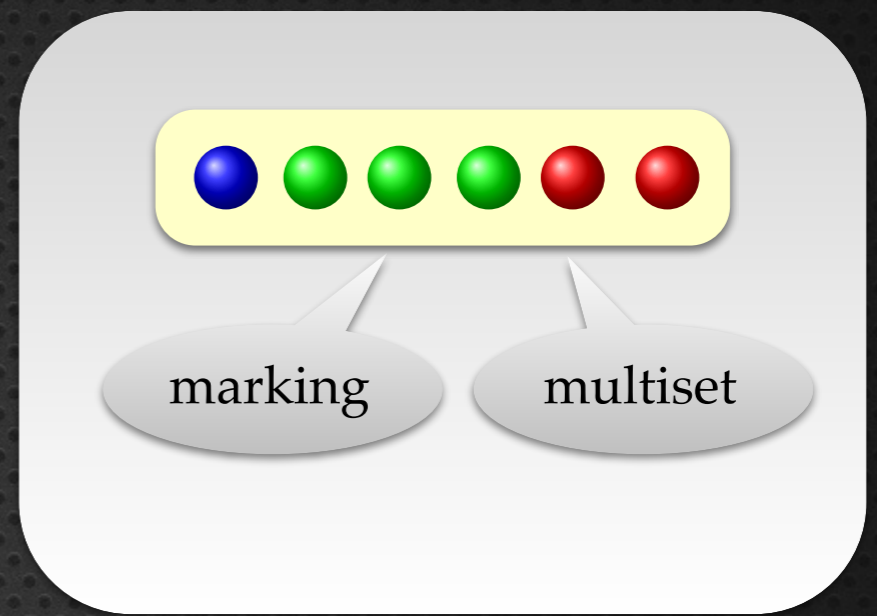
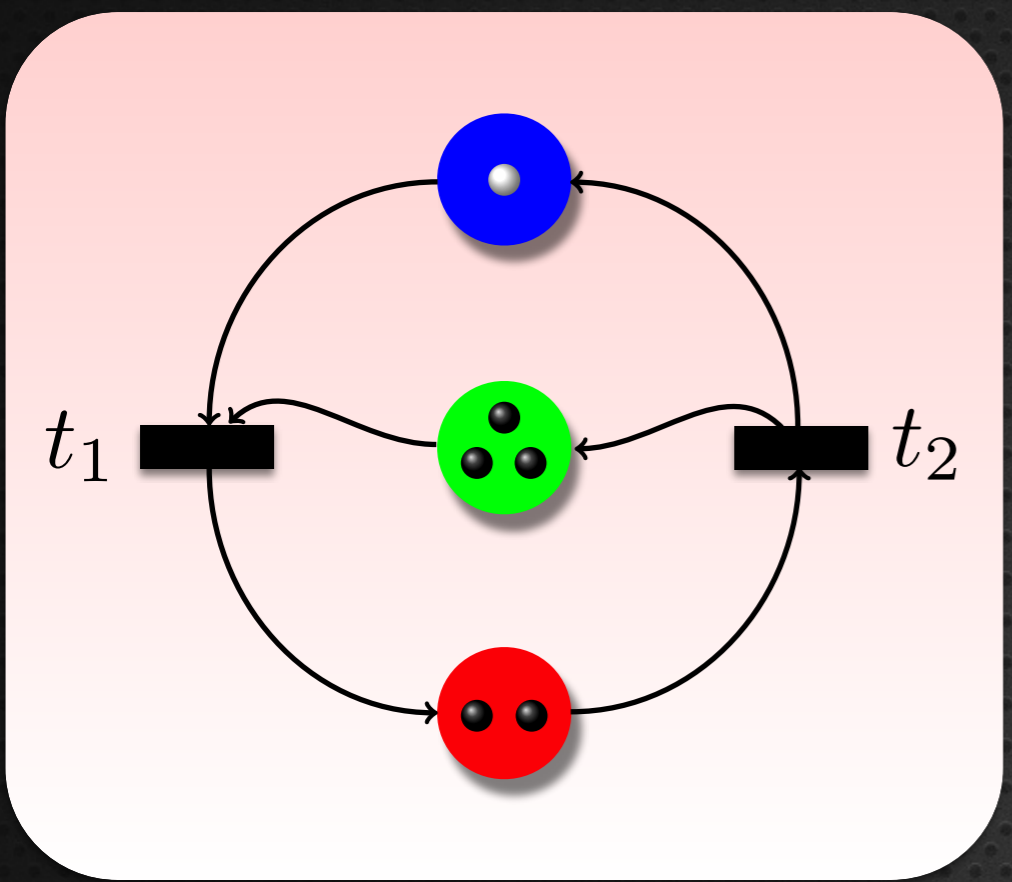


Markings

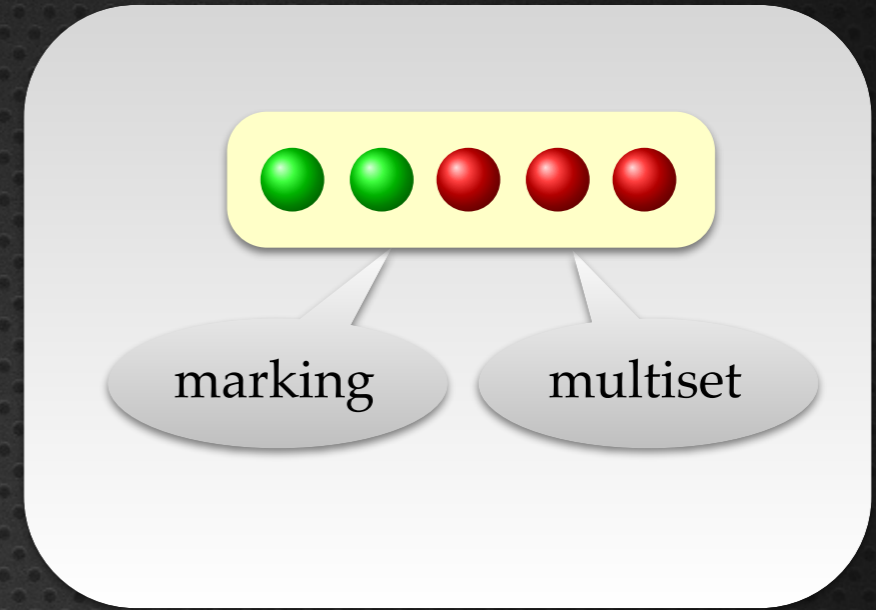
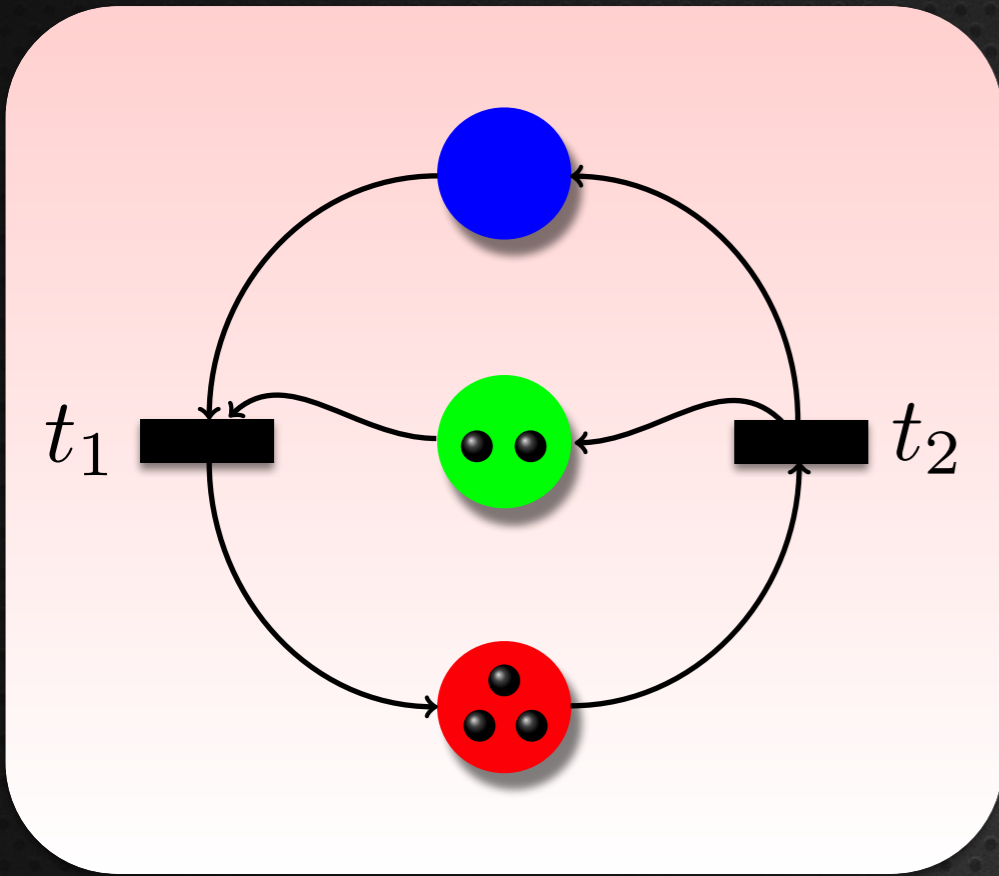


P

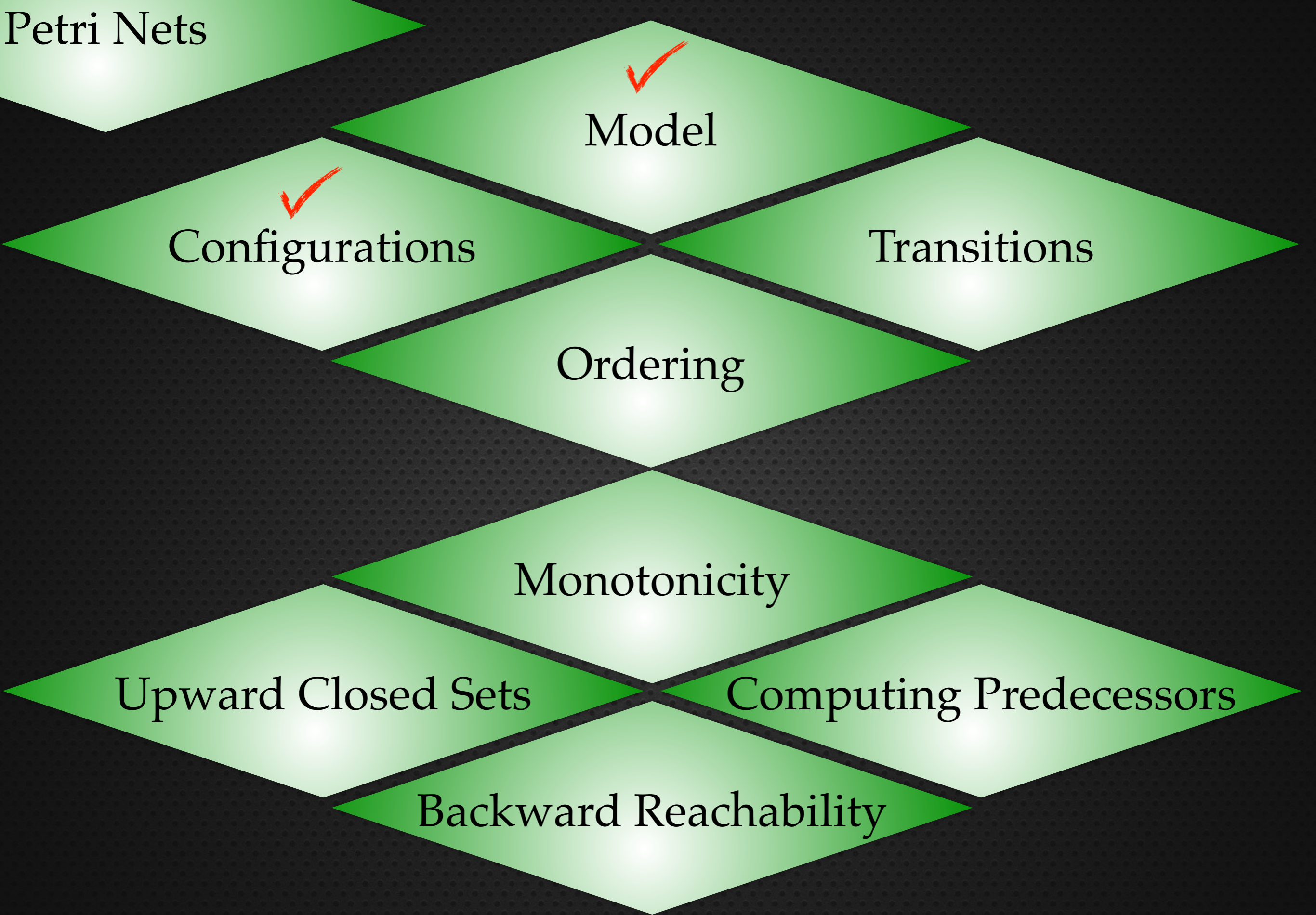
Markings



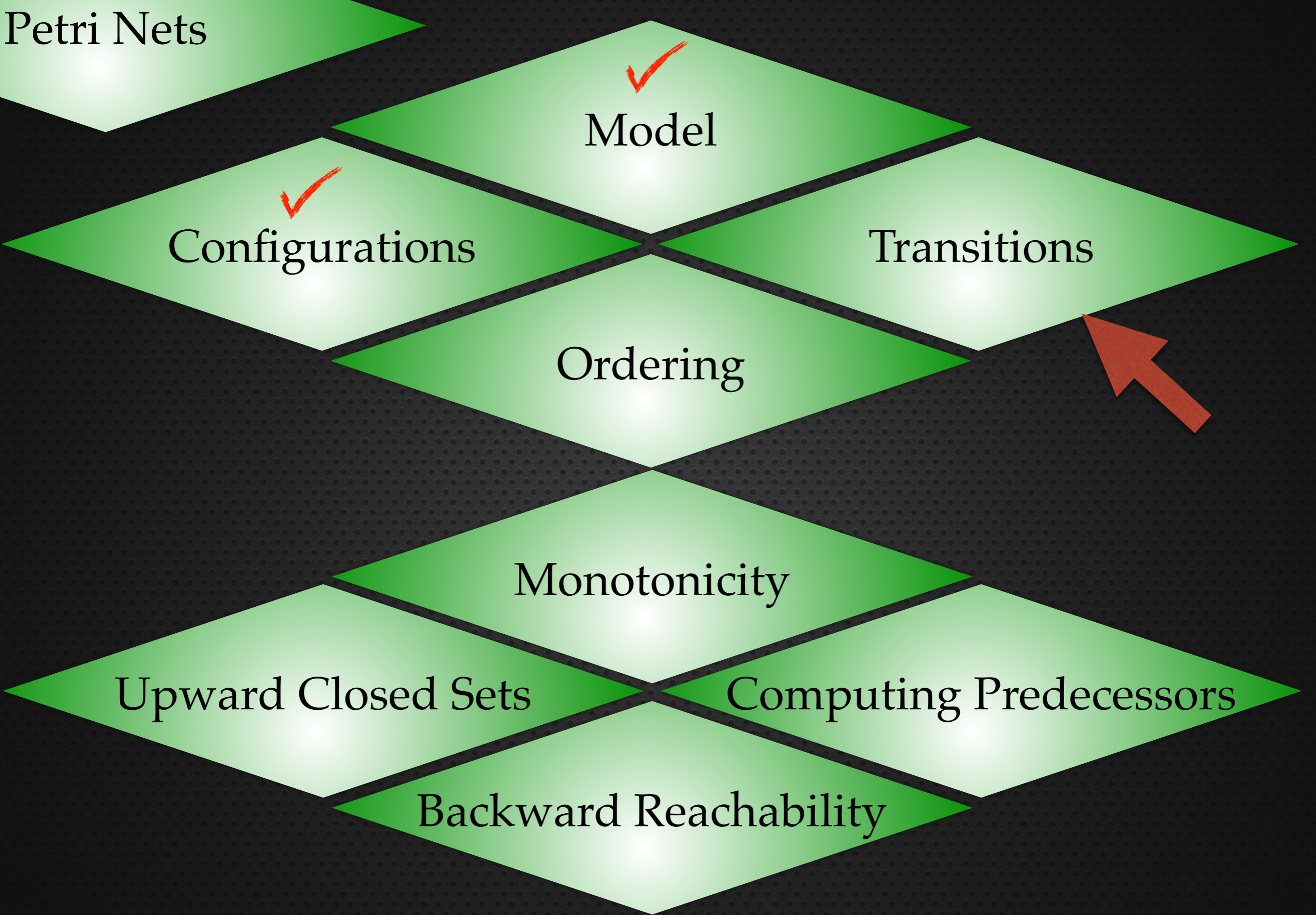
Markings



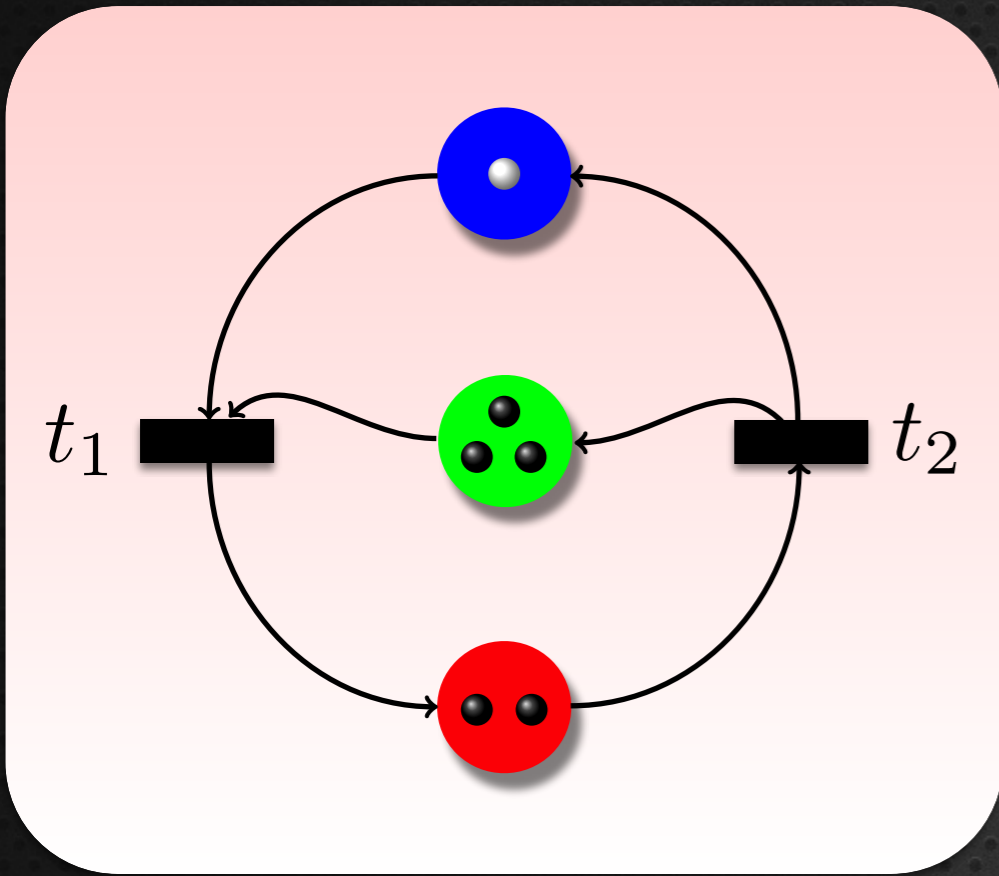
Petri Nets



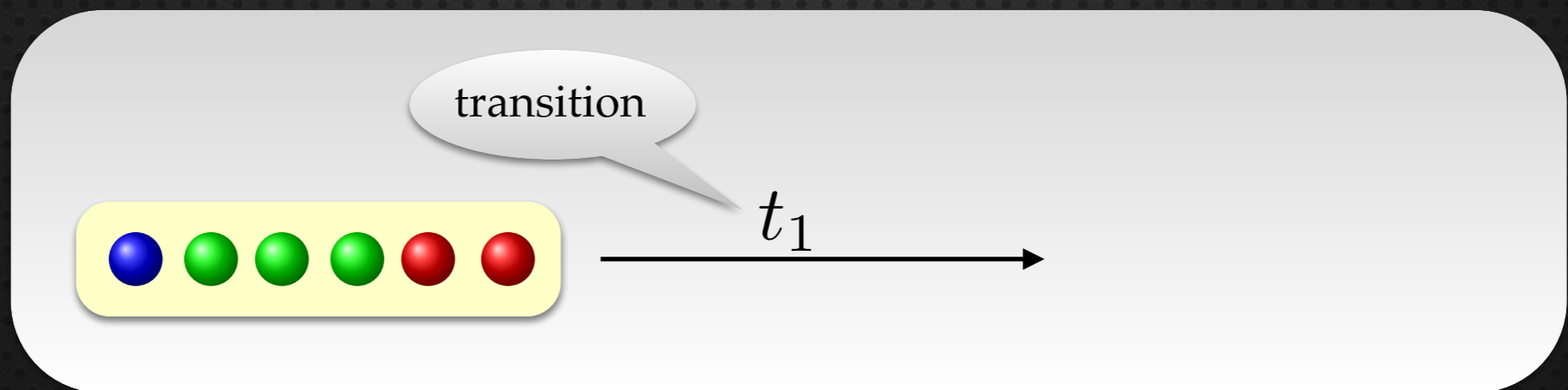
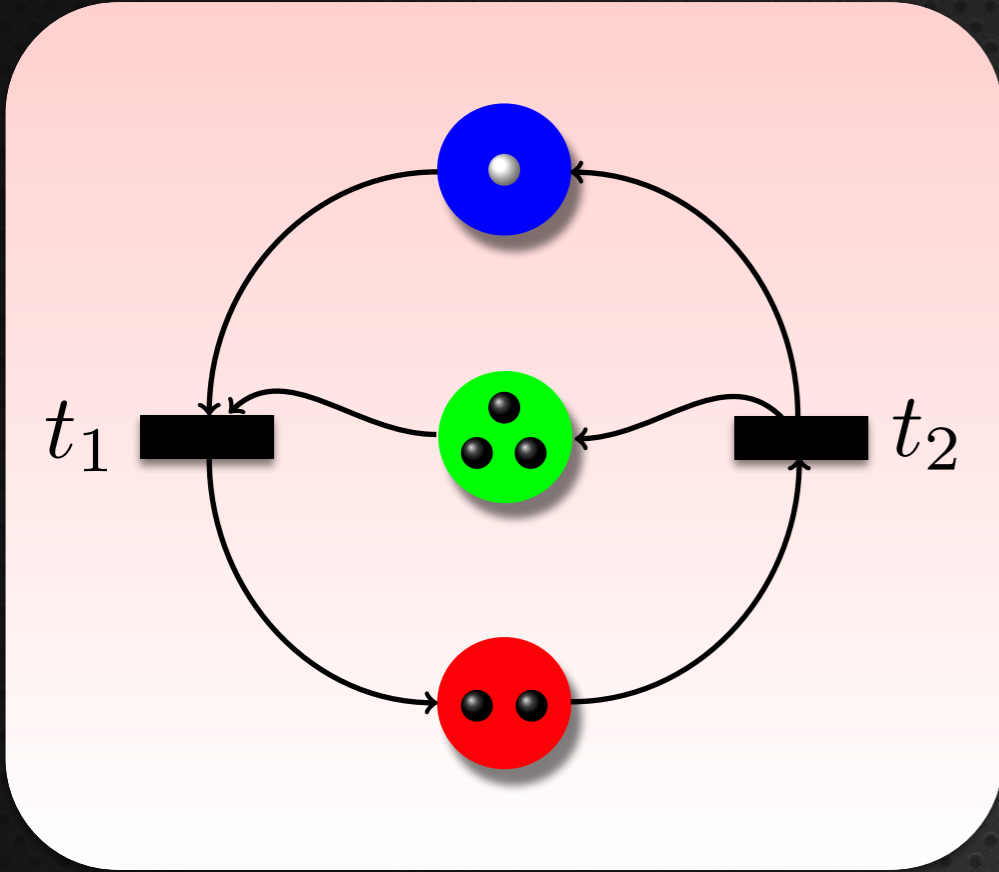
Petri Nets



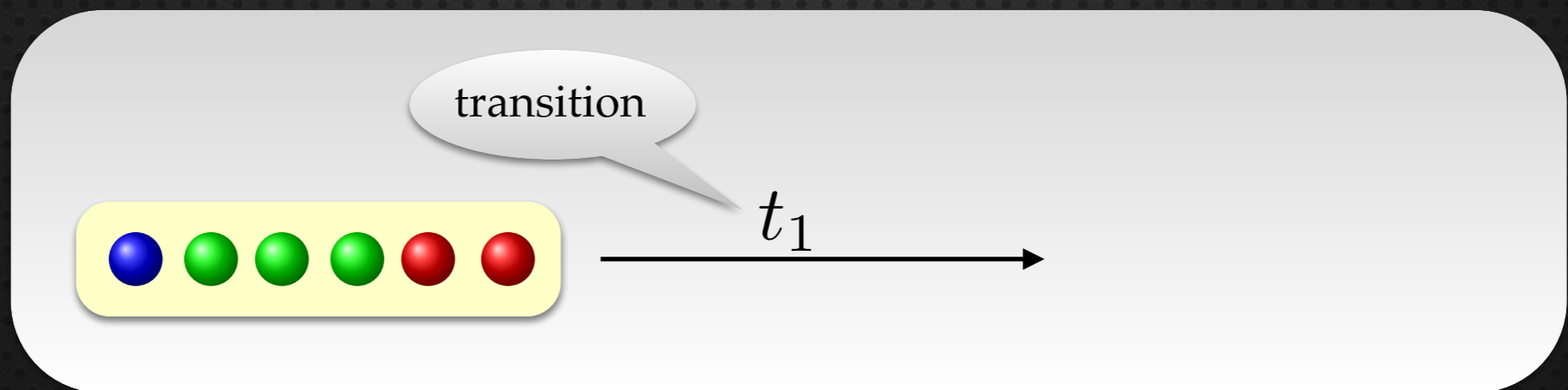
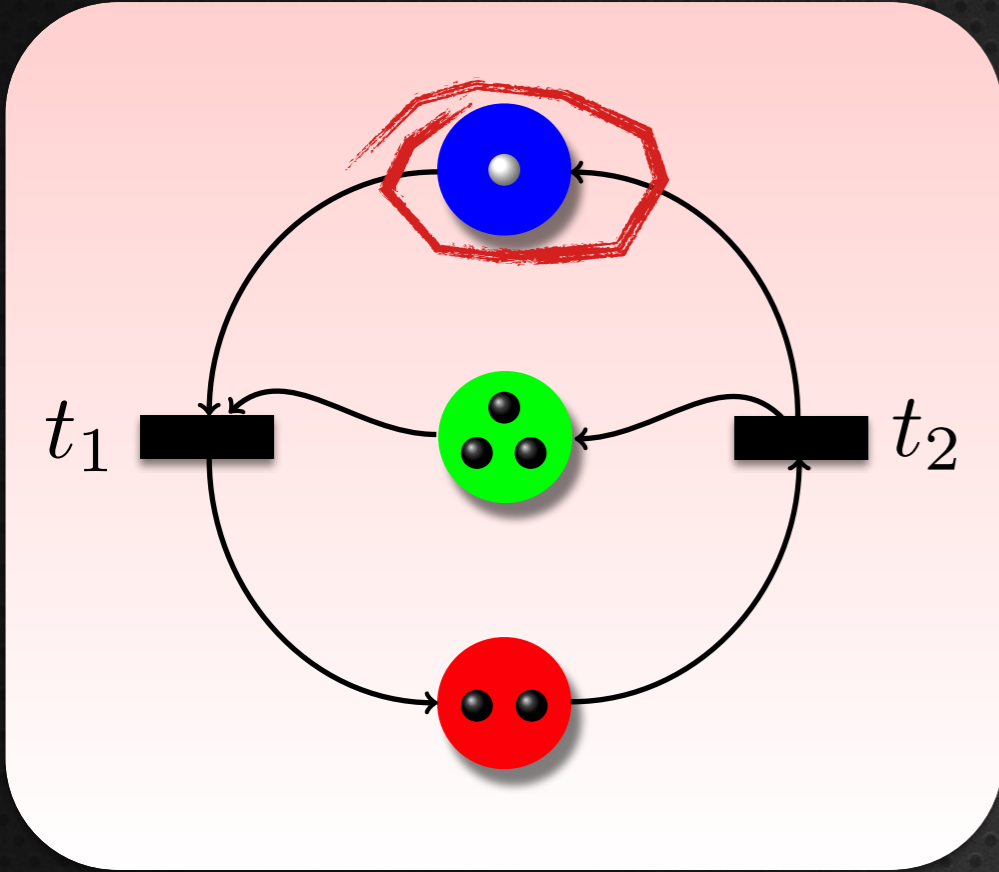
Transitions



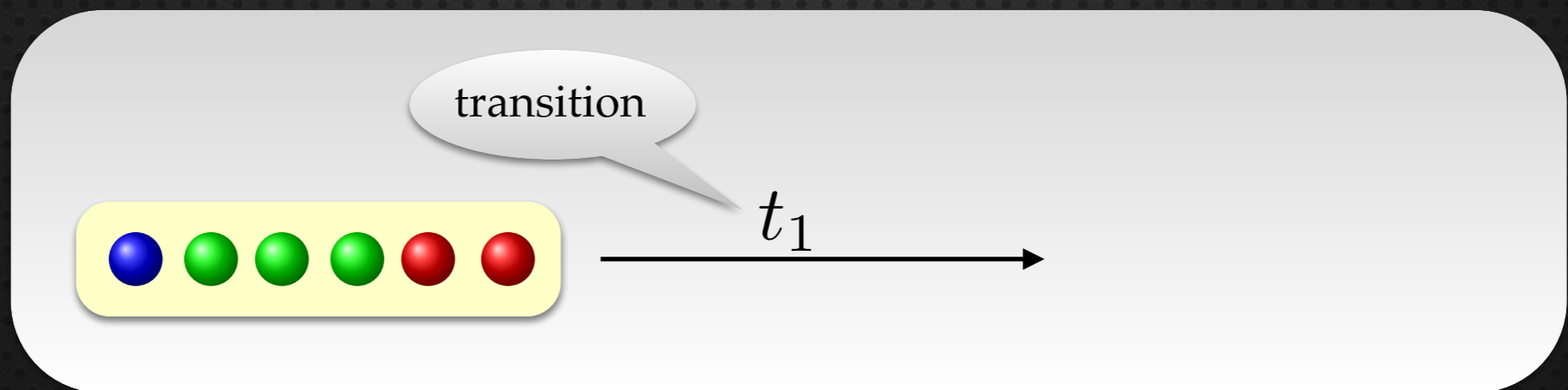
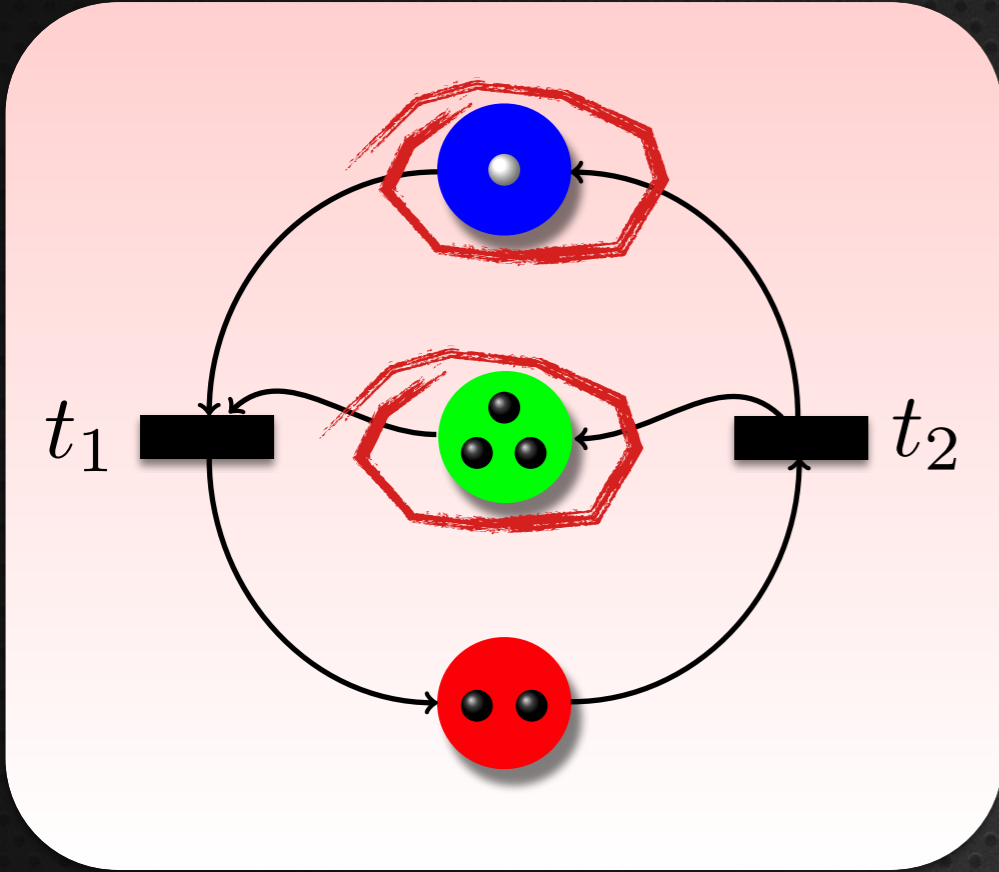
Transitions



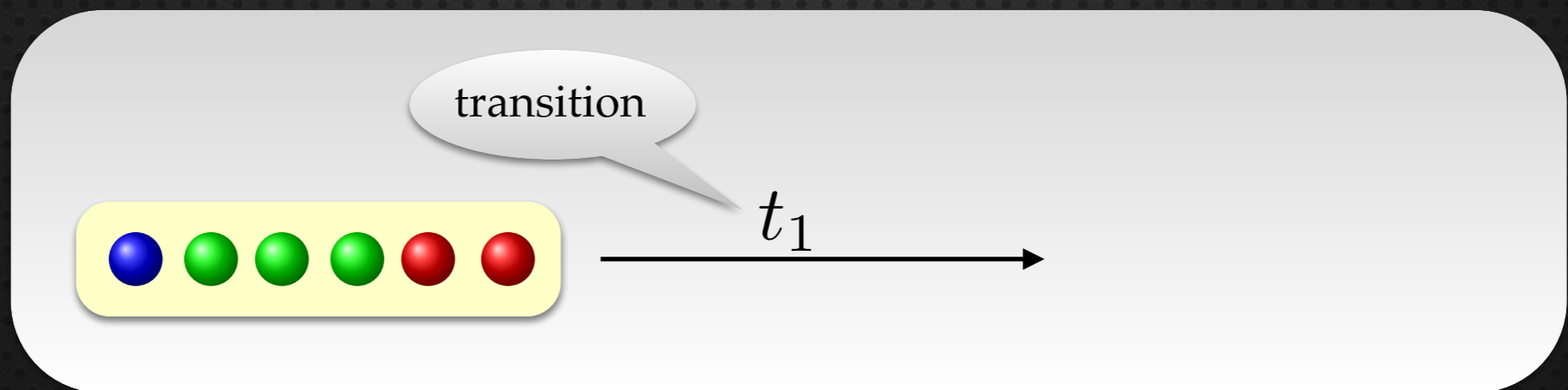
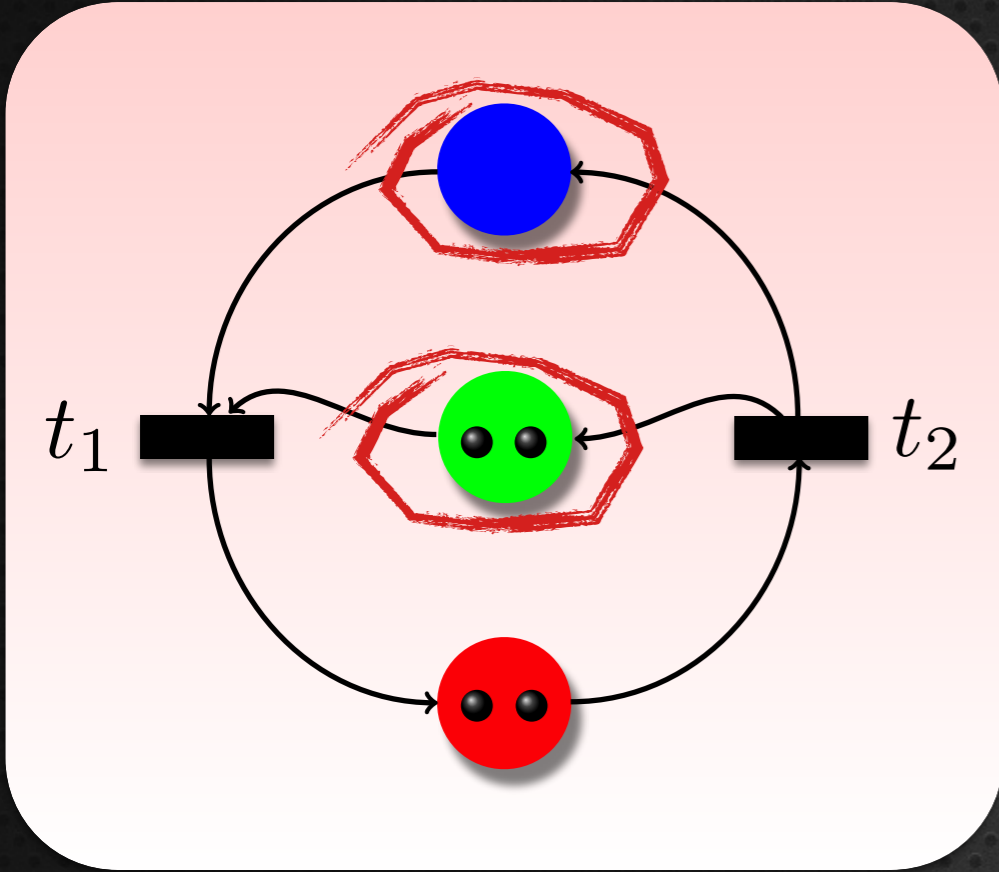
Transitions



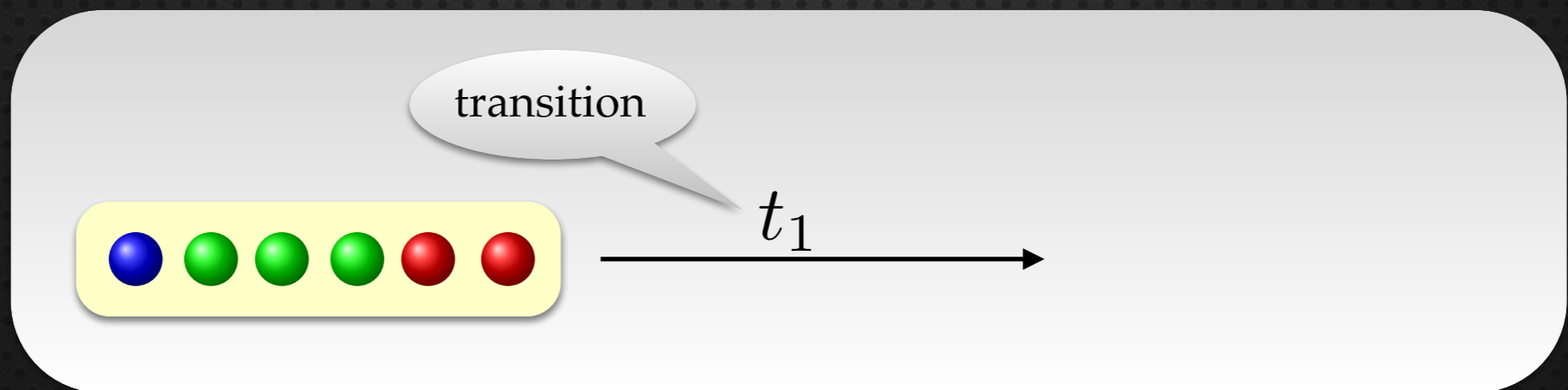
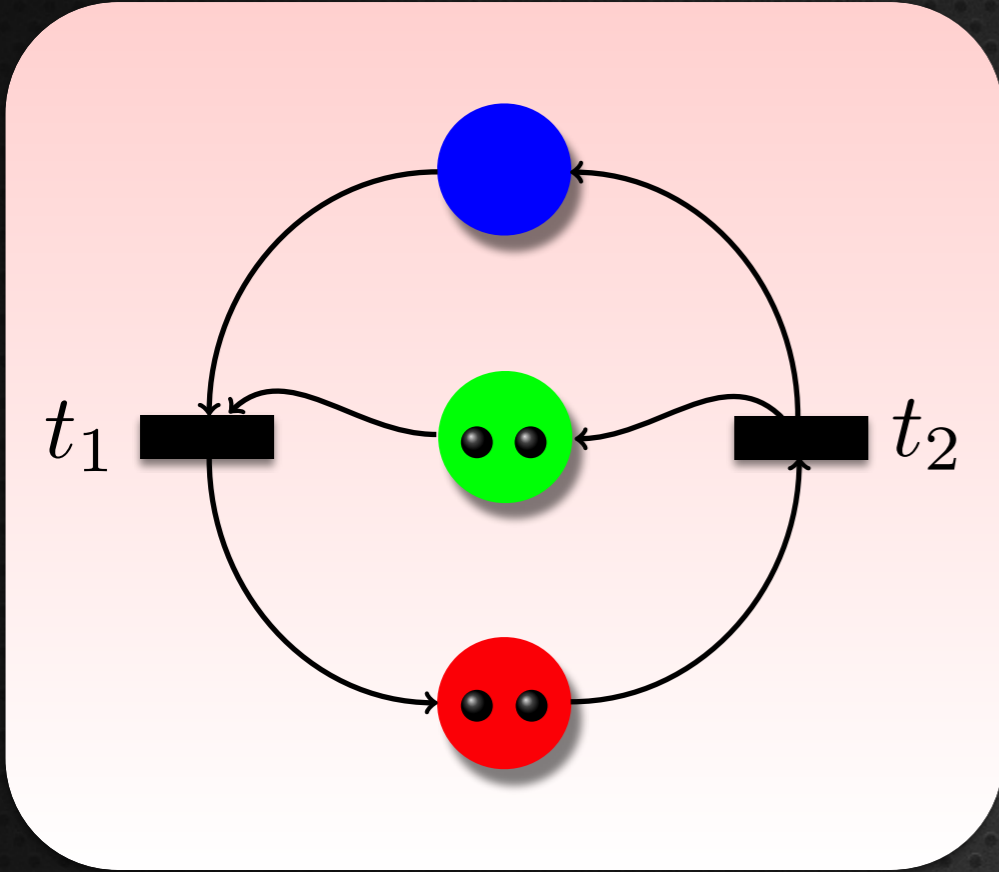
Transitions



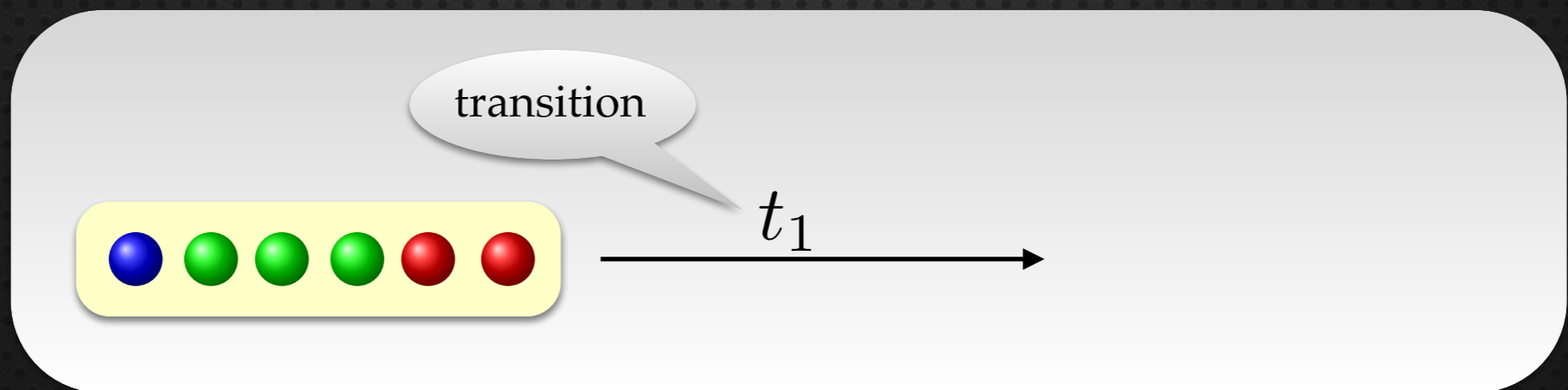
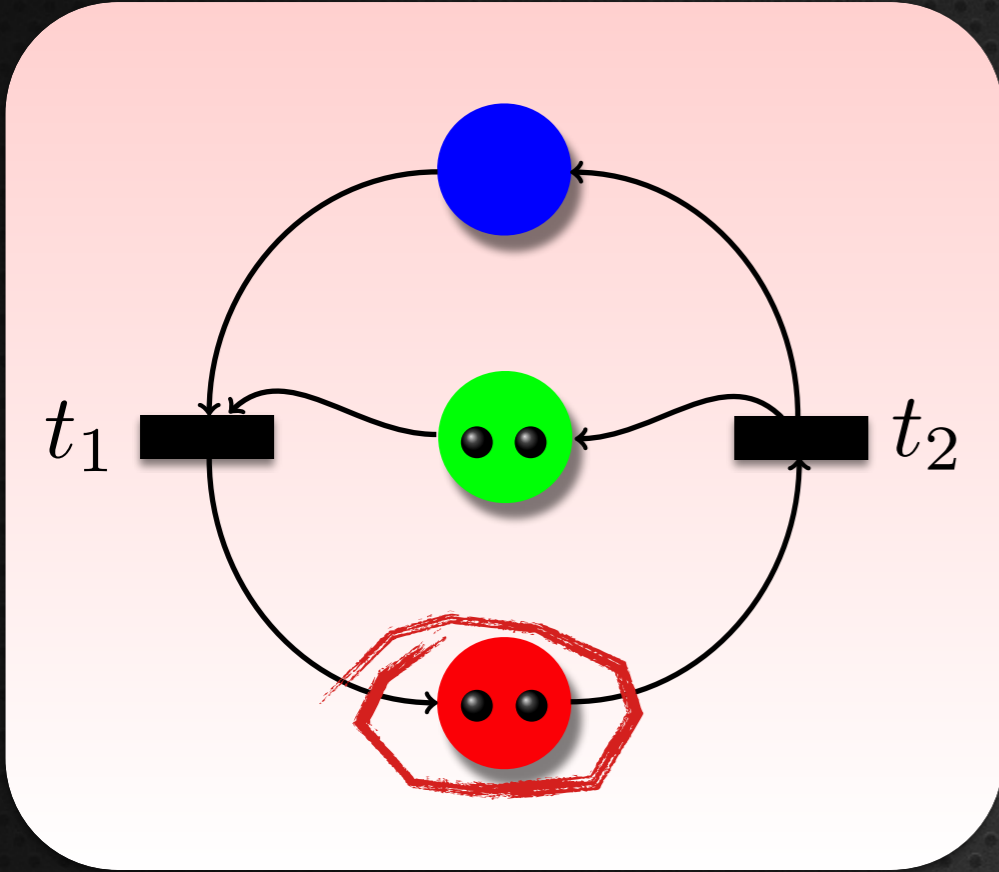
Transitions



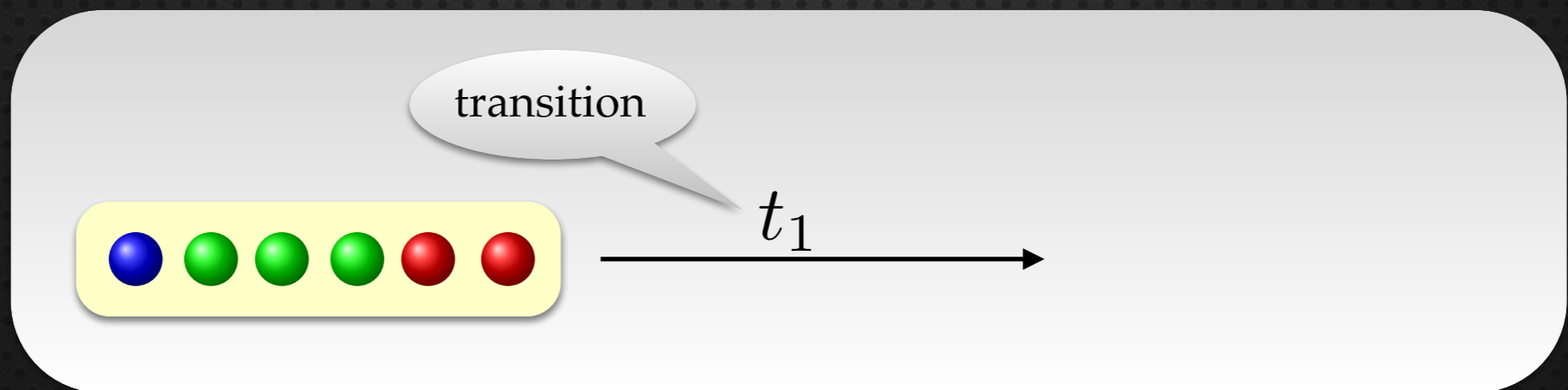
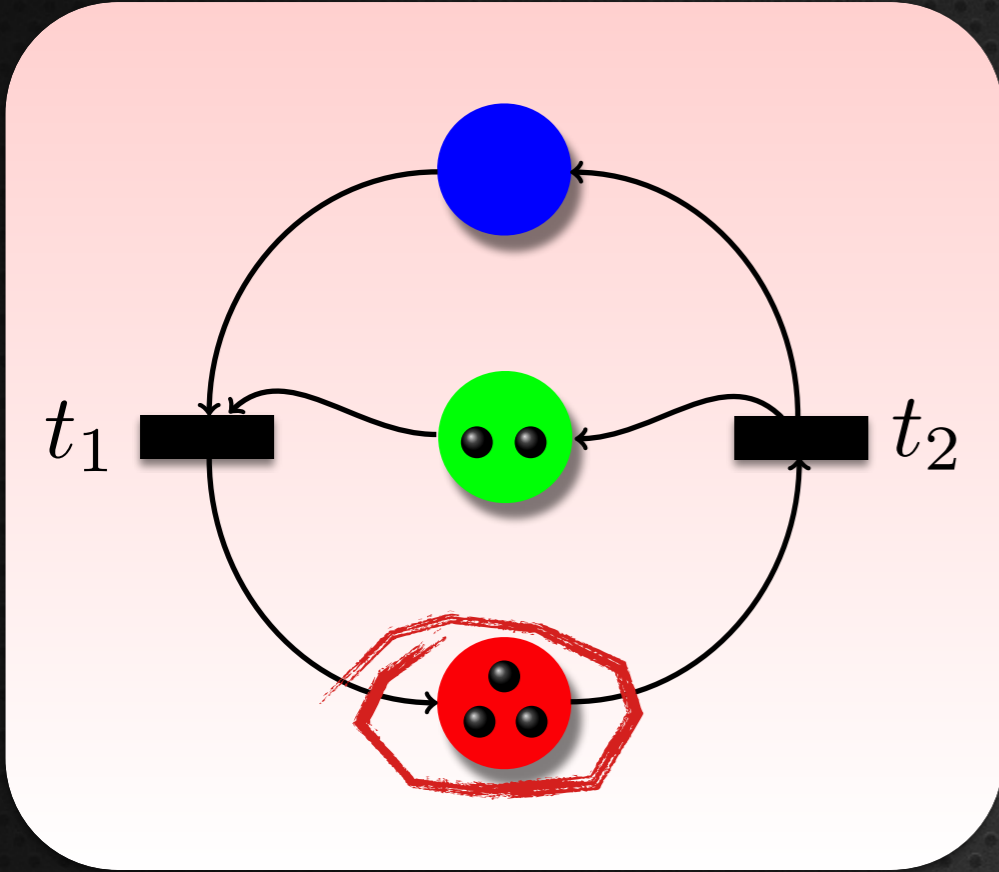
Transitions



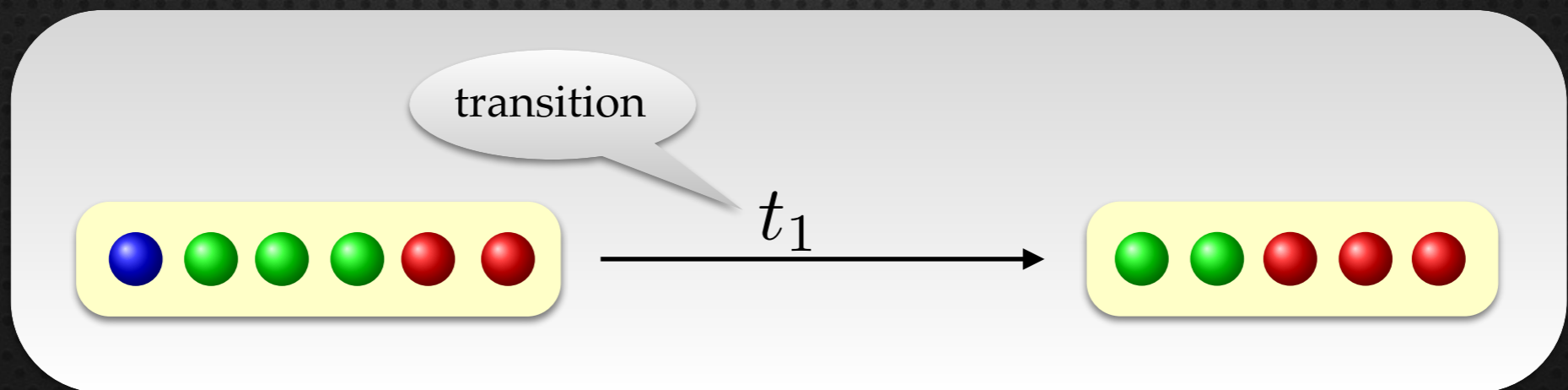
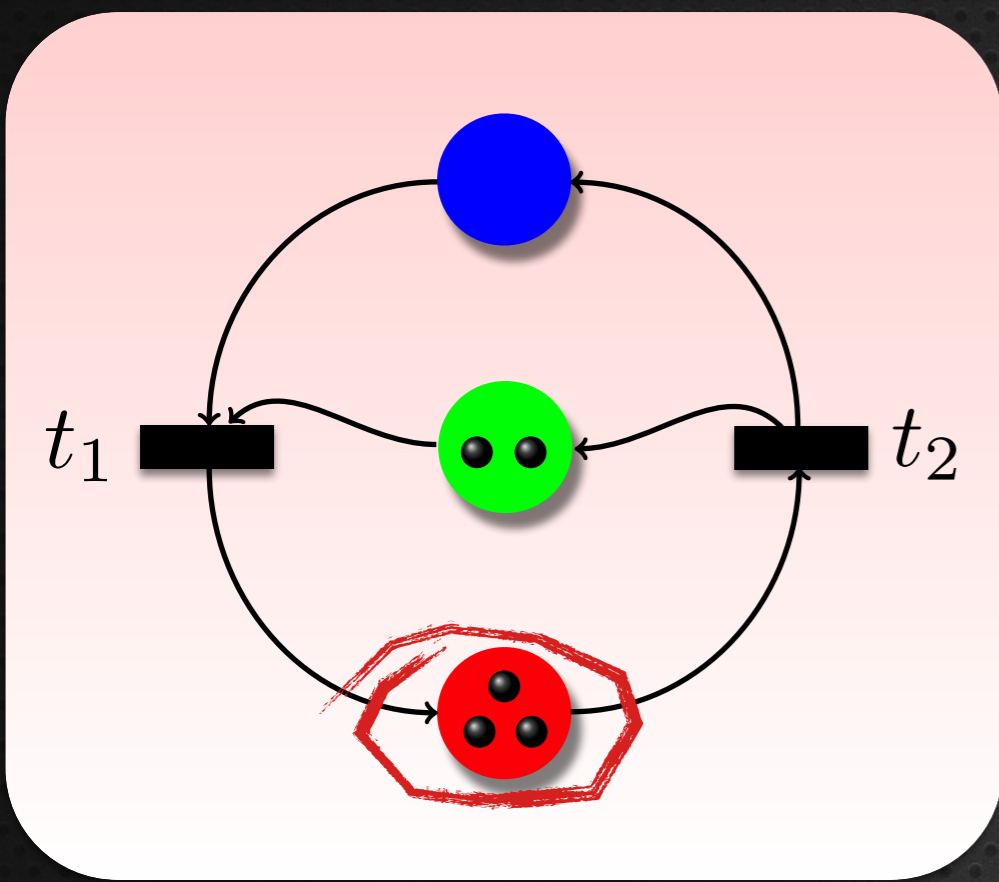
Transitions



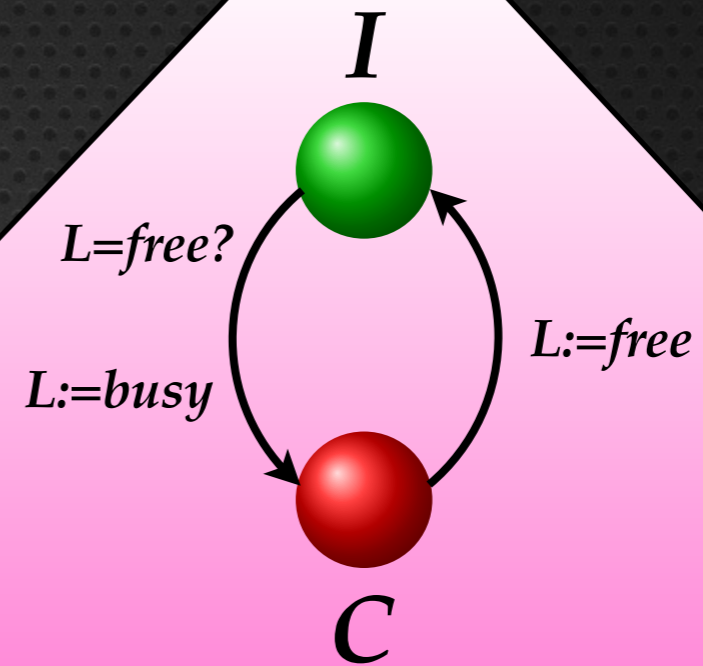
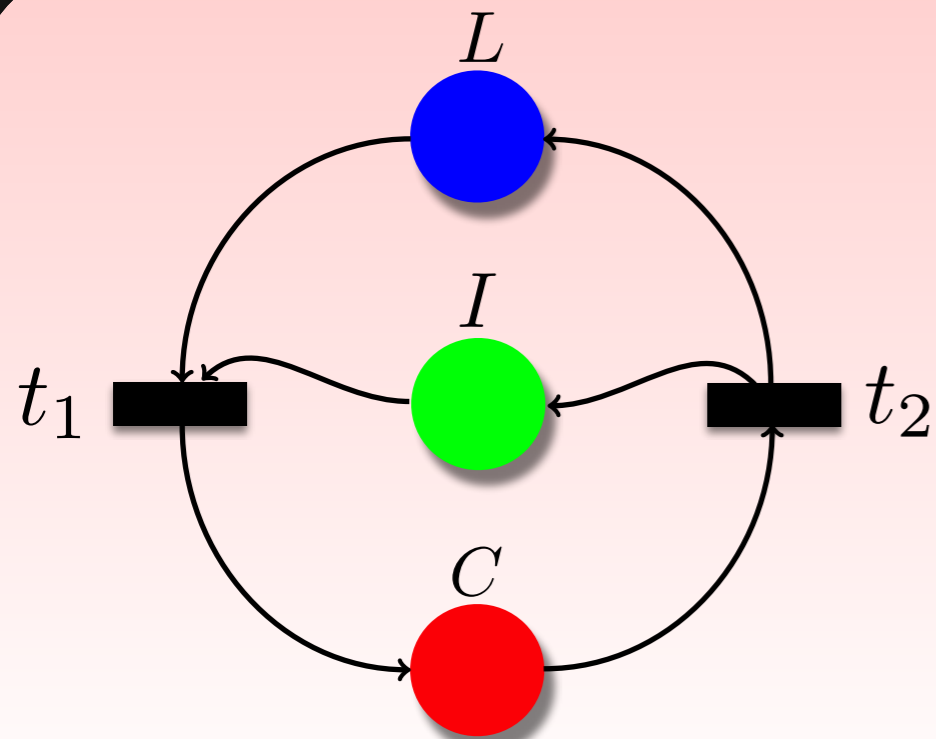
Transitions








Transitions



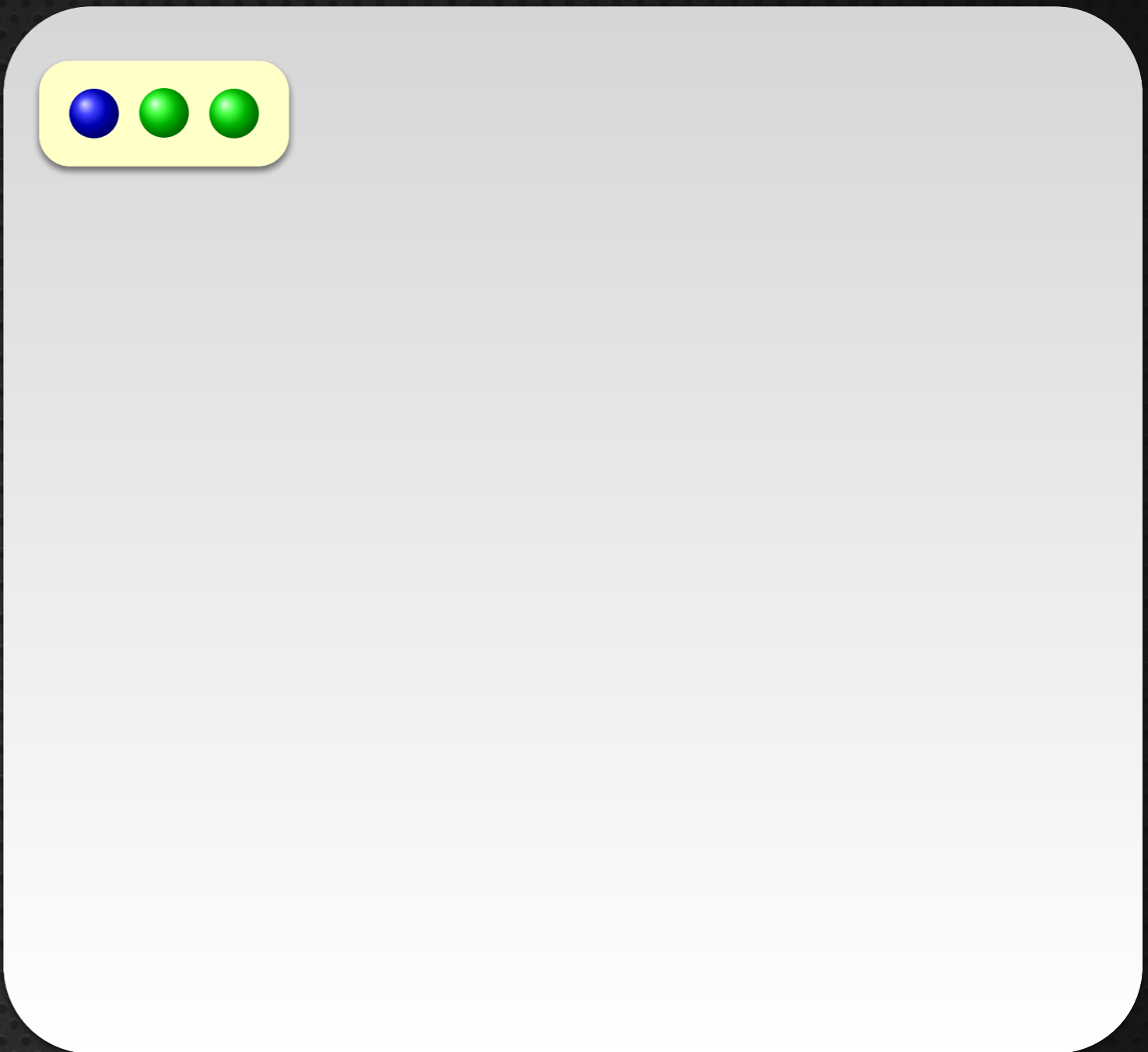
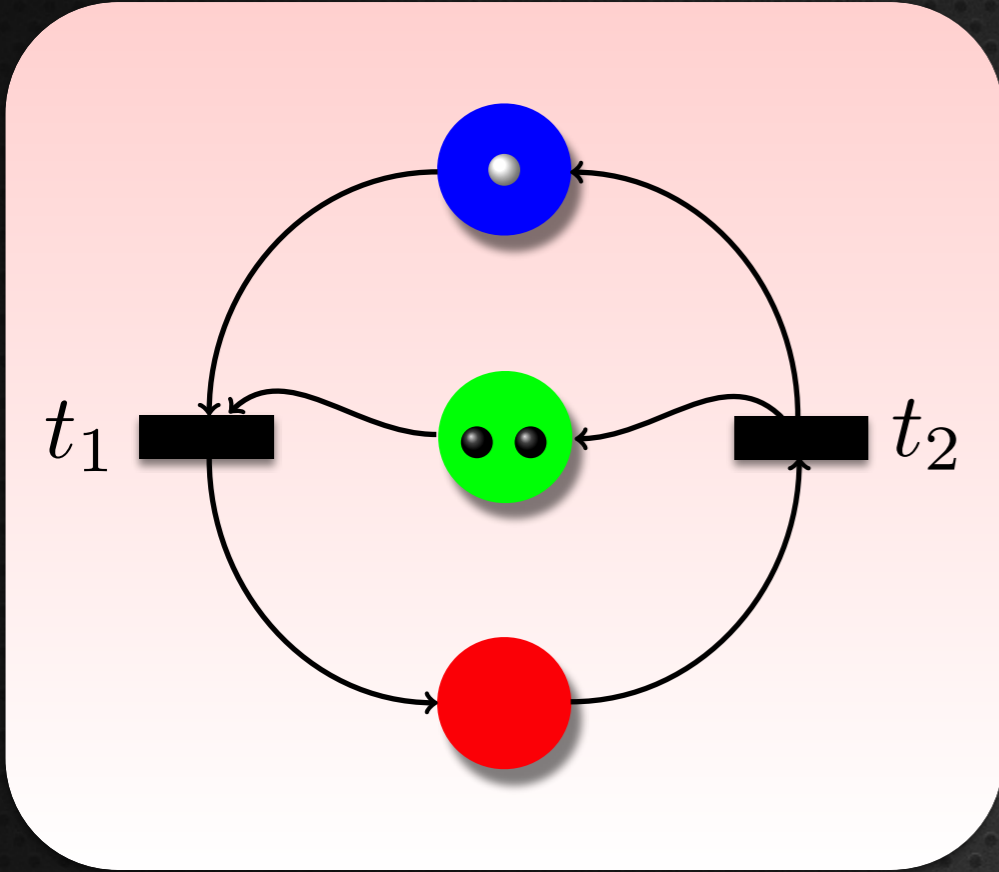
Modeling



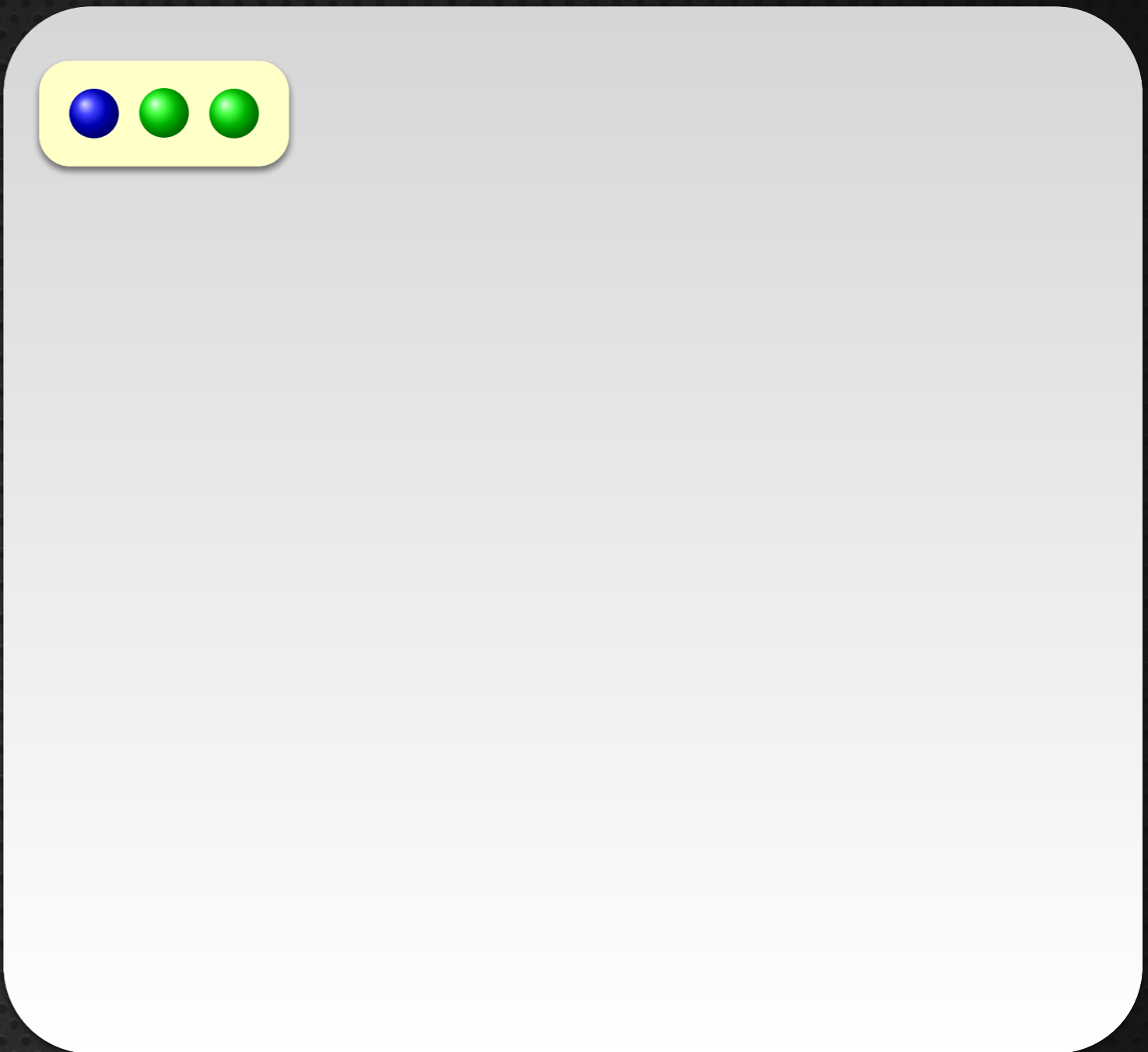
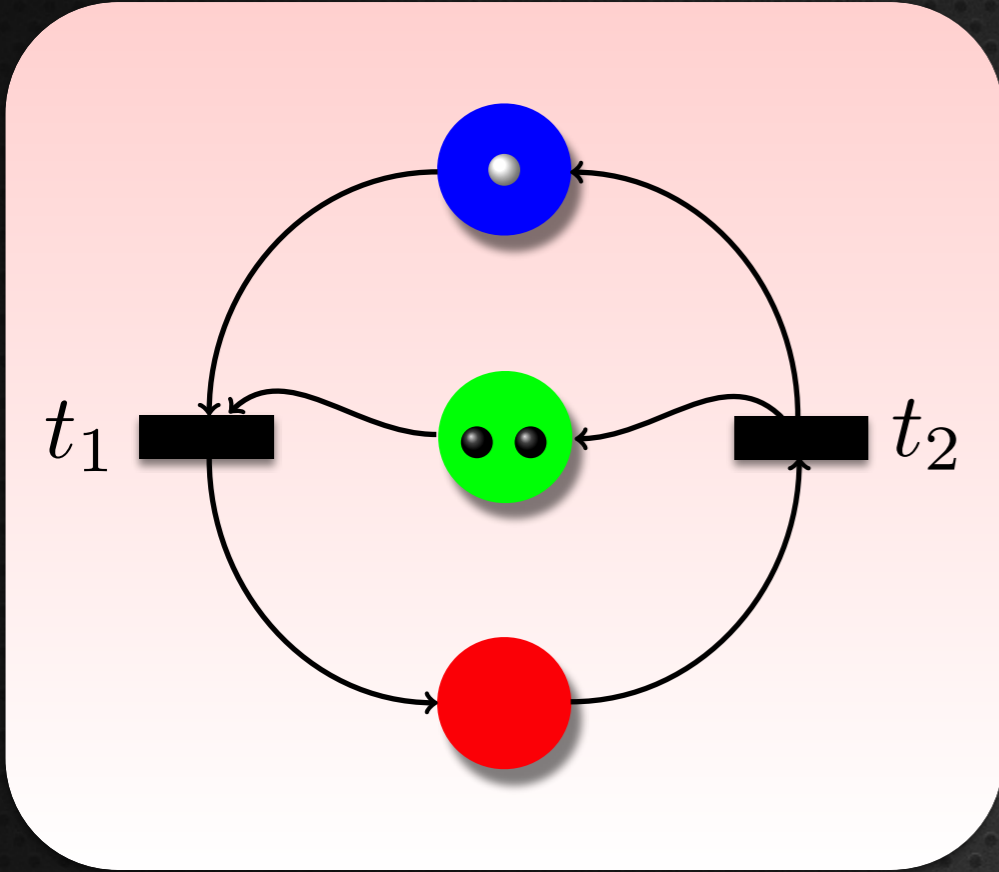
- Encoding (counter abstraction)

- # tokens in  = # processes in 
- # tokens in  = # processes in 
- one/no token in  = lock free/busy

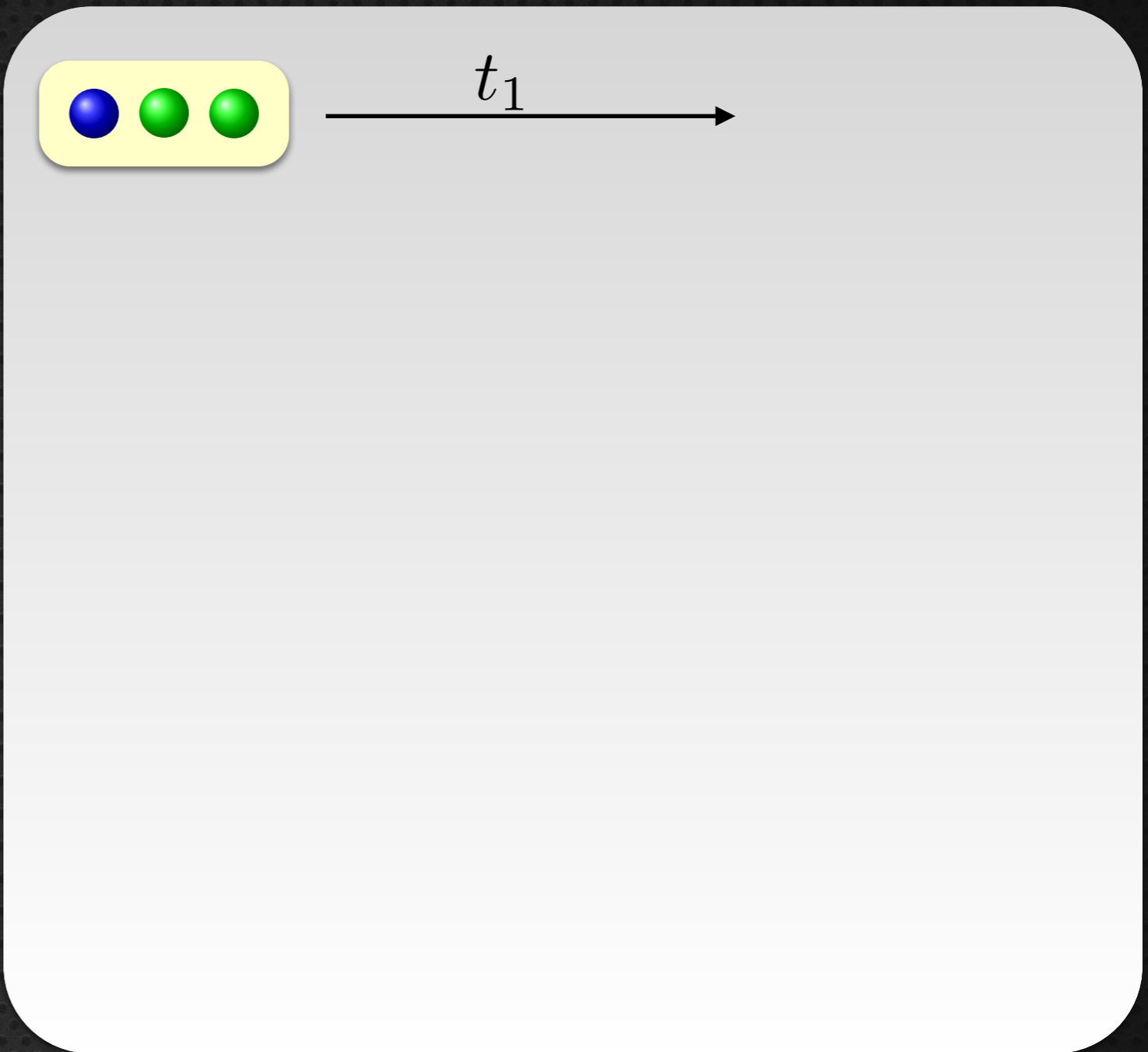
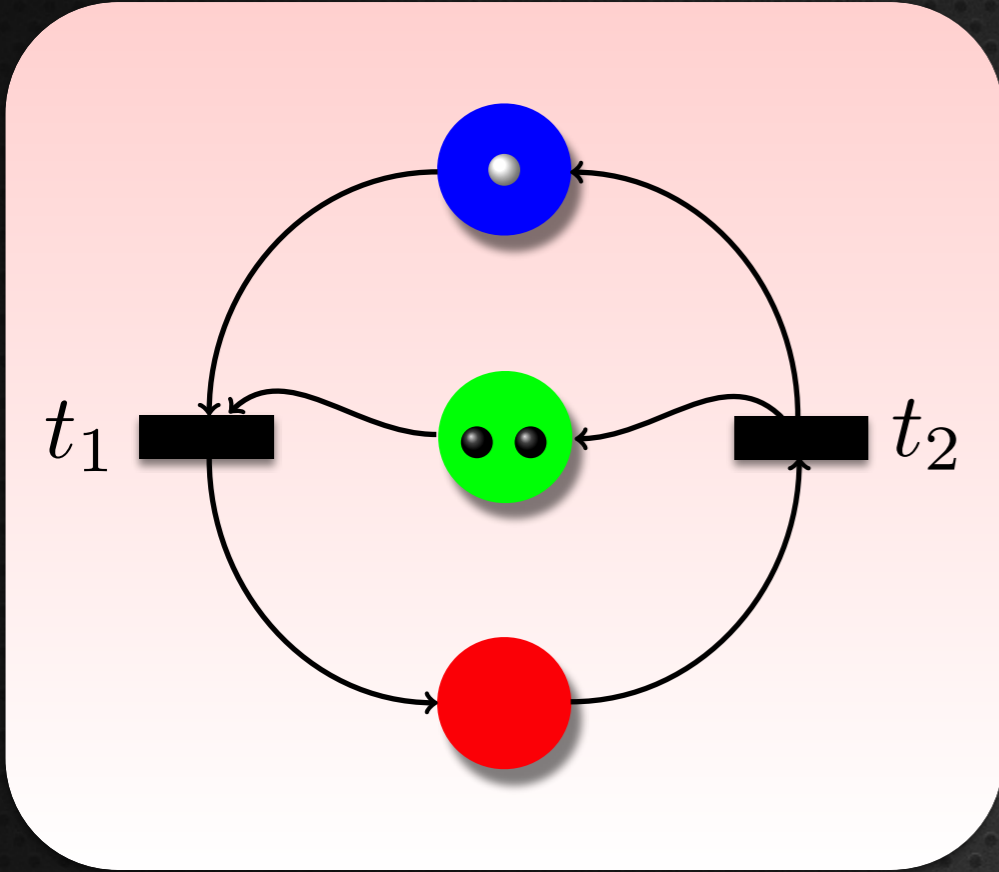
Transitions



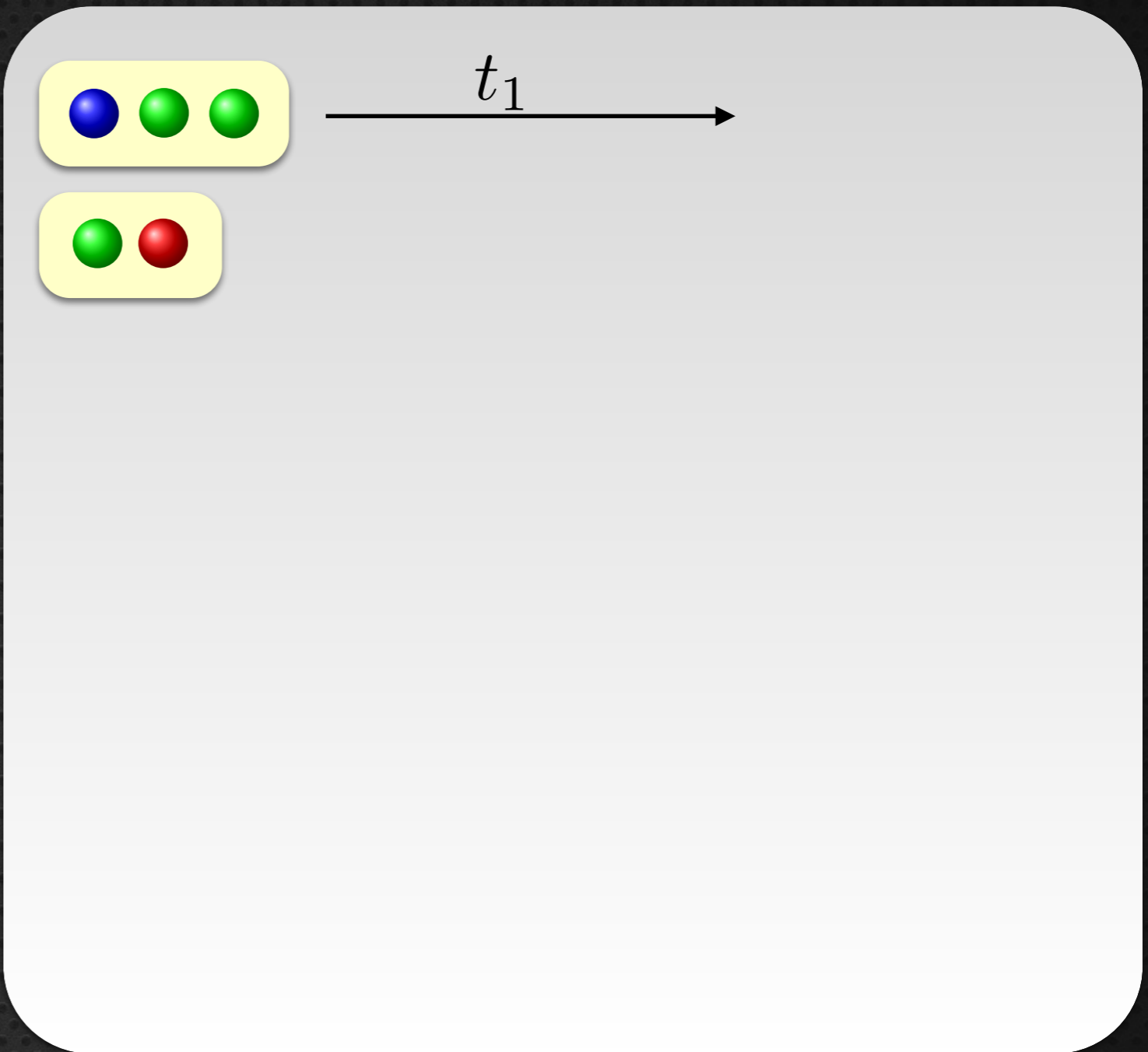
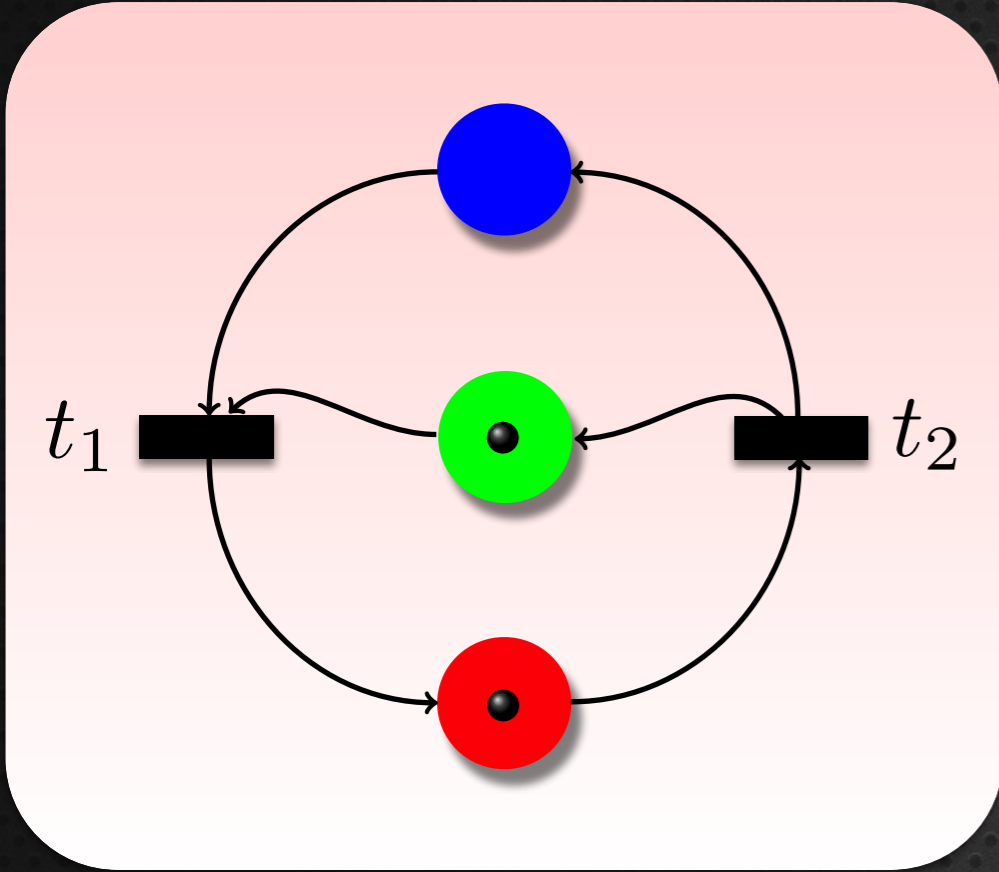
Transitions



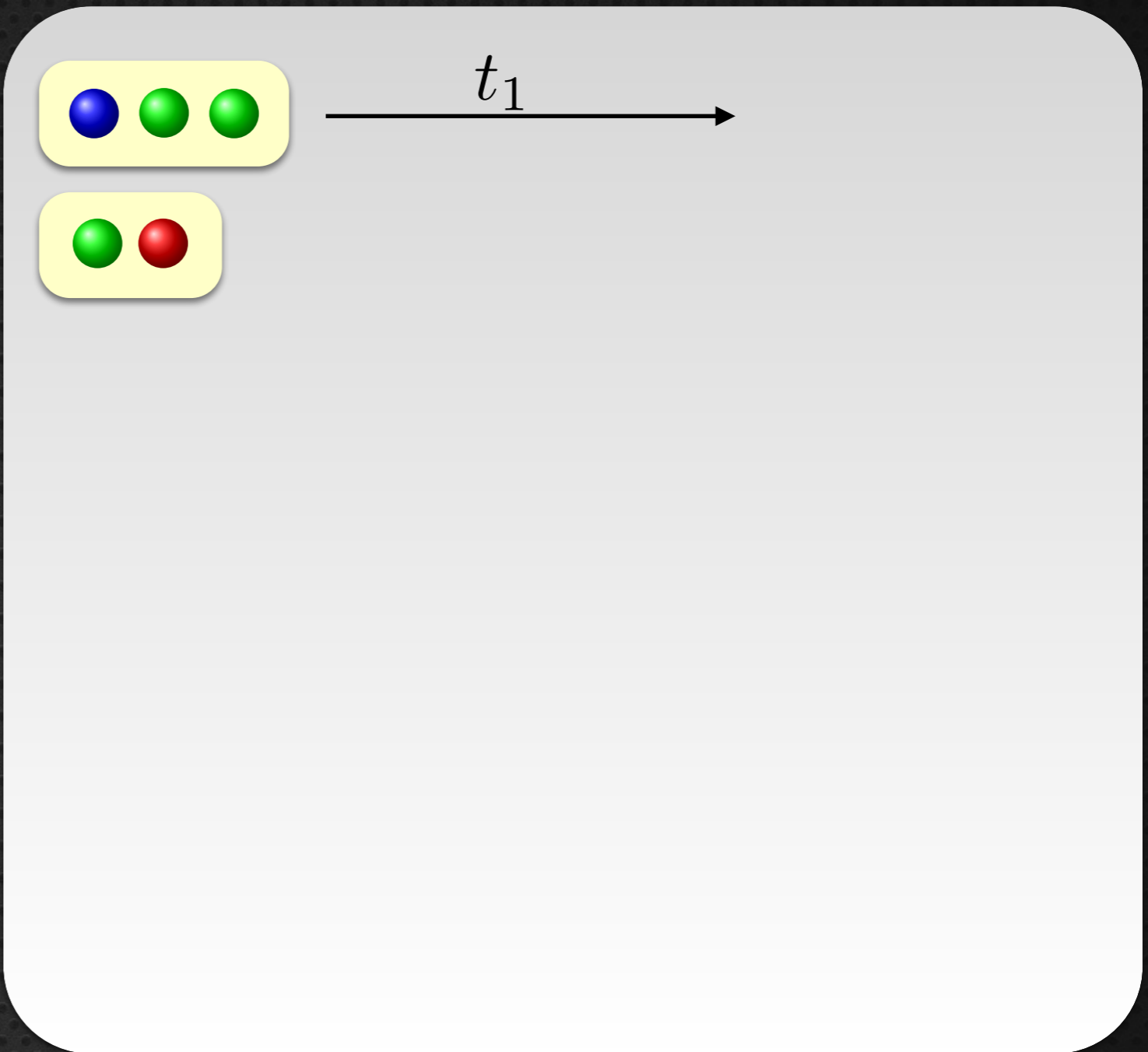
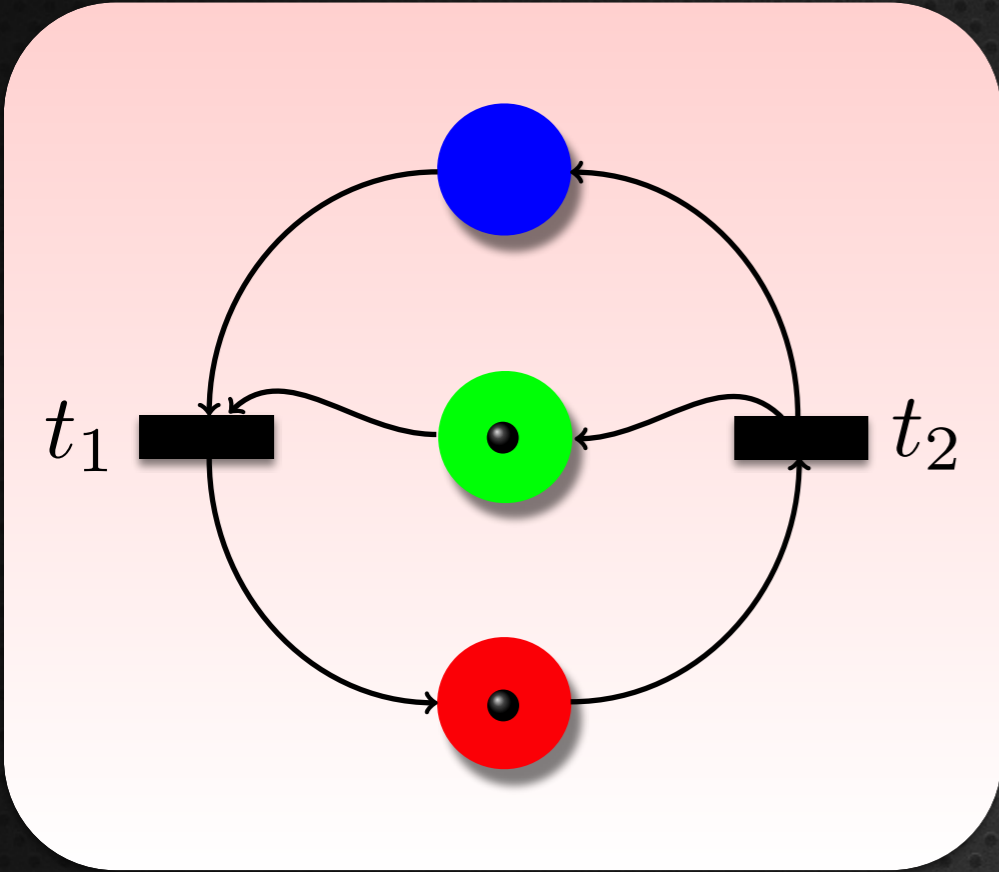
Transitions



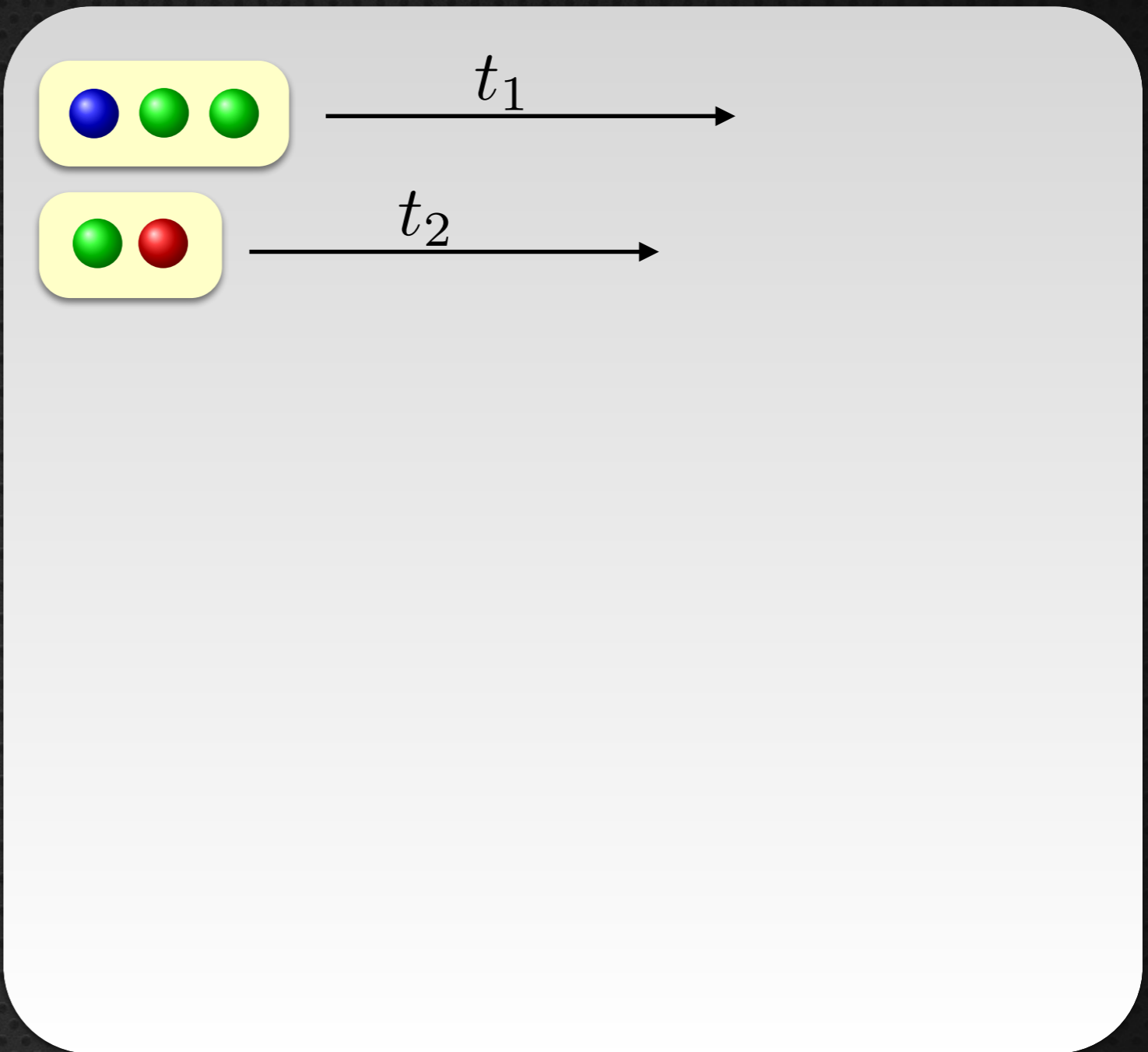
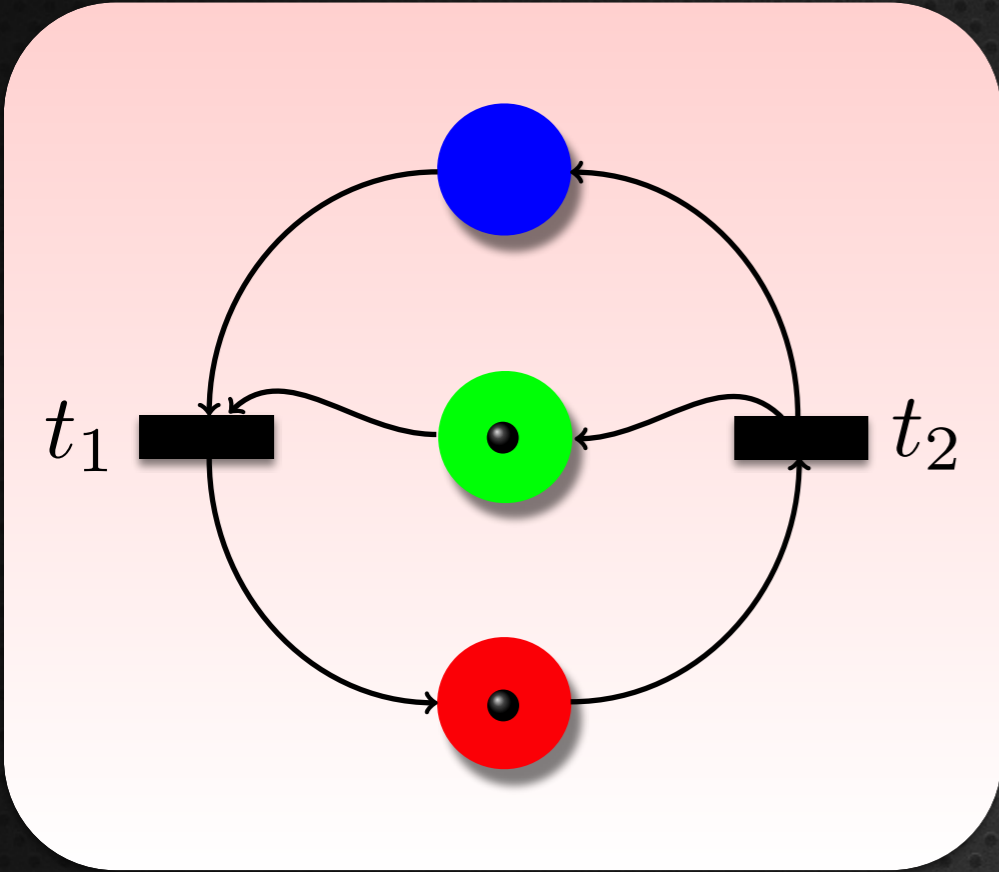
Transitions



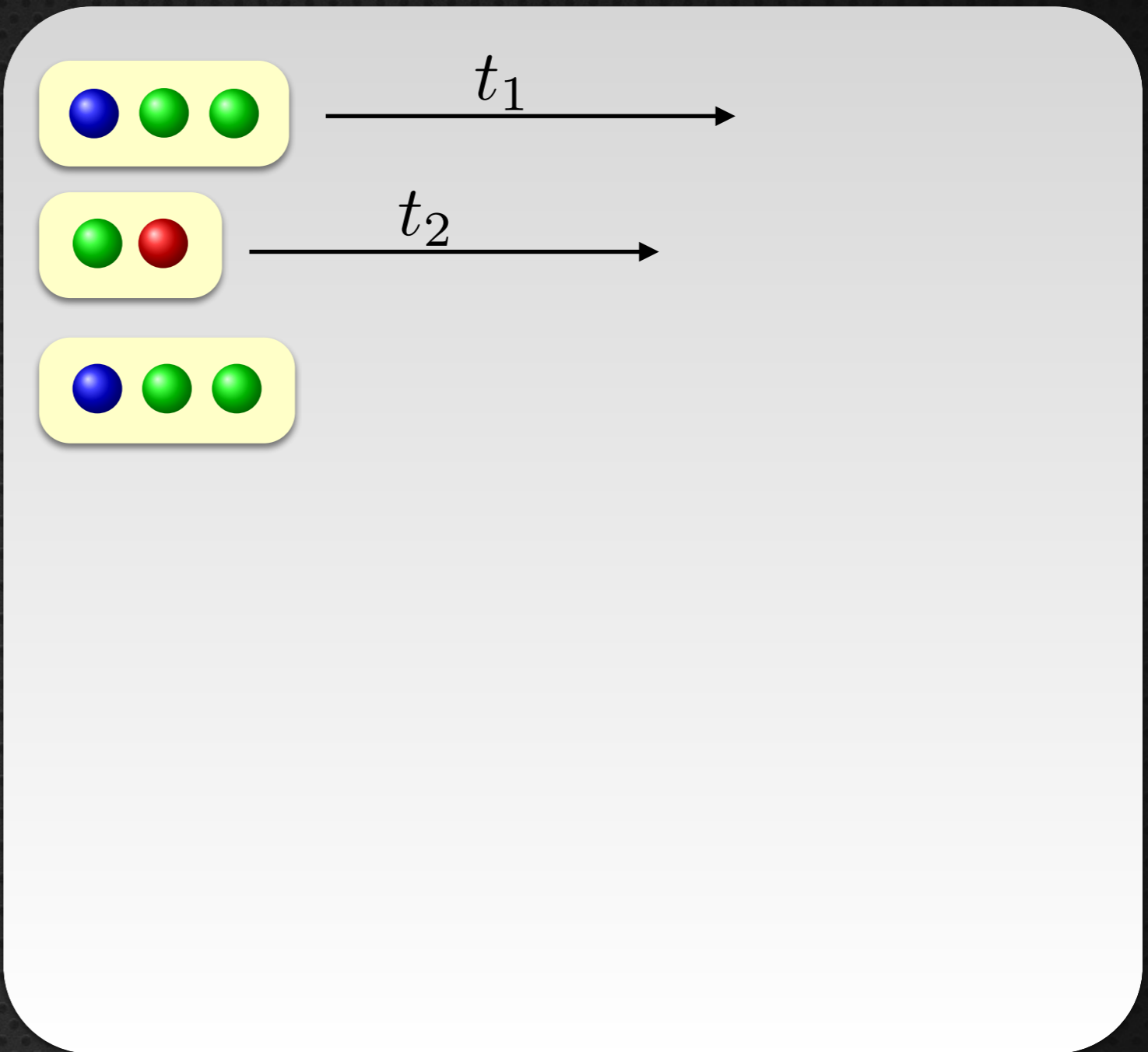
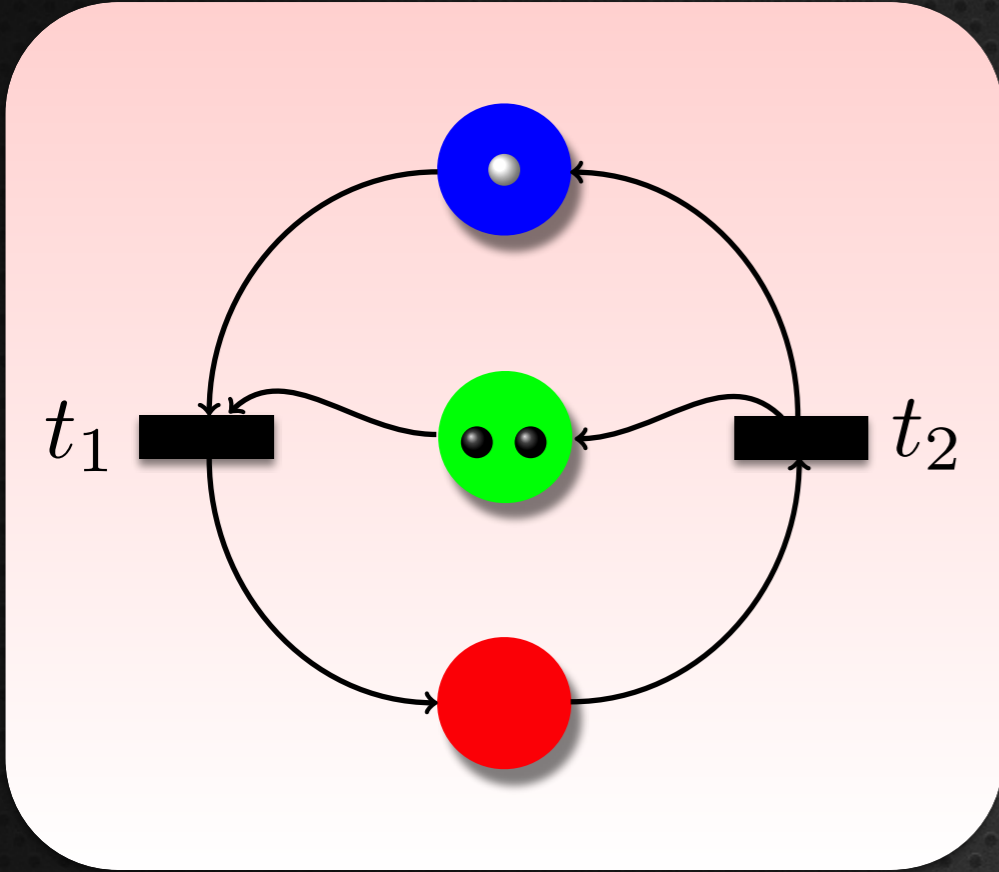
Transitions



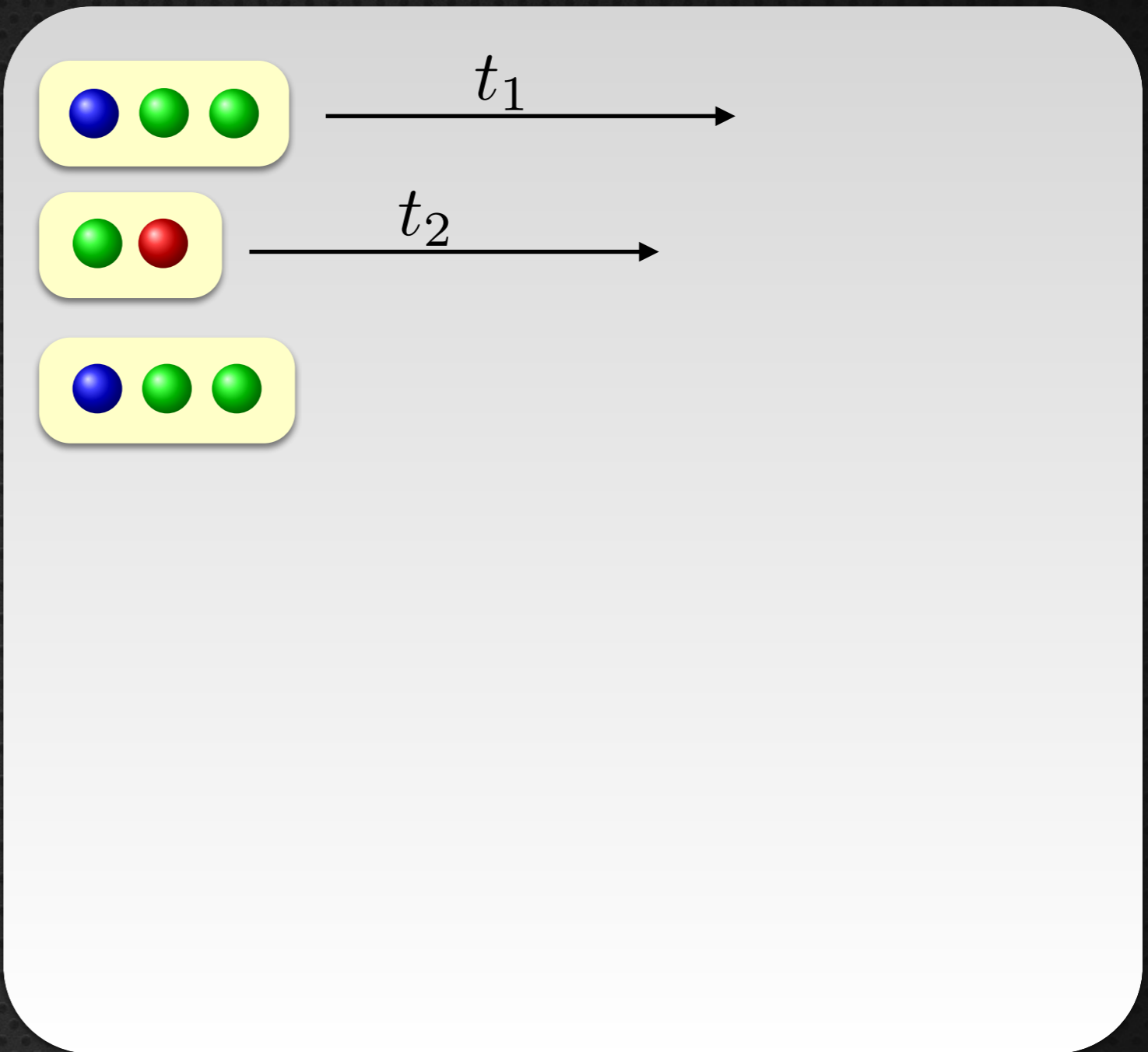
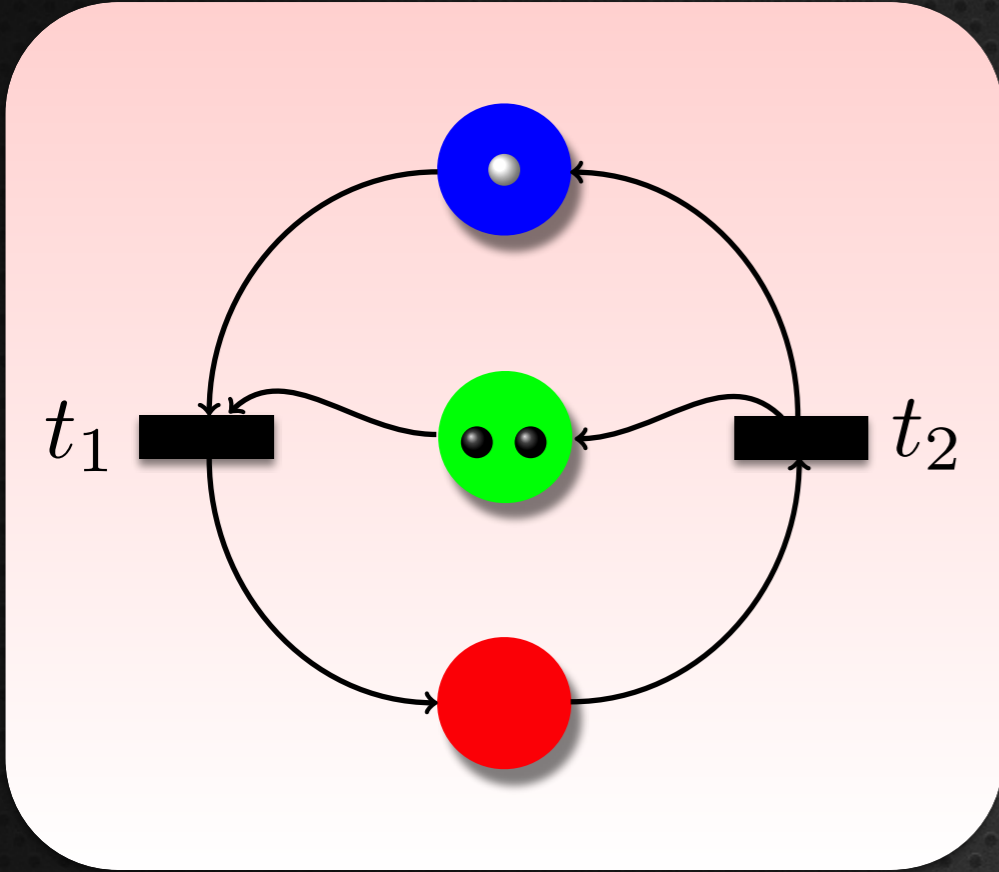
Transitions



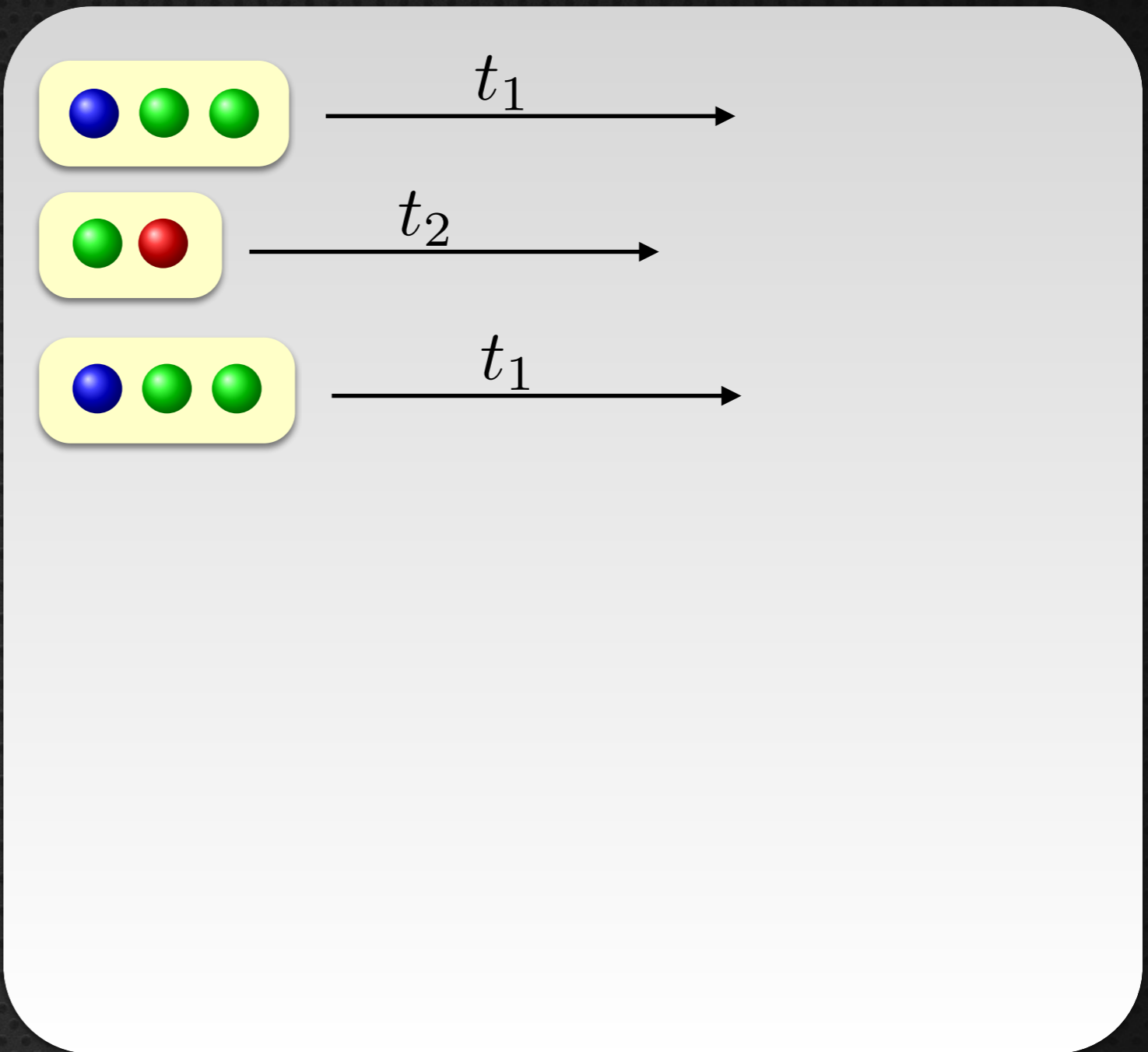
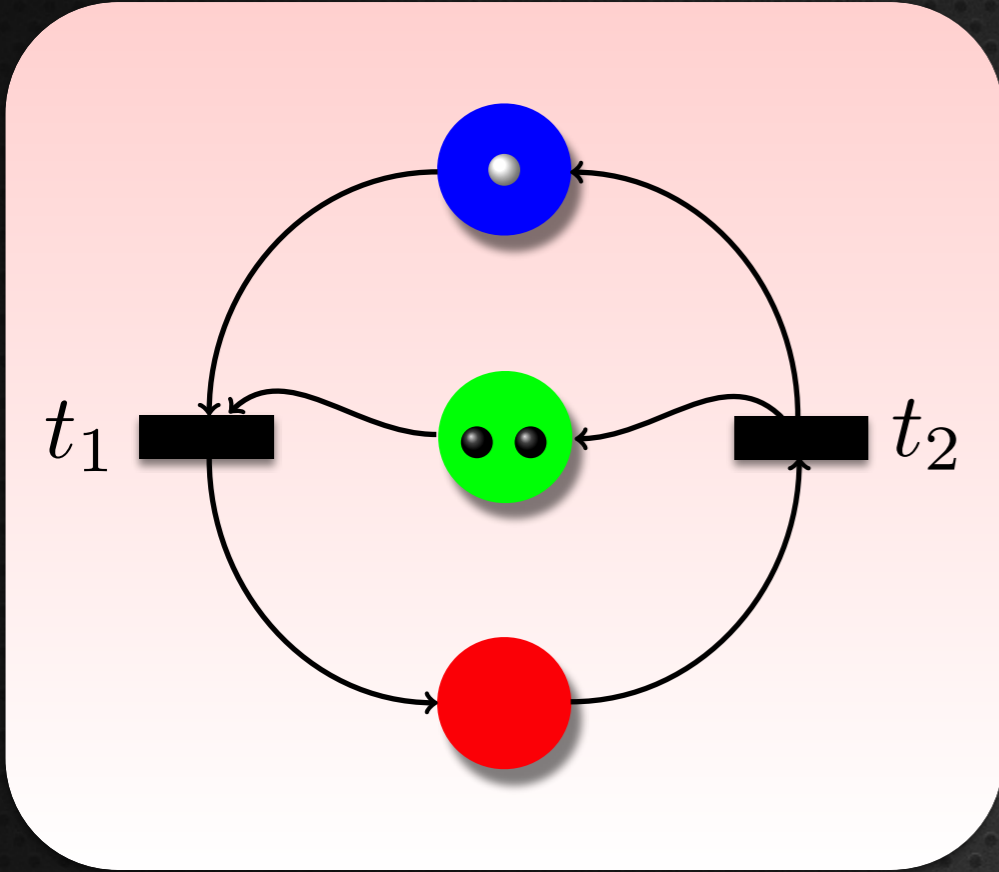
Transitions



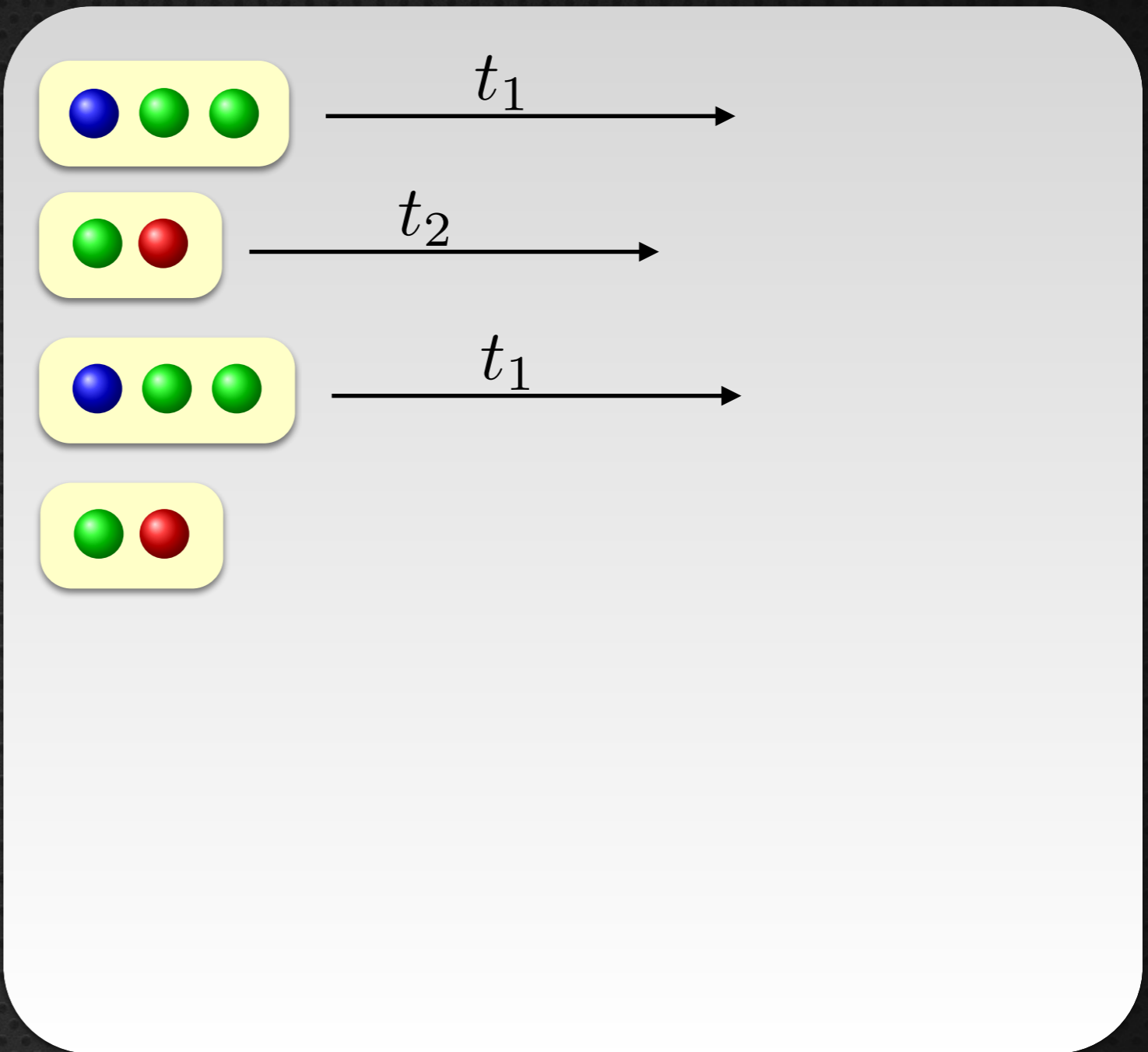
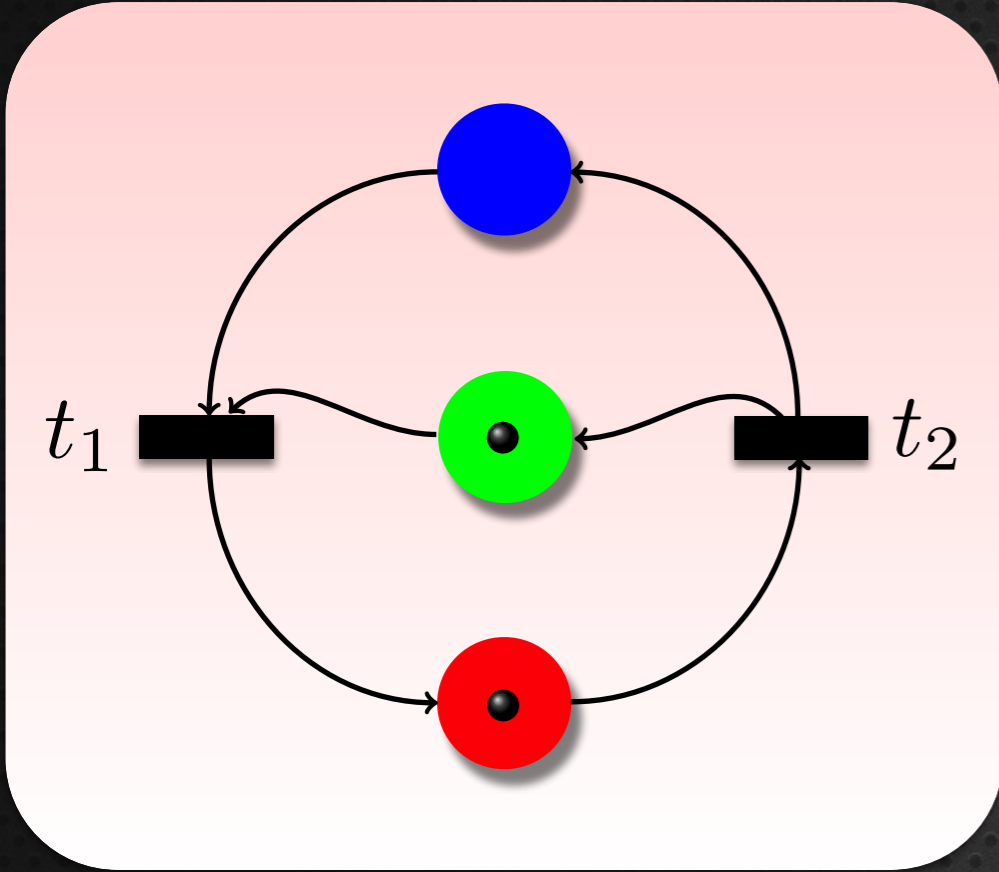
Transitions



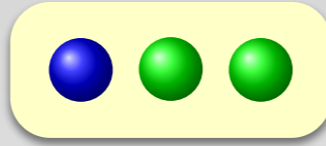
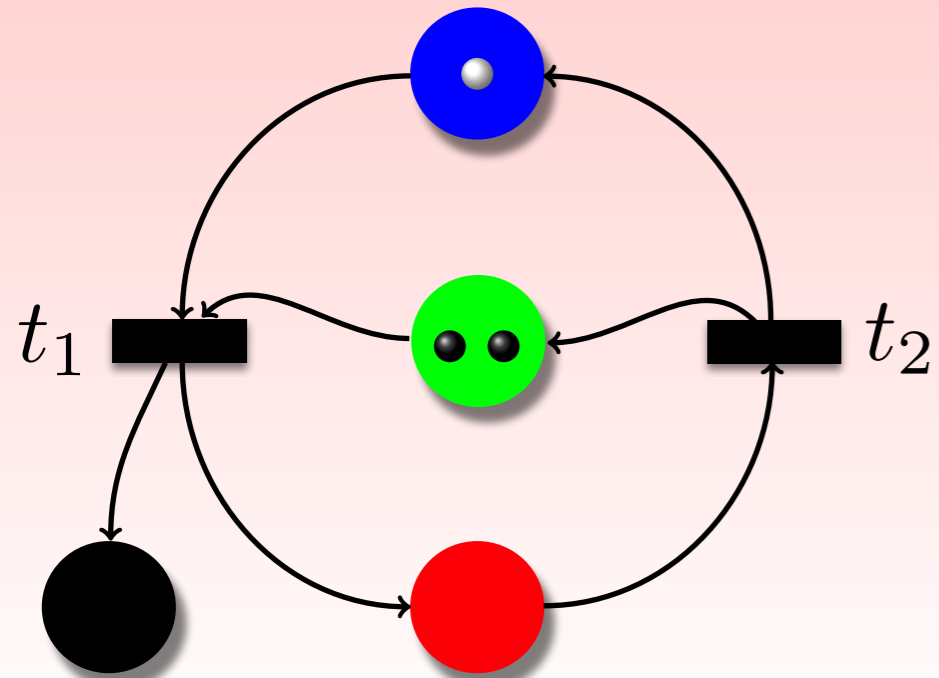
Transitions



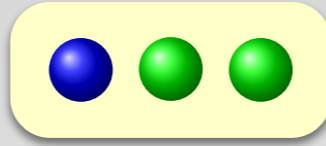
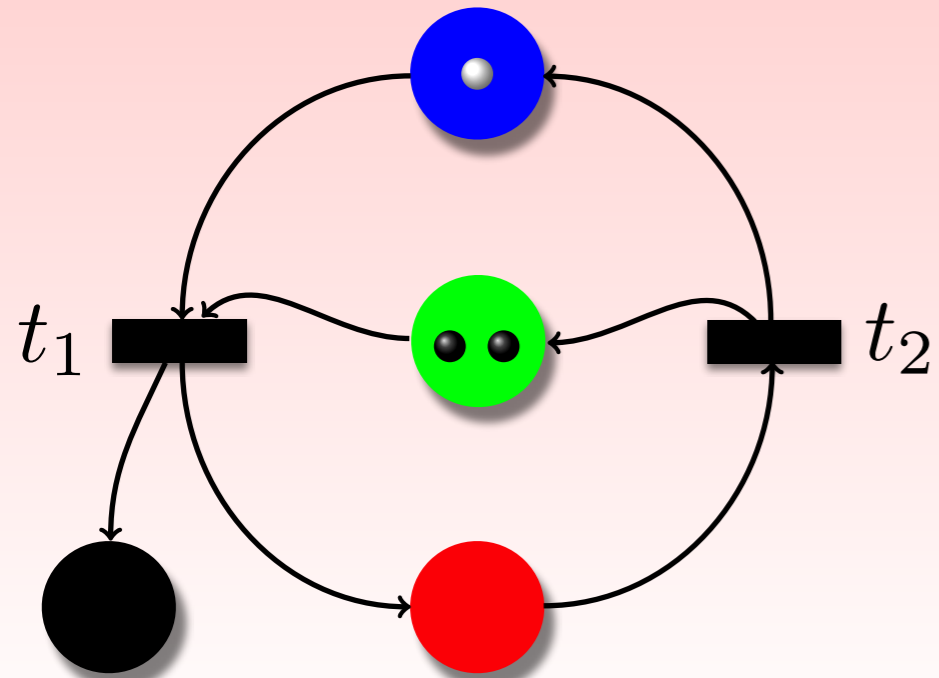
Transitions



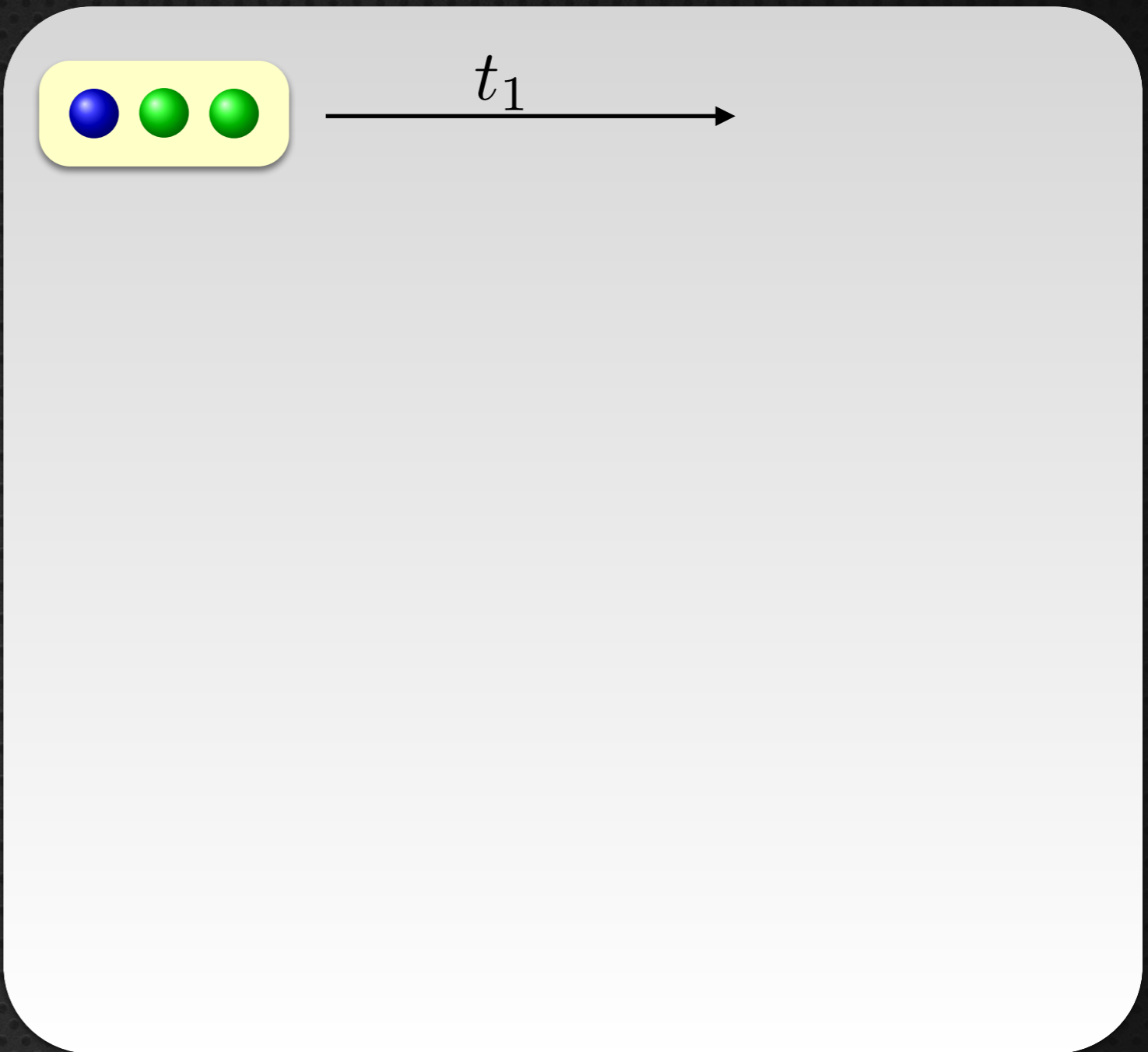
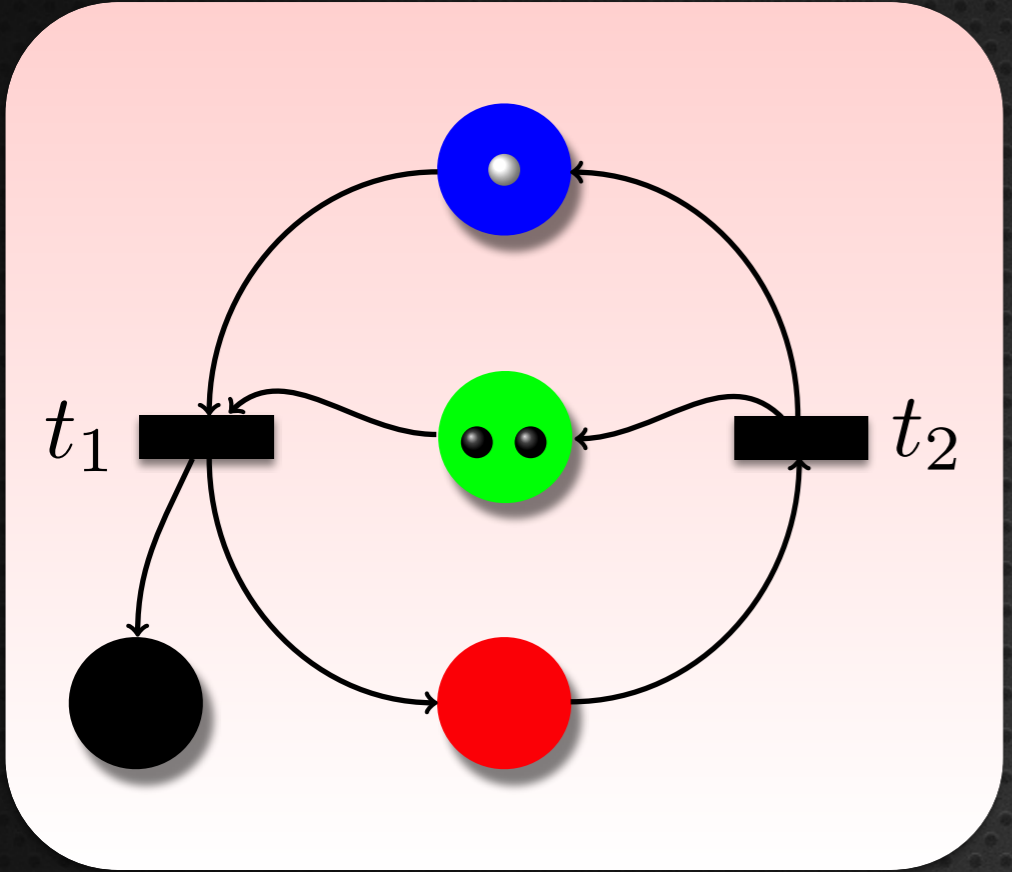
Transitions



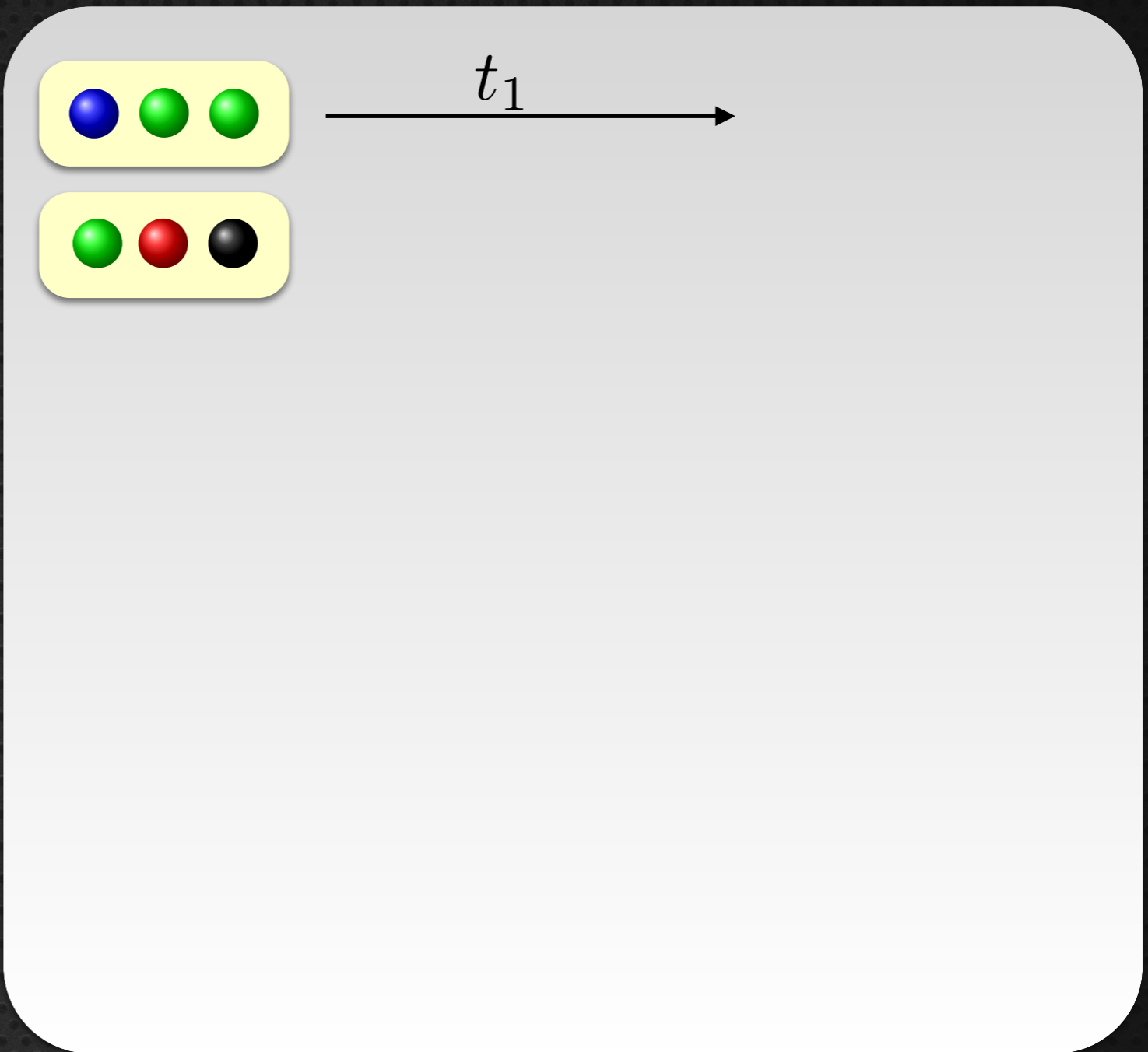
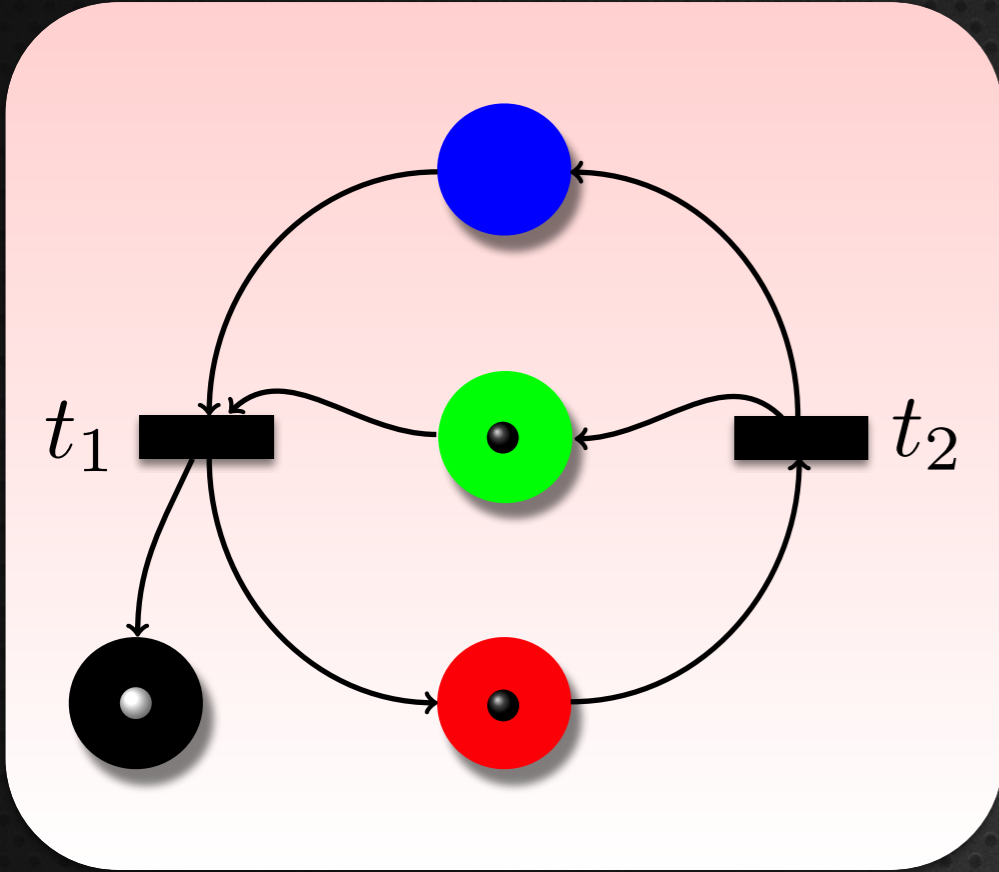
Transitions



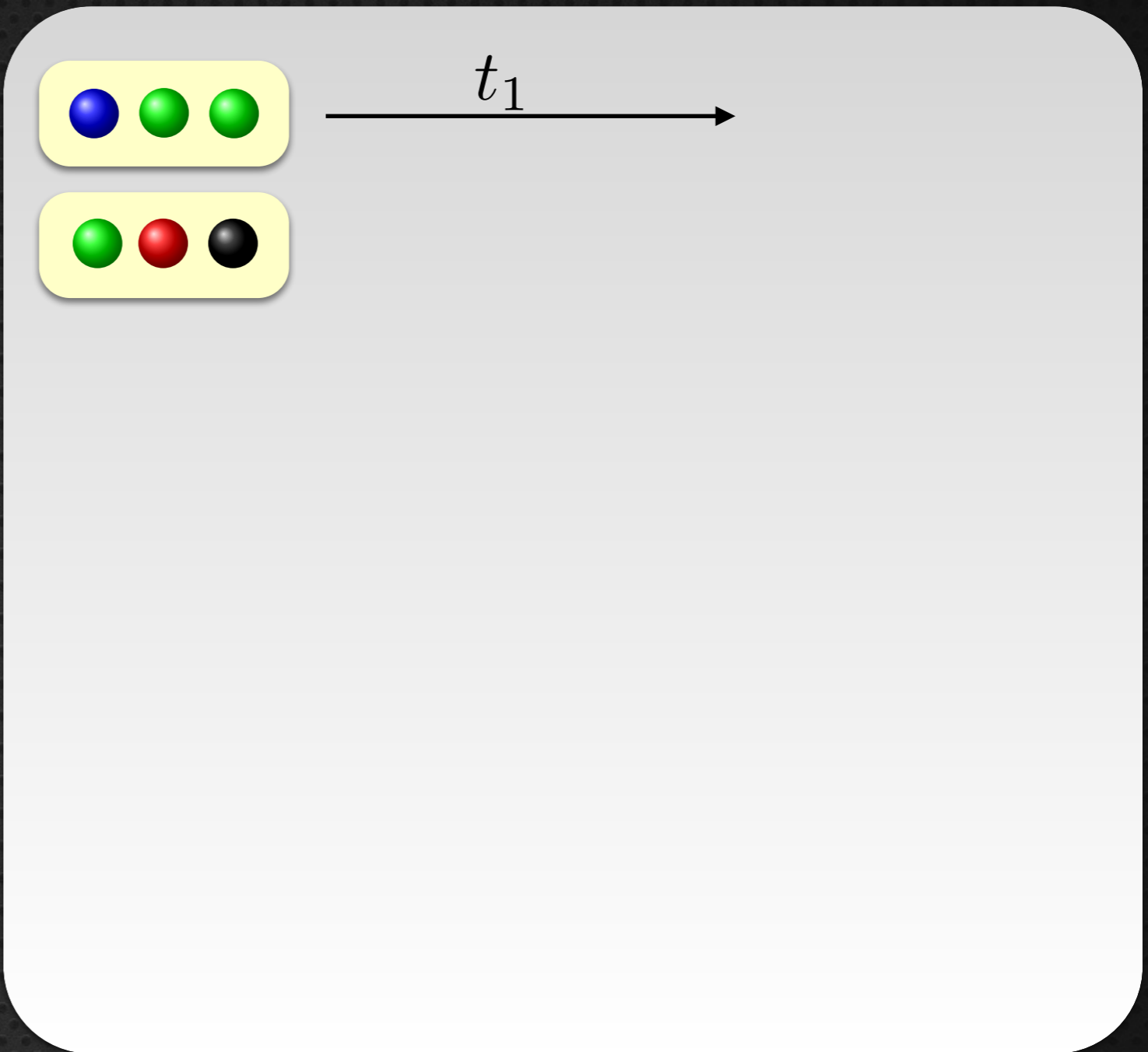
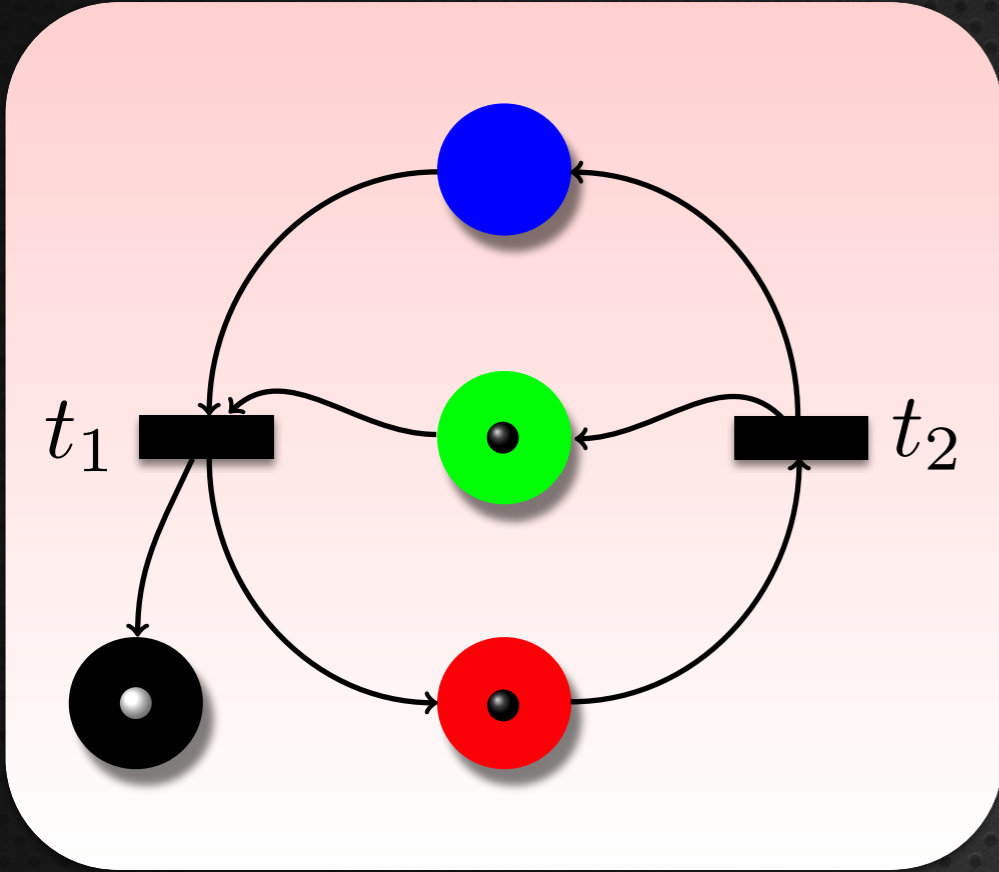
Transitions



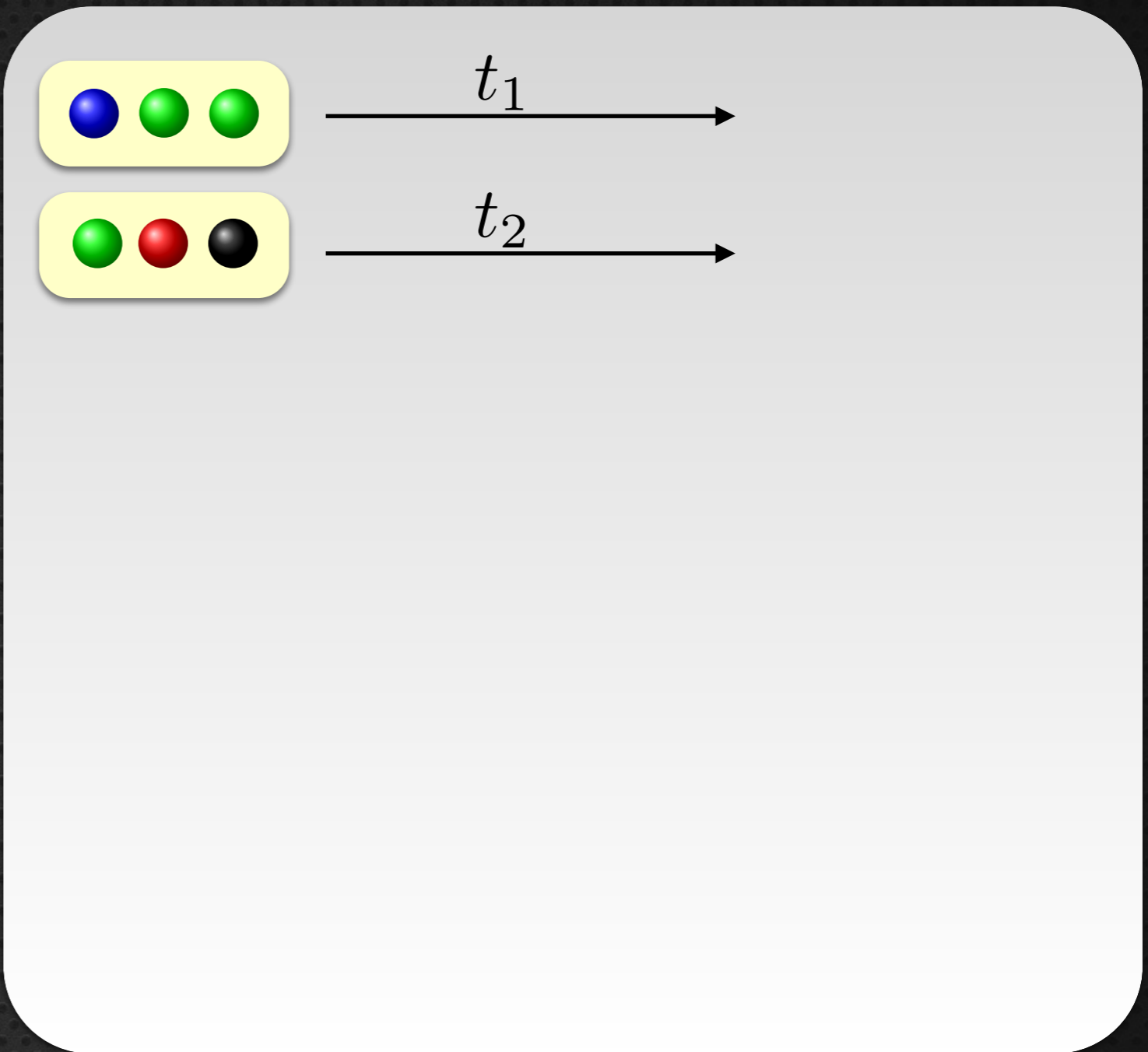
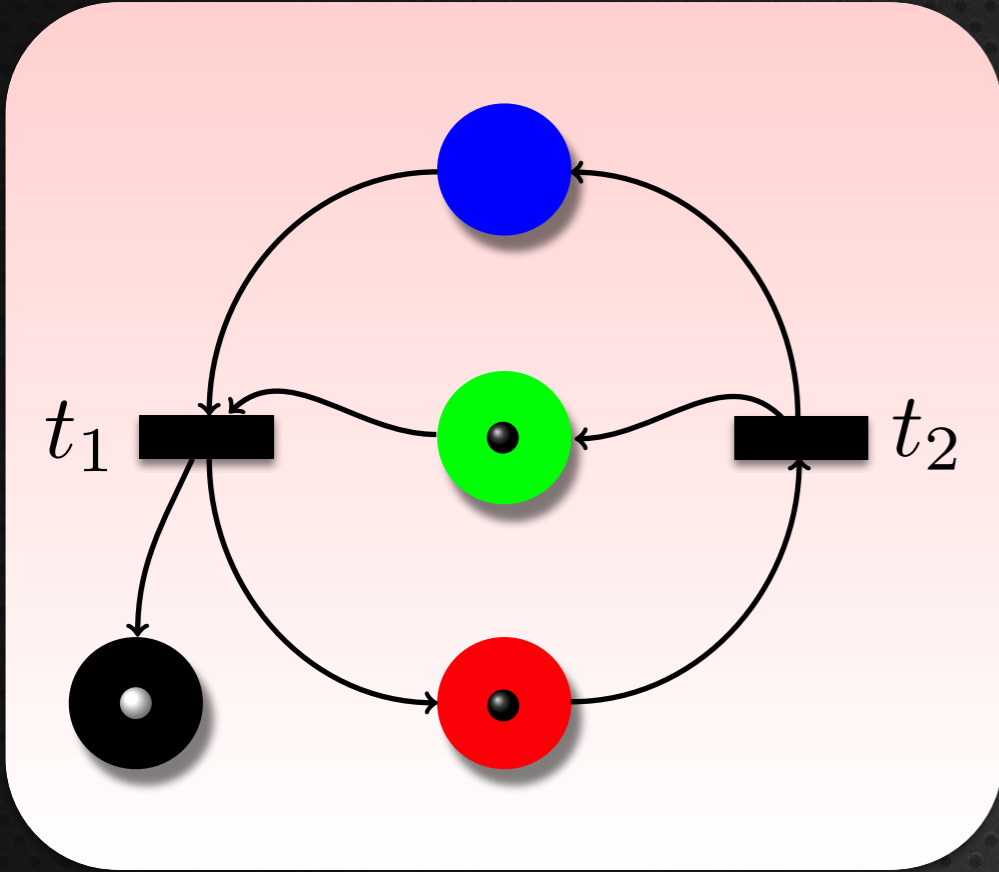
Transitions



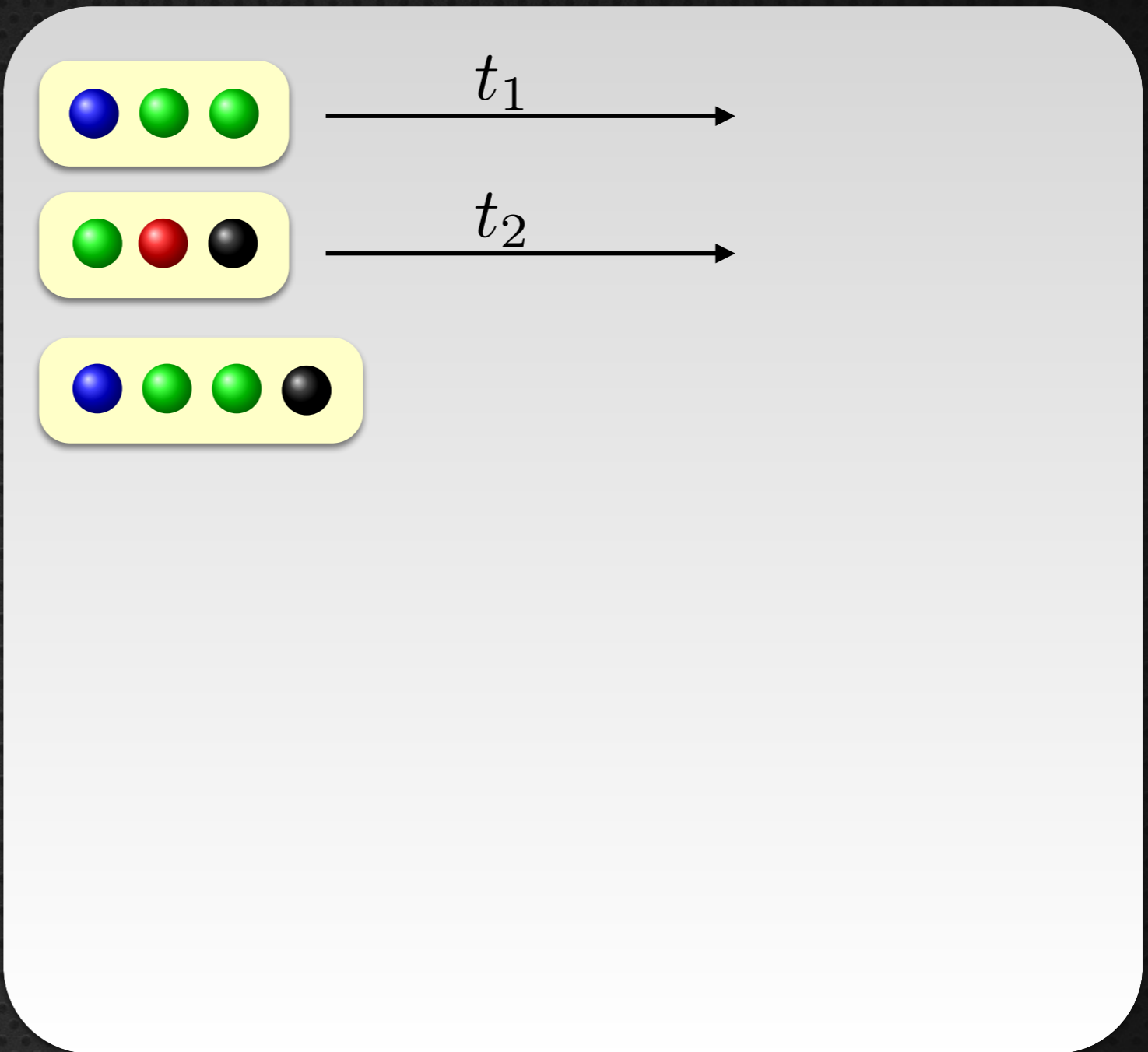
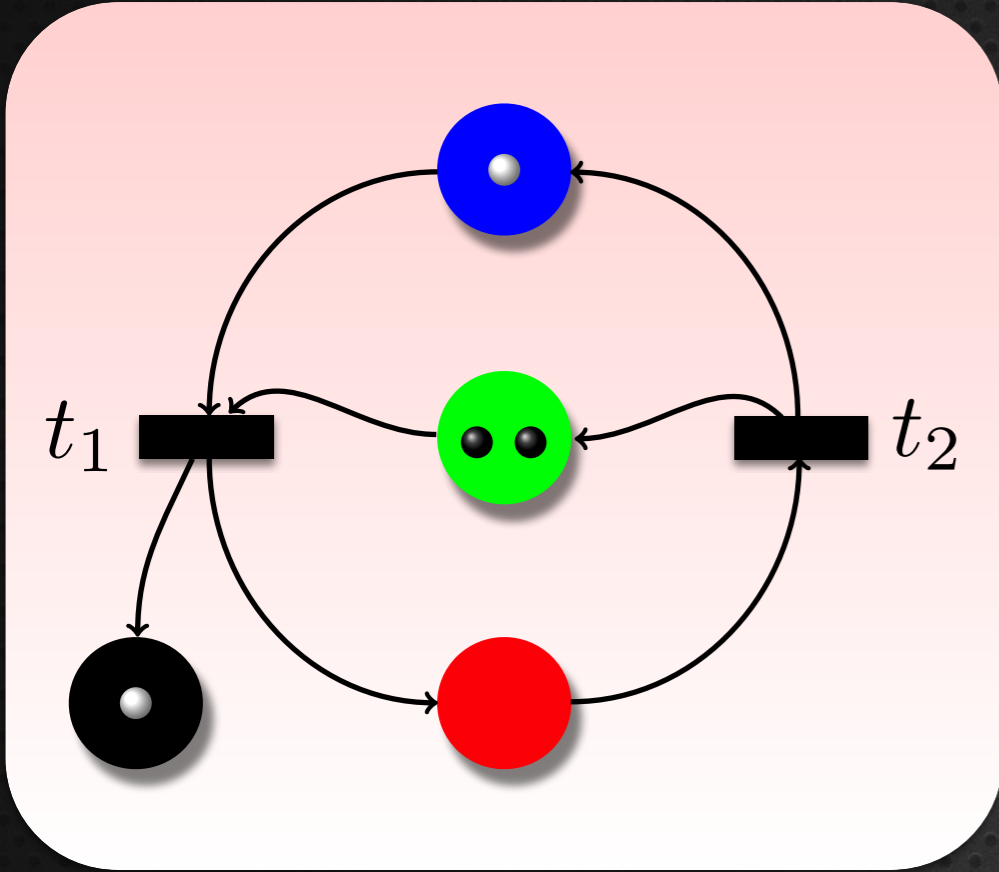
Transitions



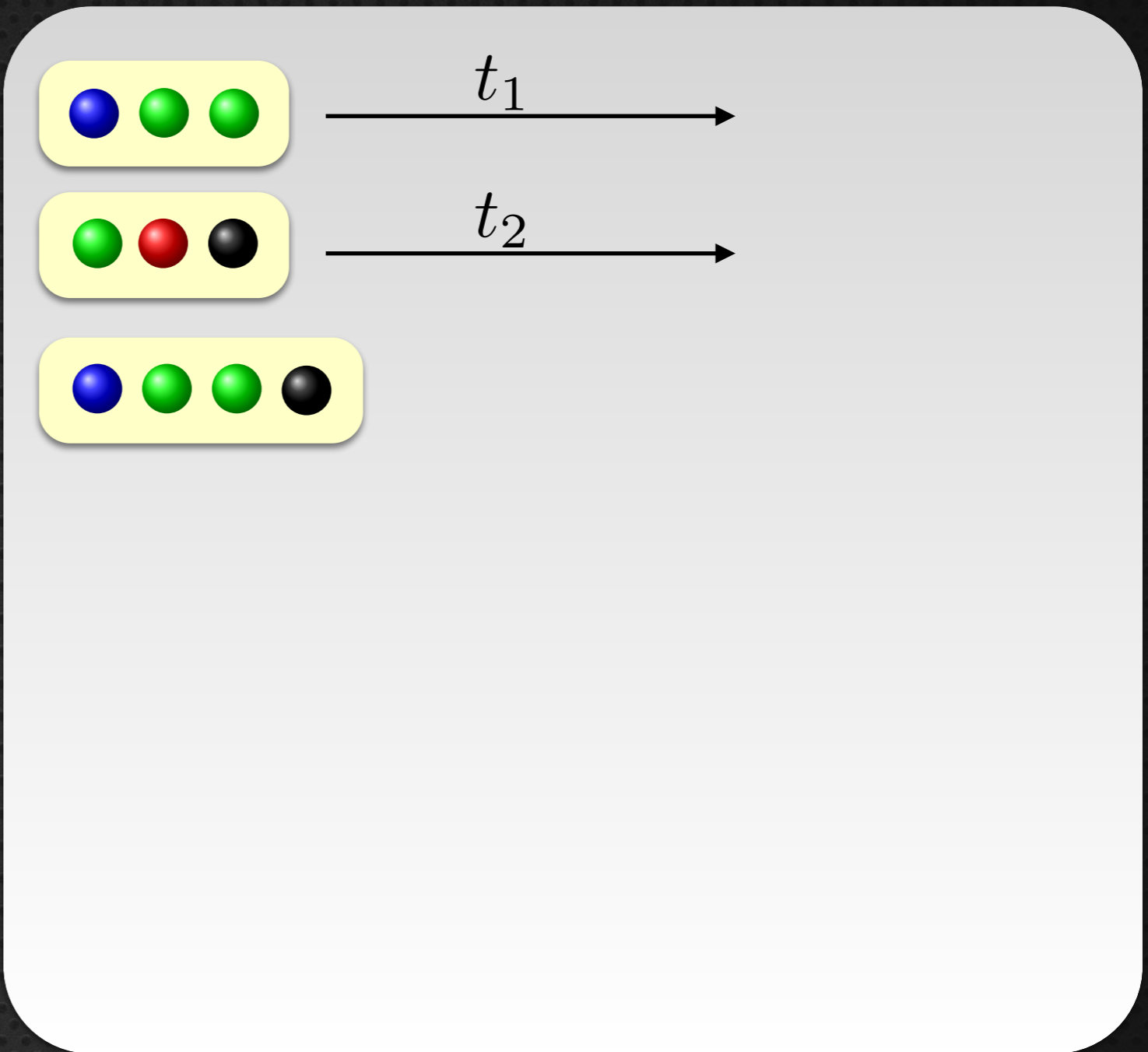
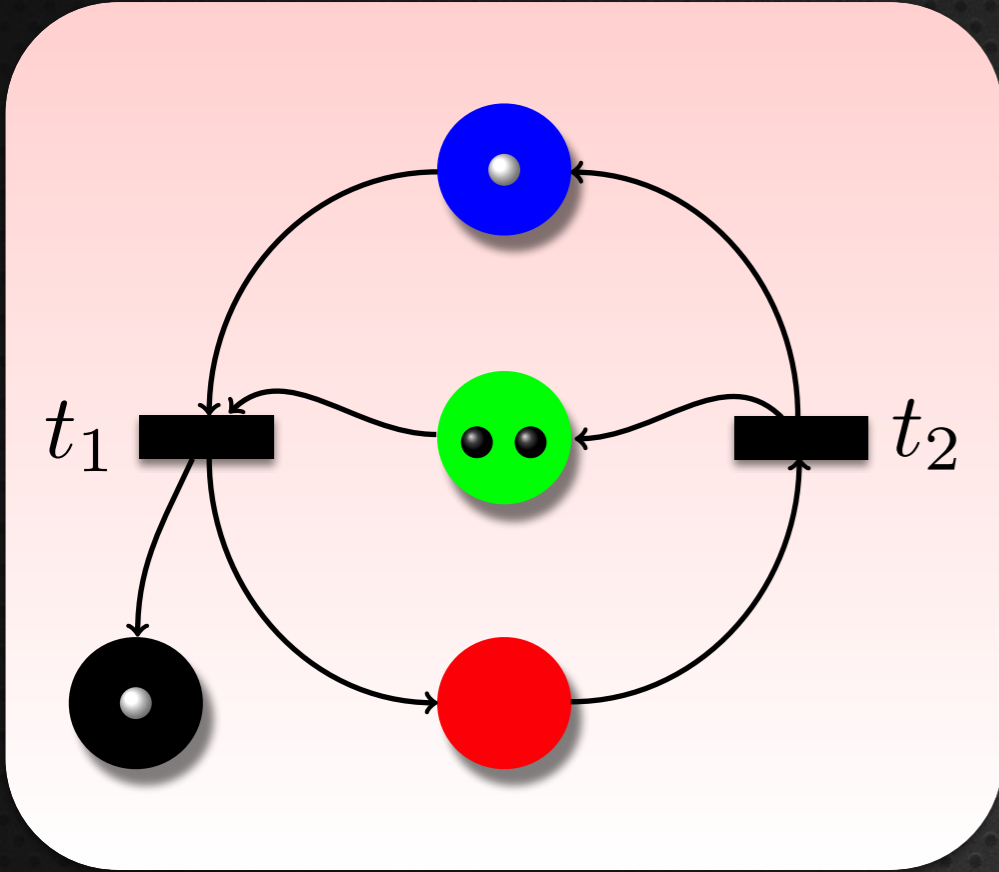
Transitions



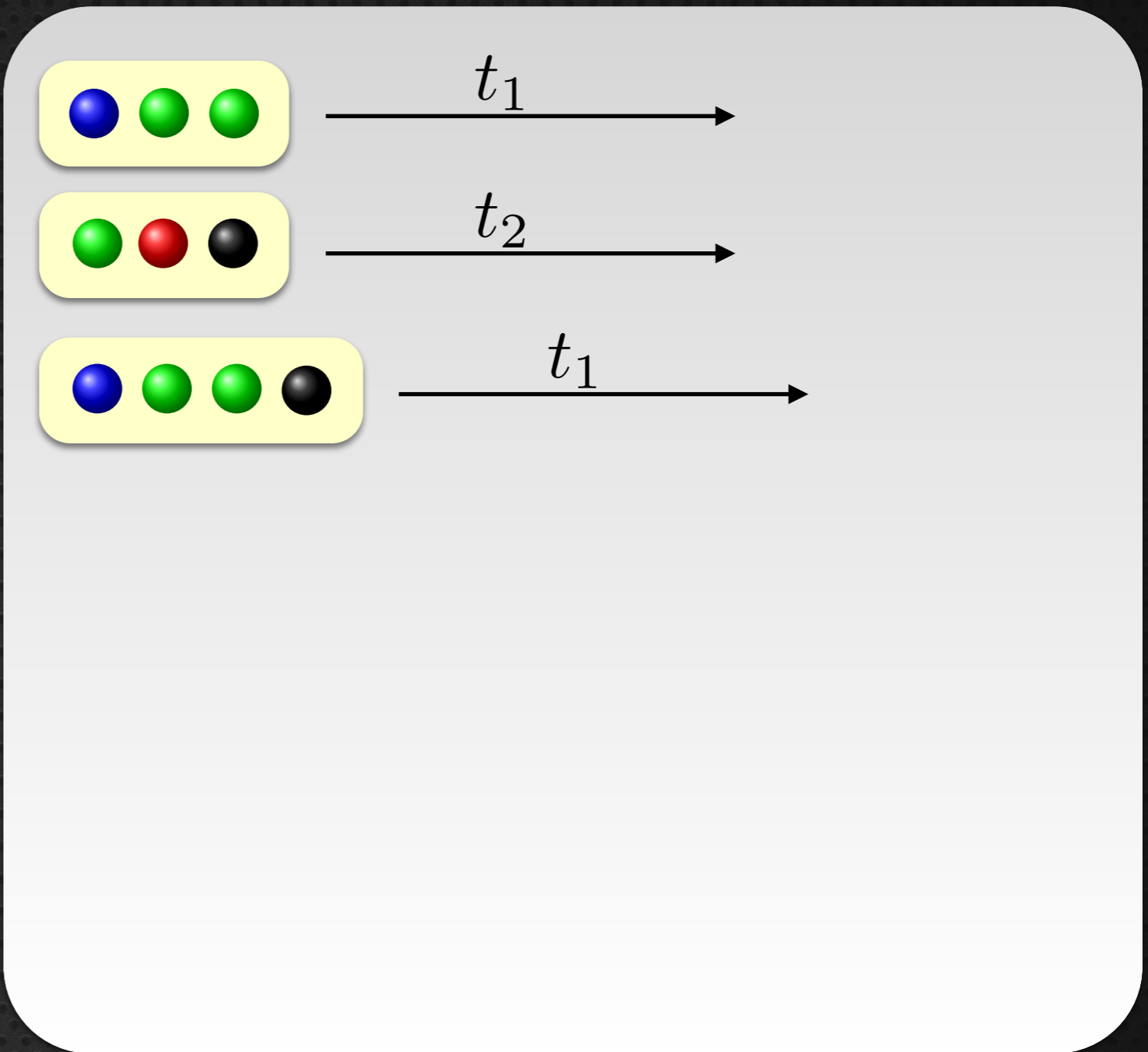
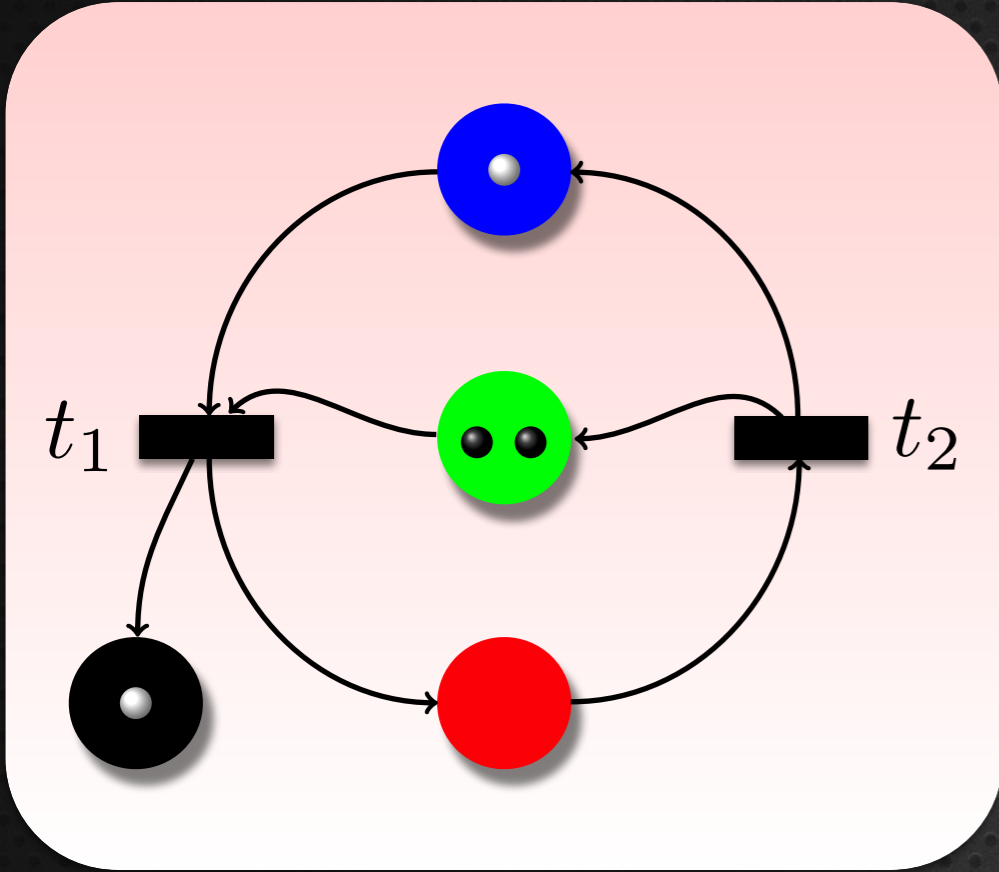
Transitions



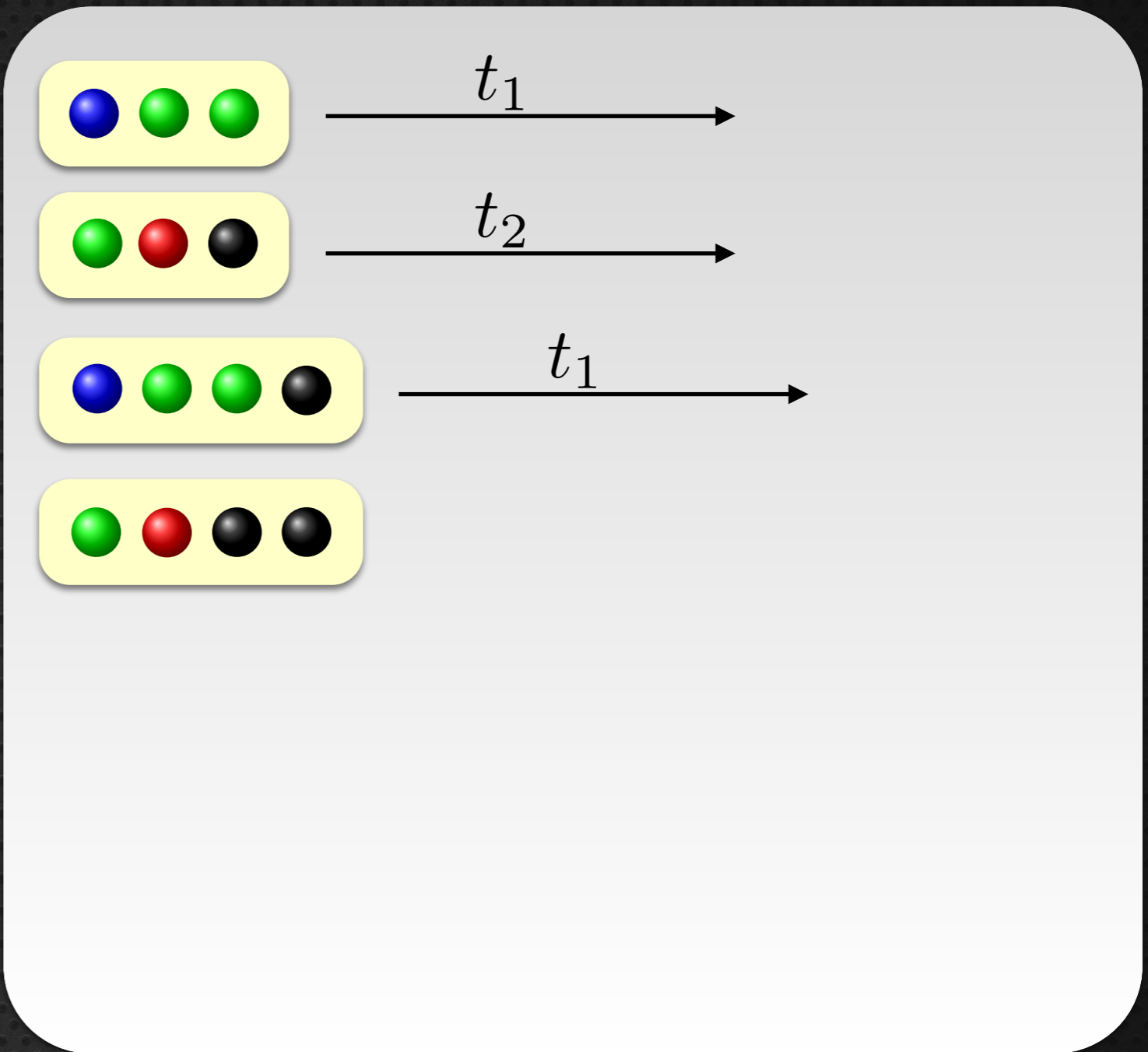
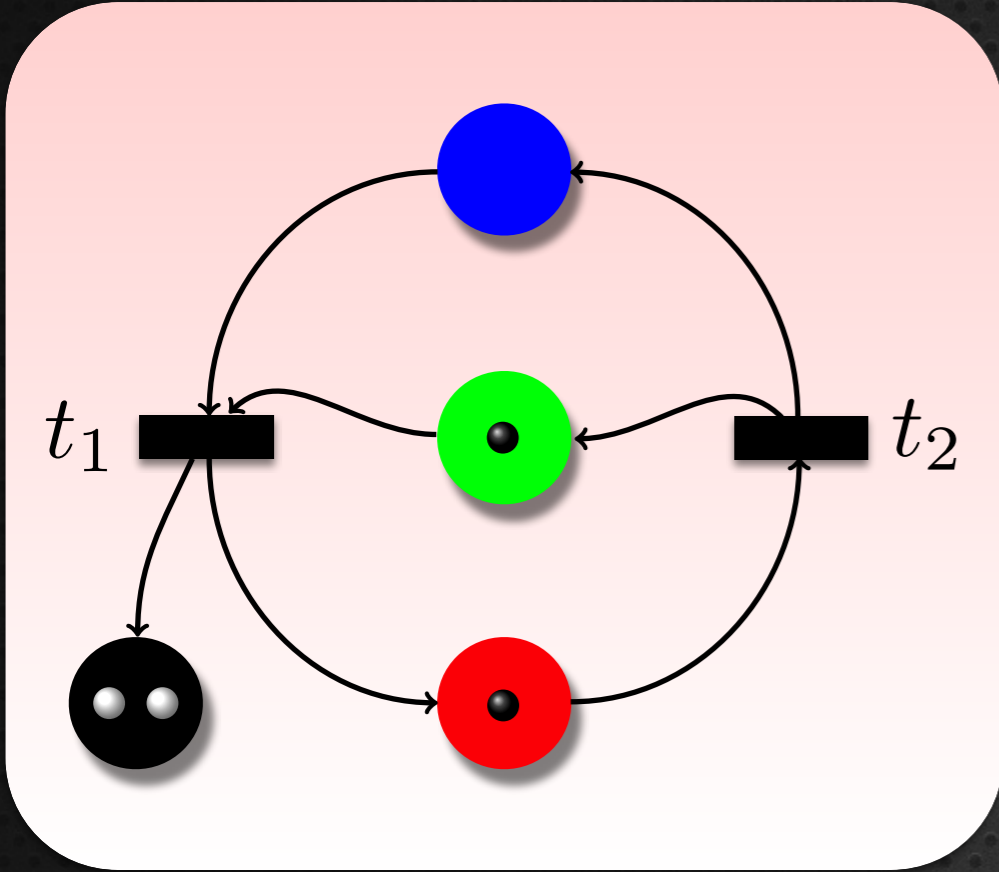
Transitions



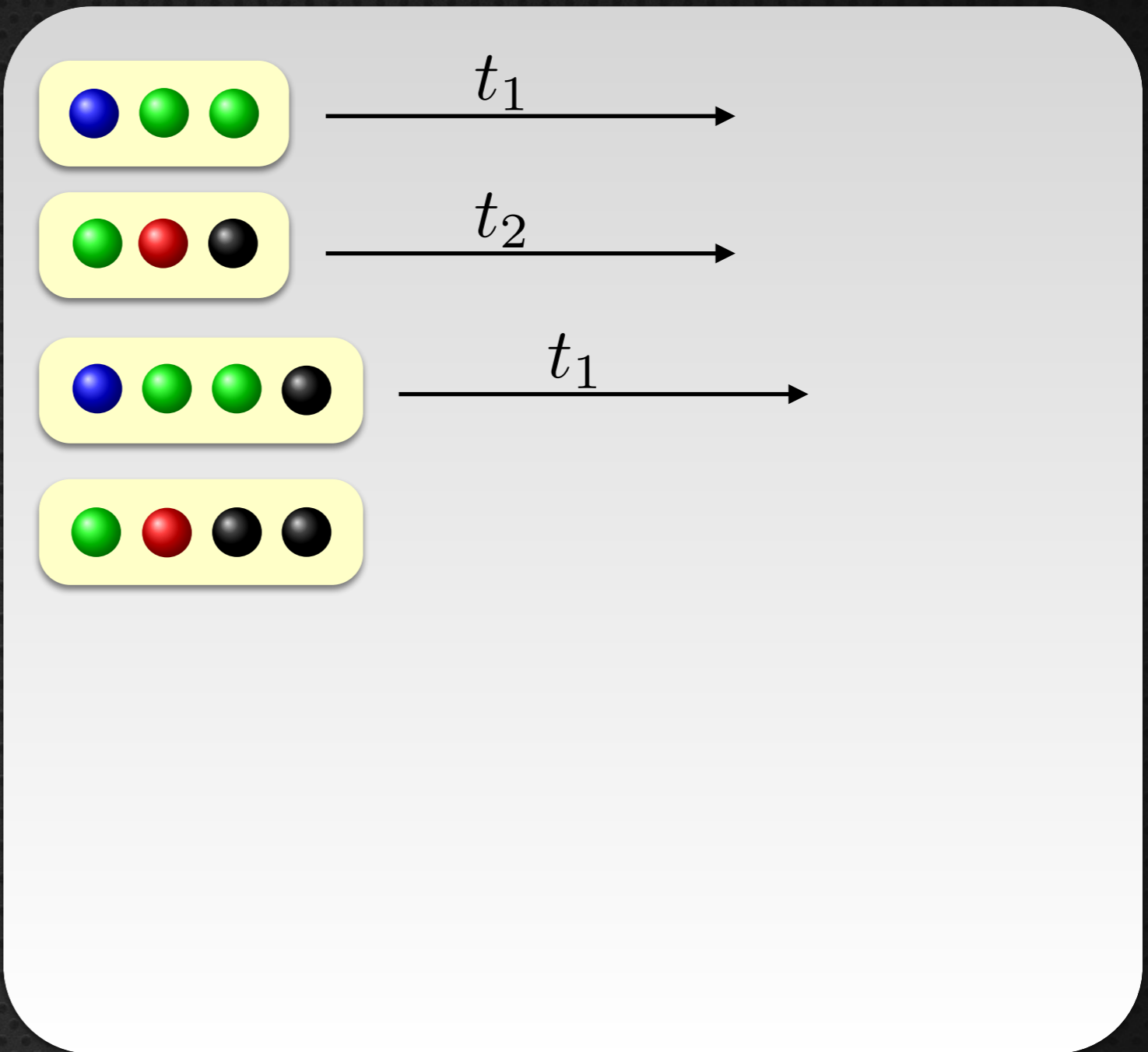
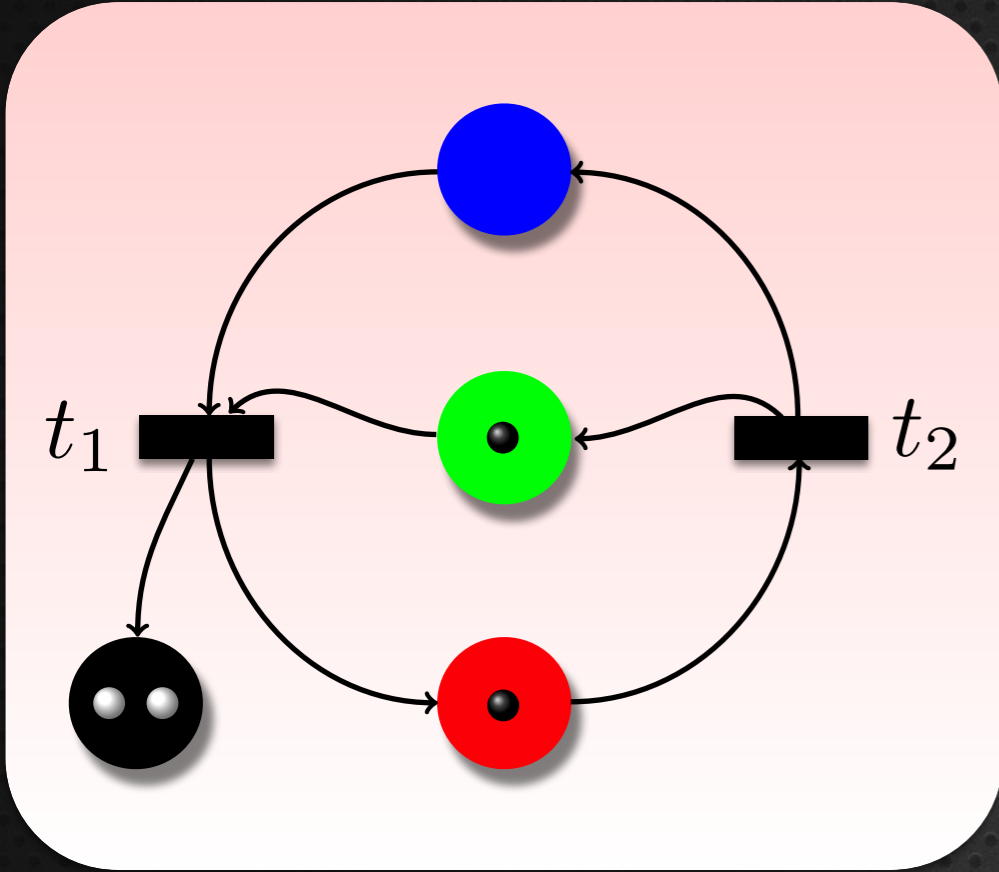
Transitions



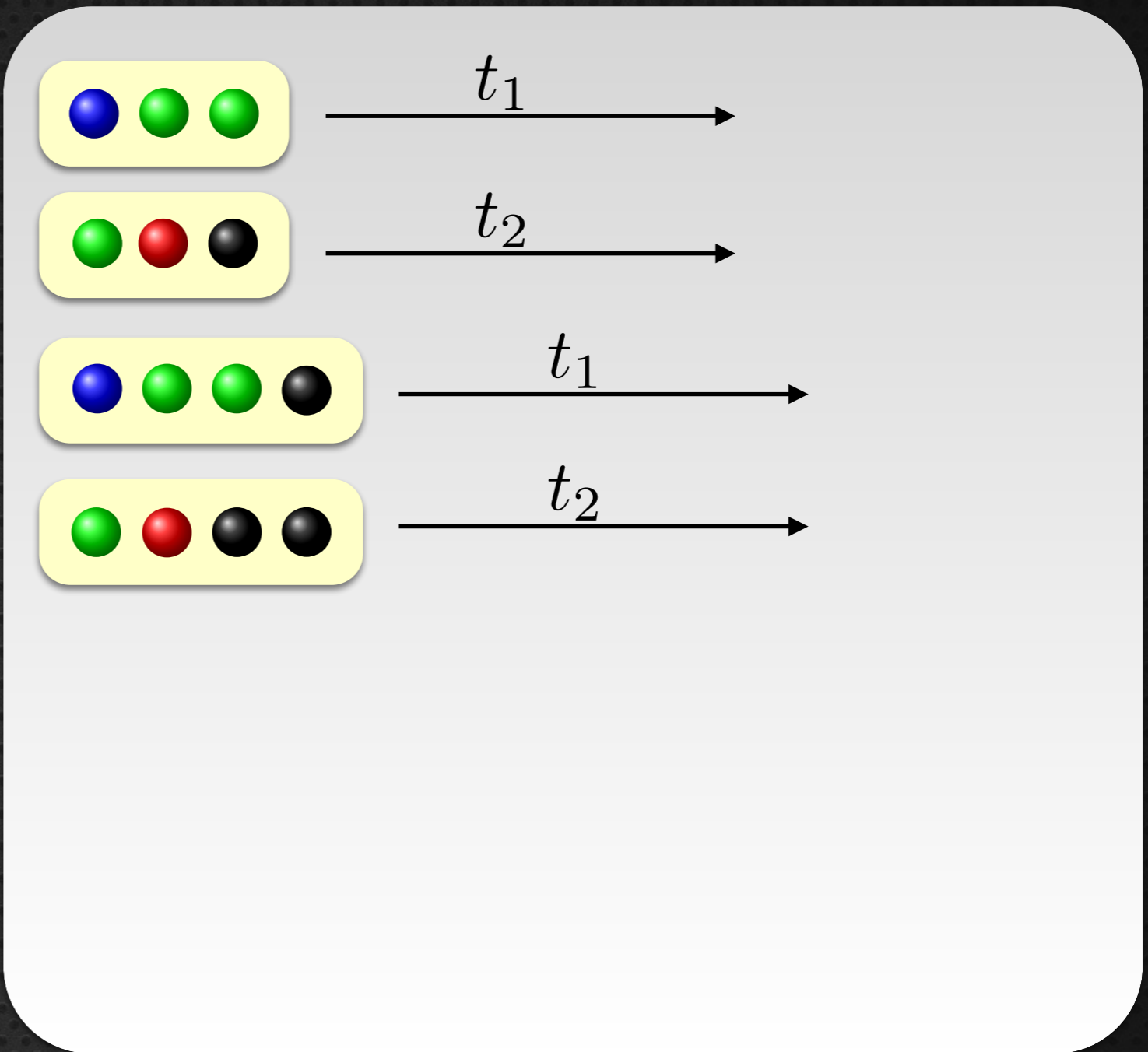
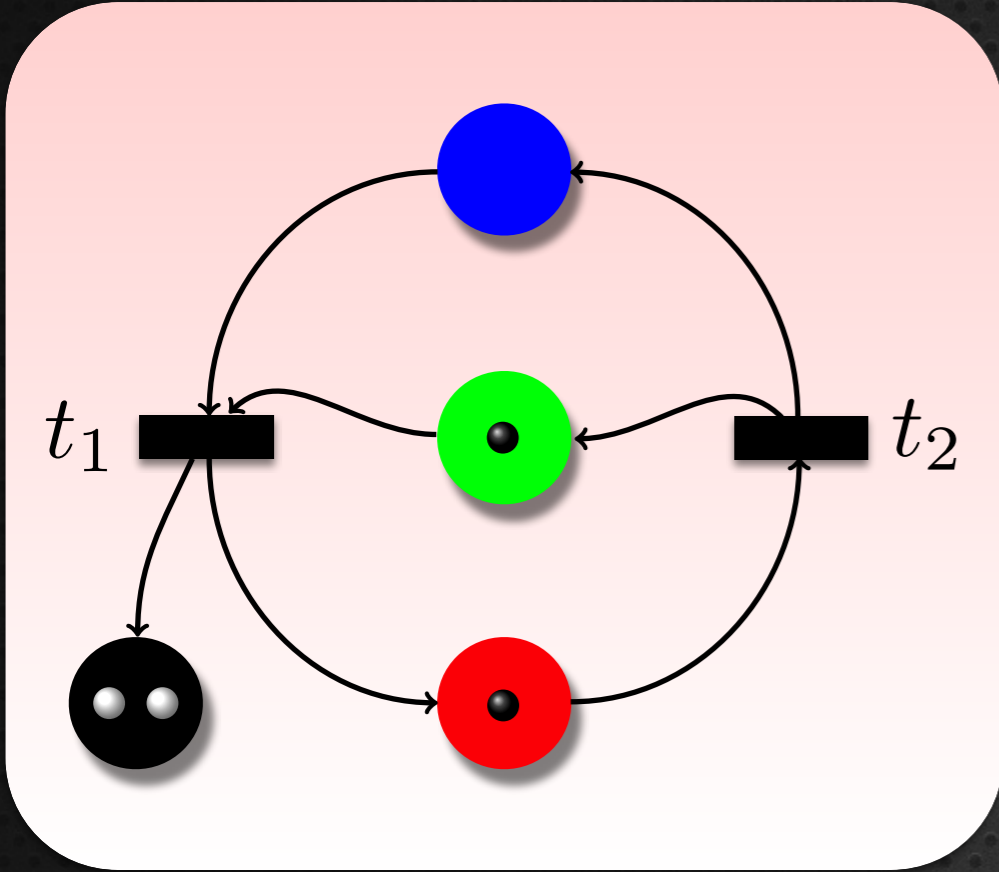
Transitions



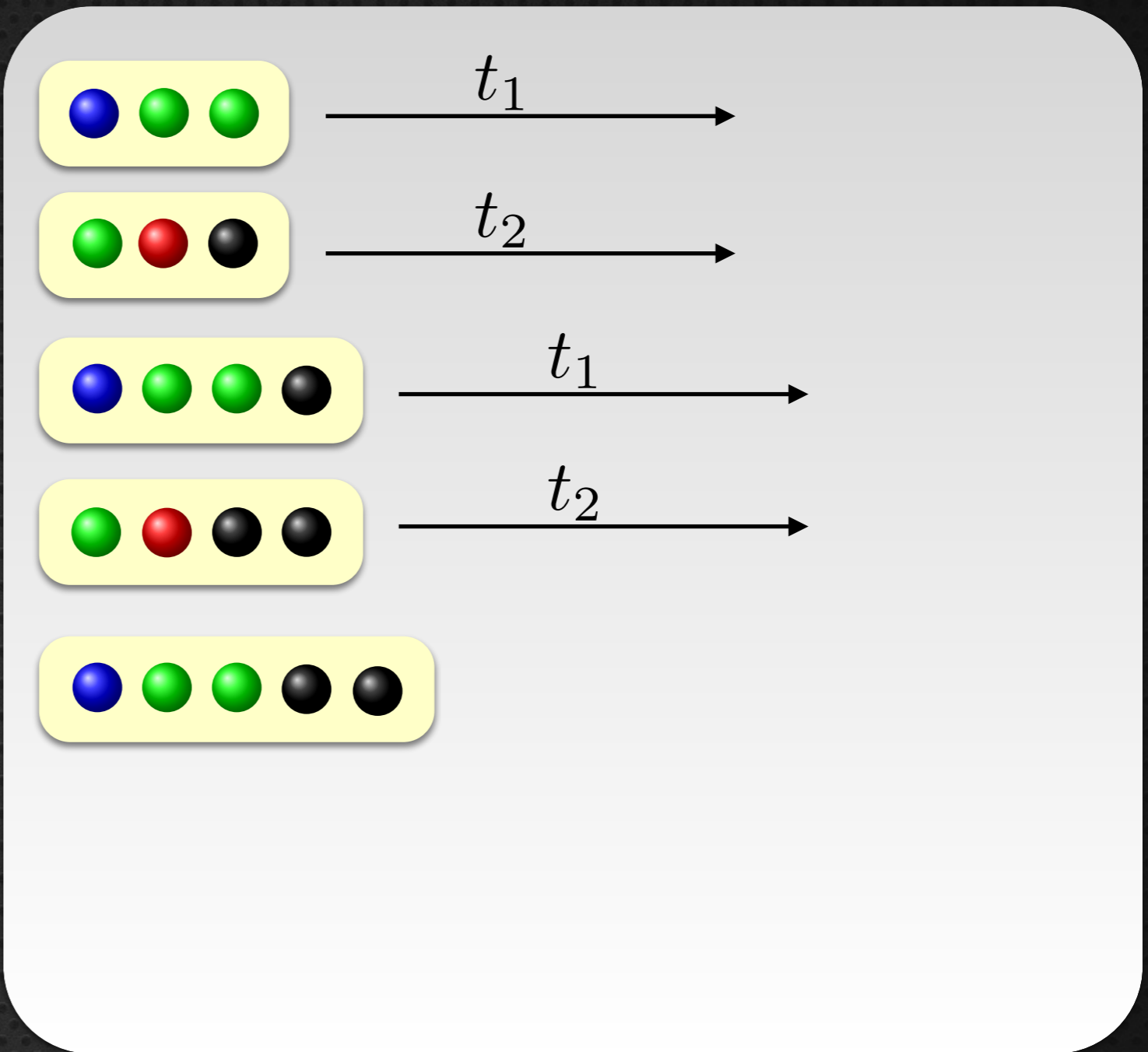
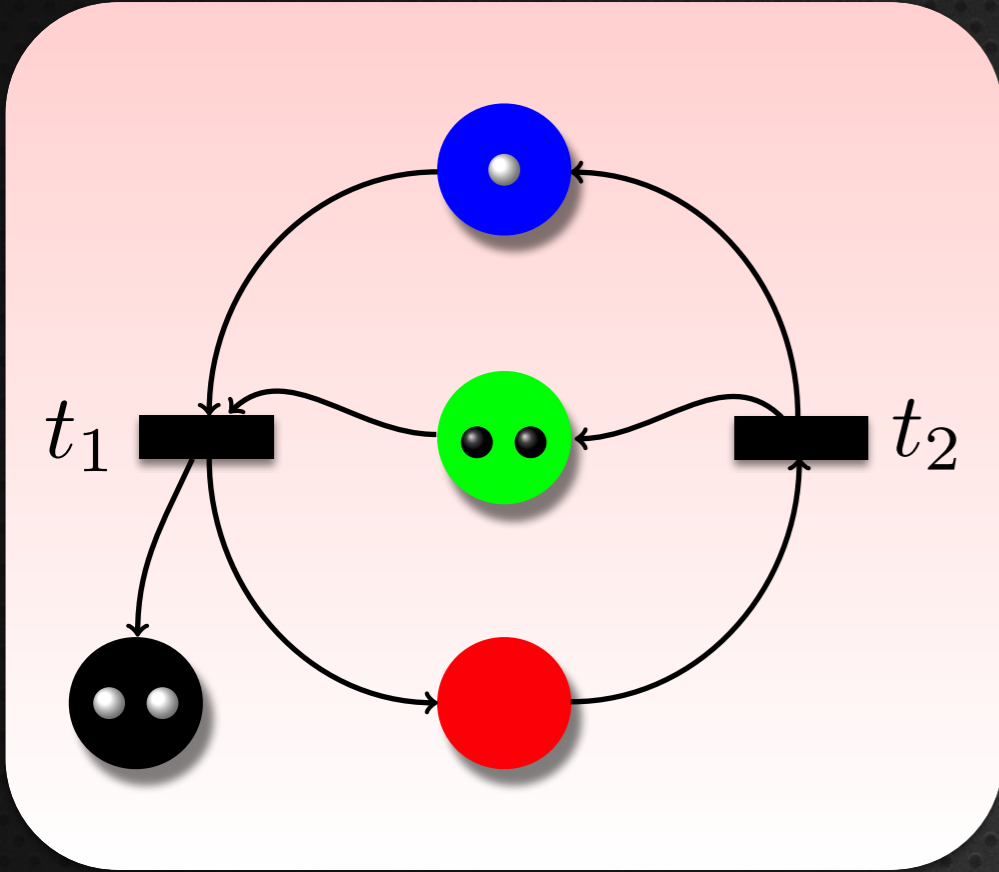
Transitions



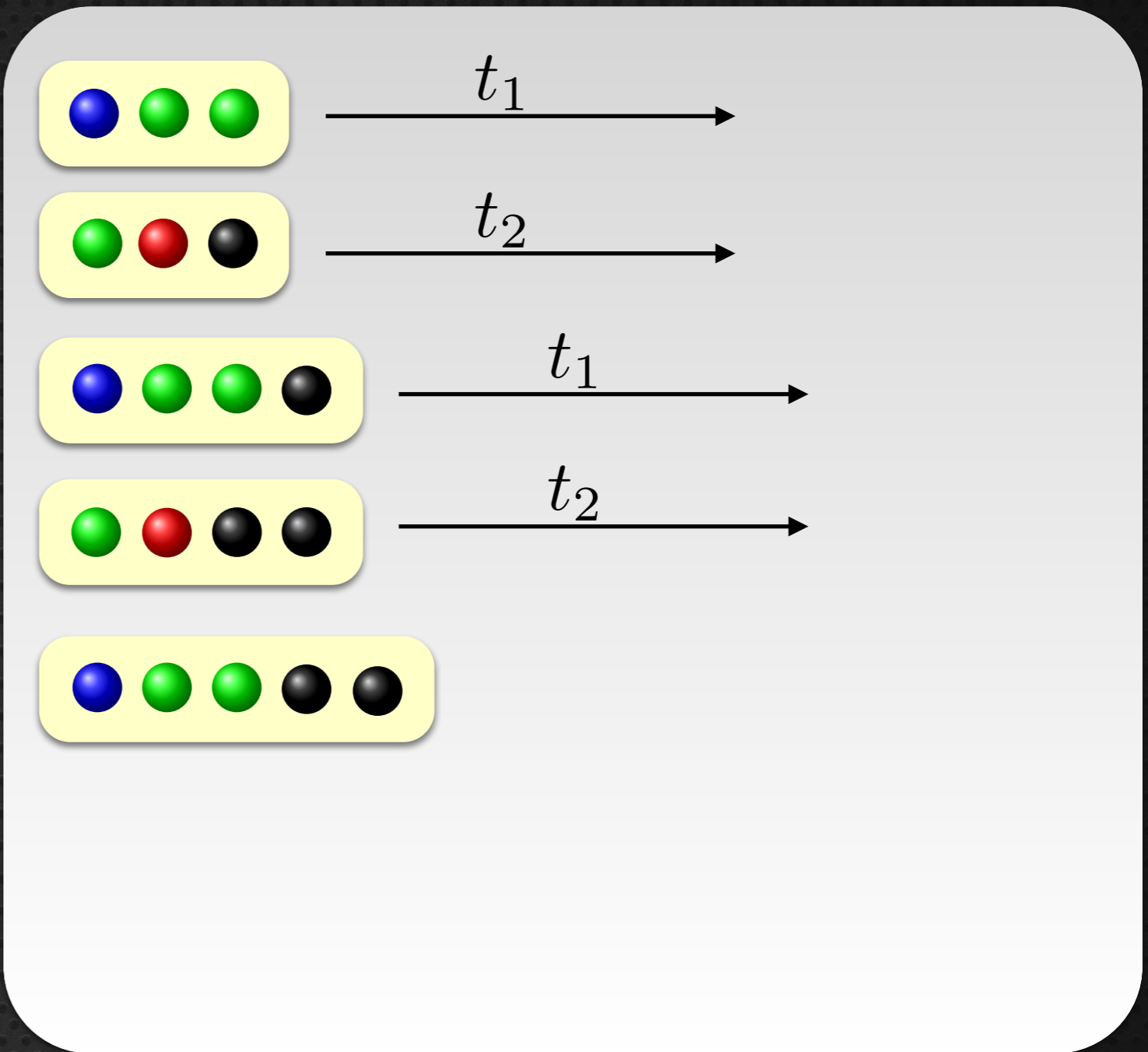
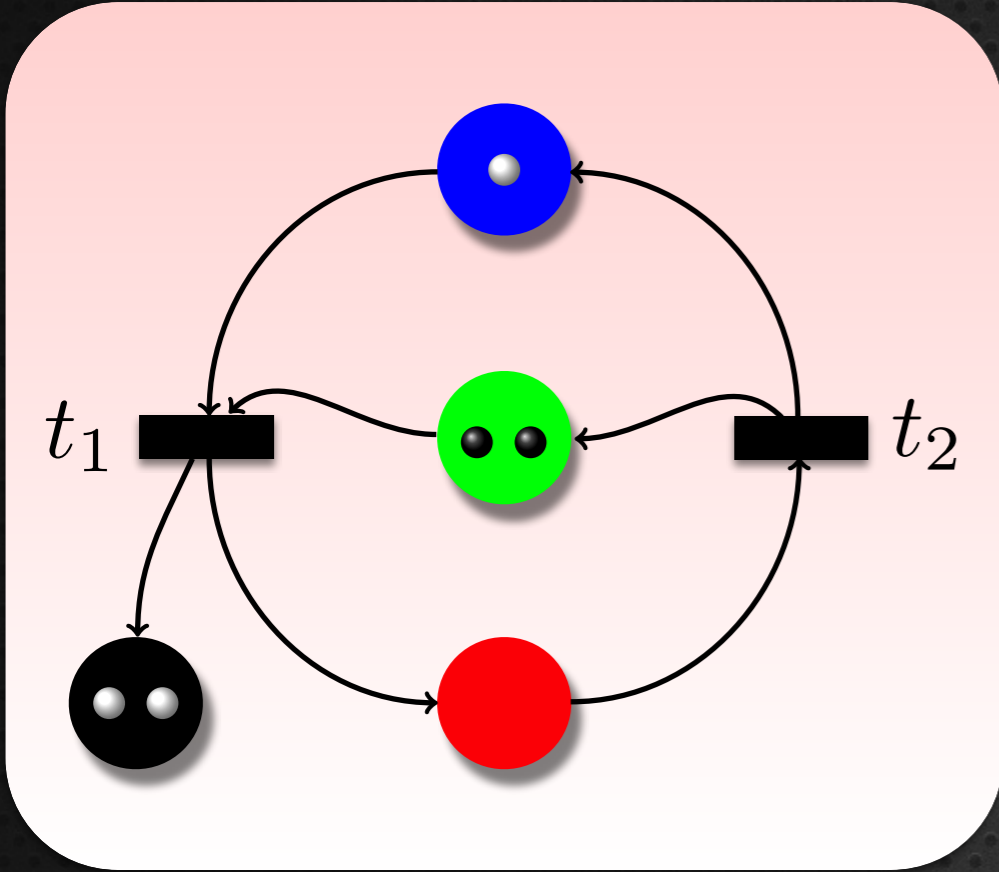
Transitions



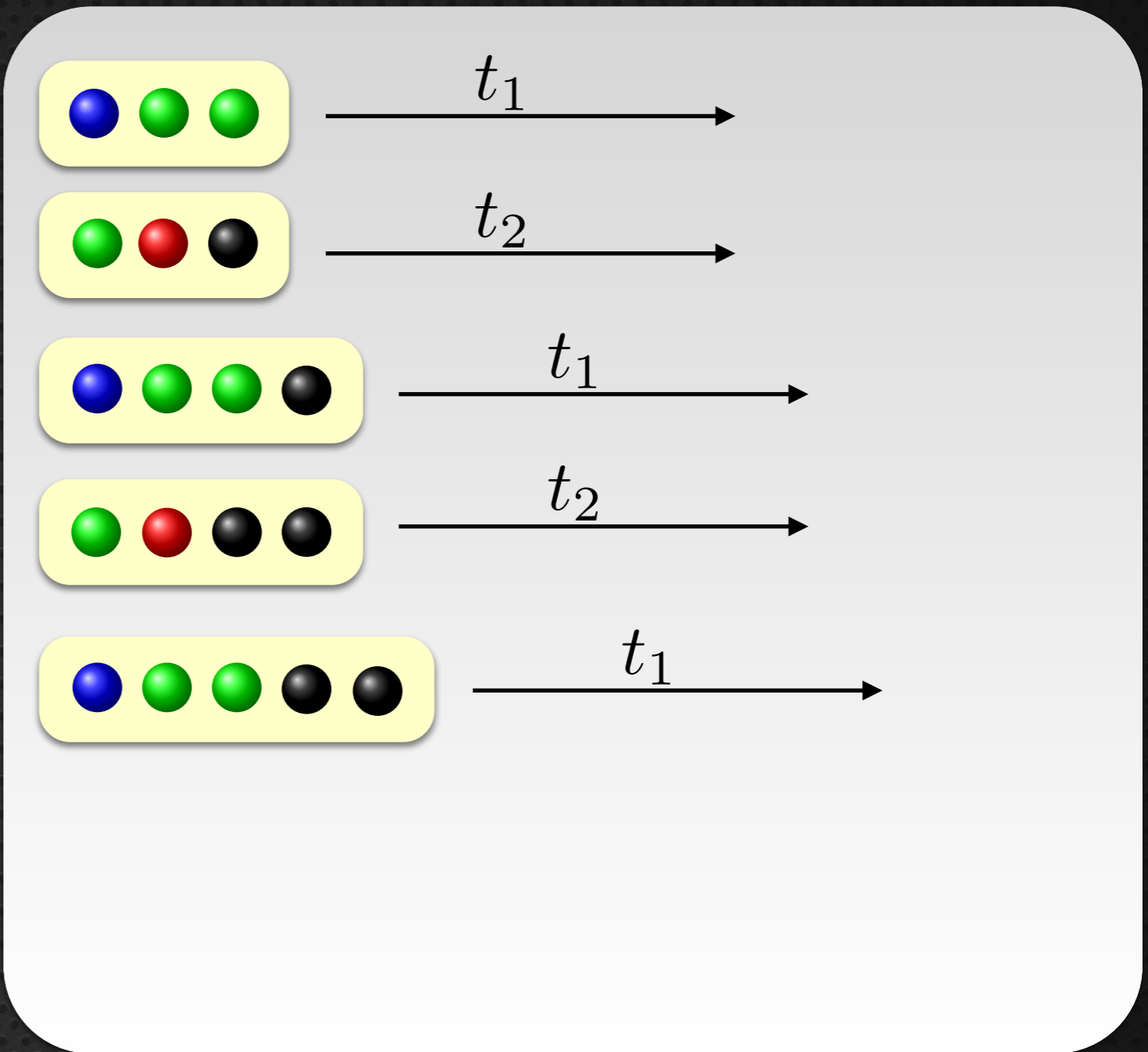
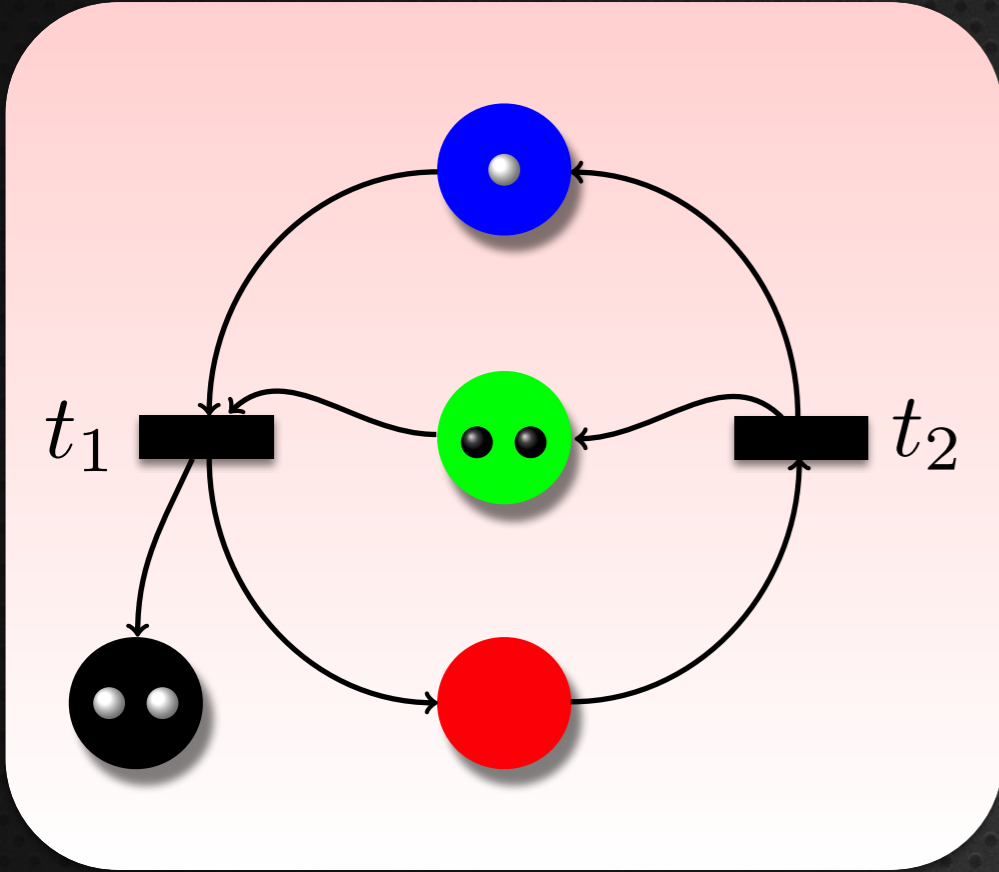
Transitions



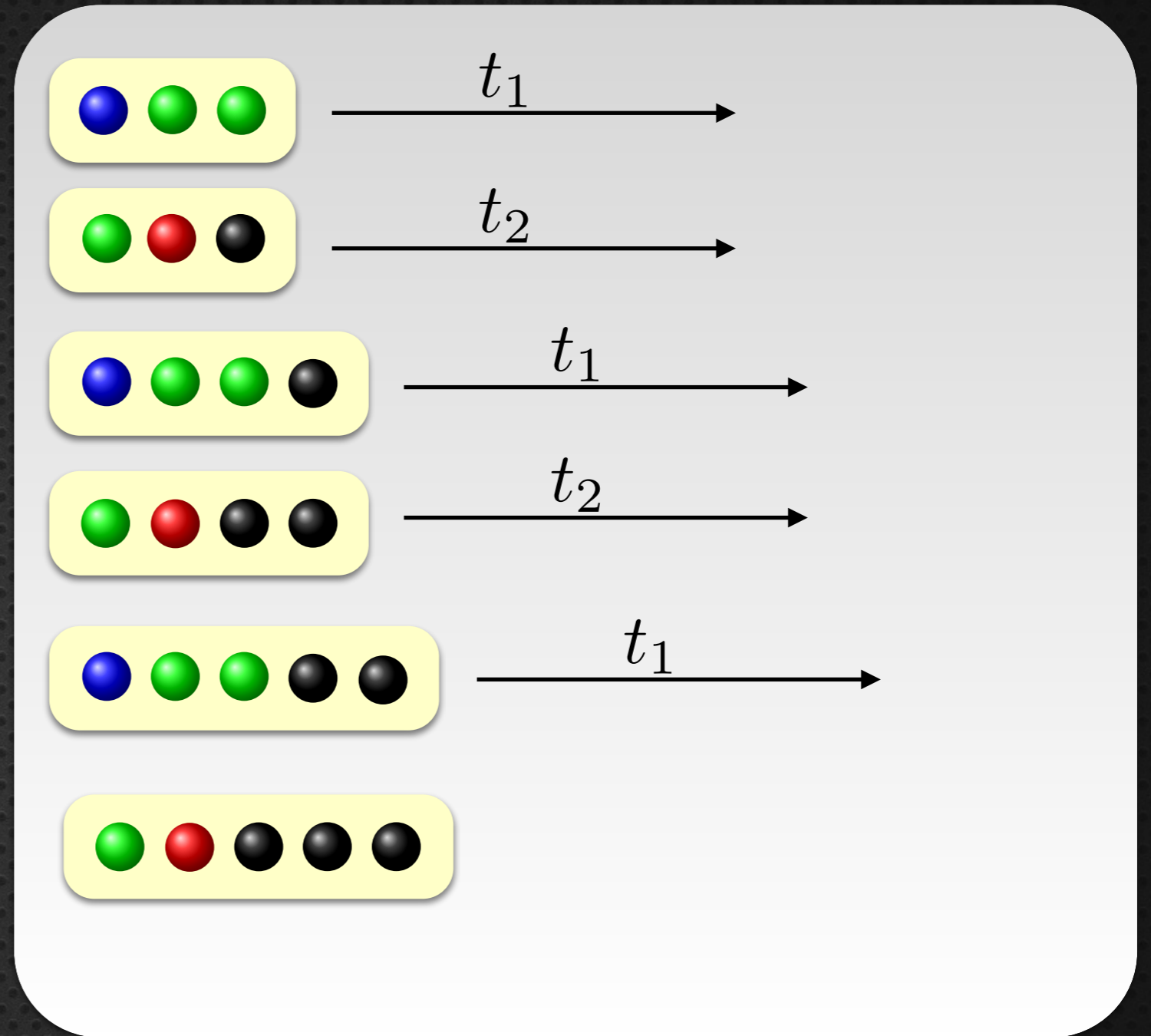
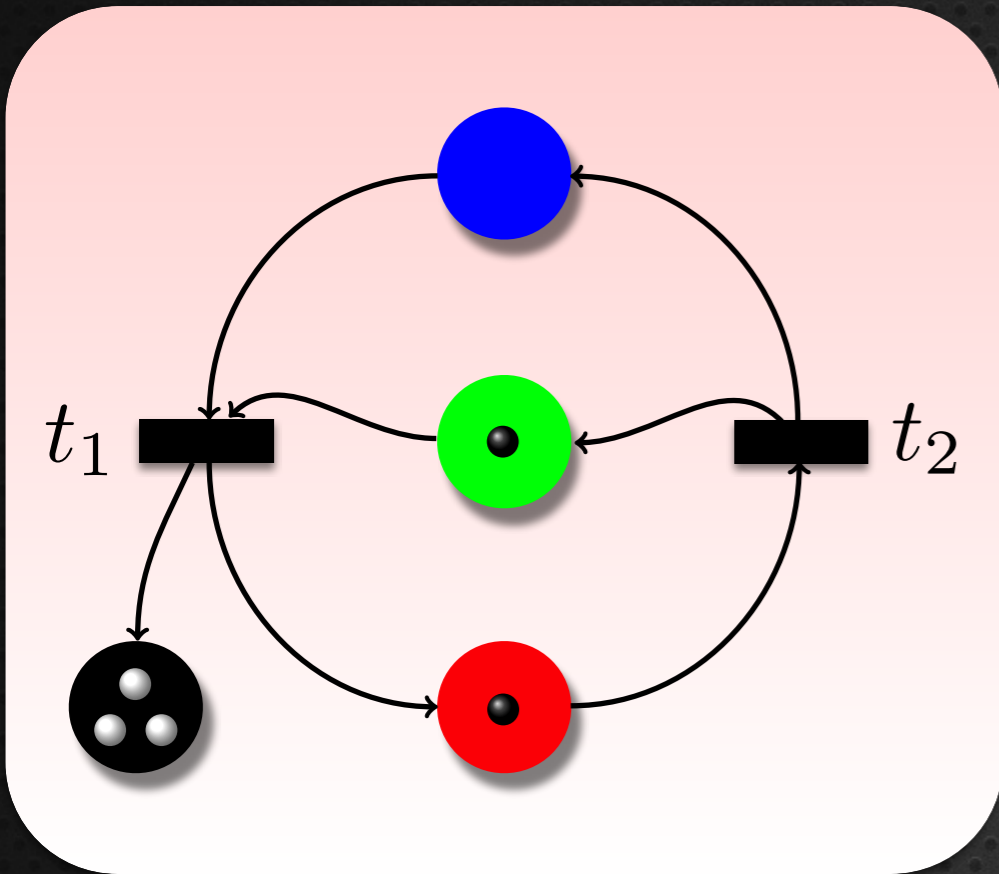
Transitions



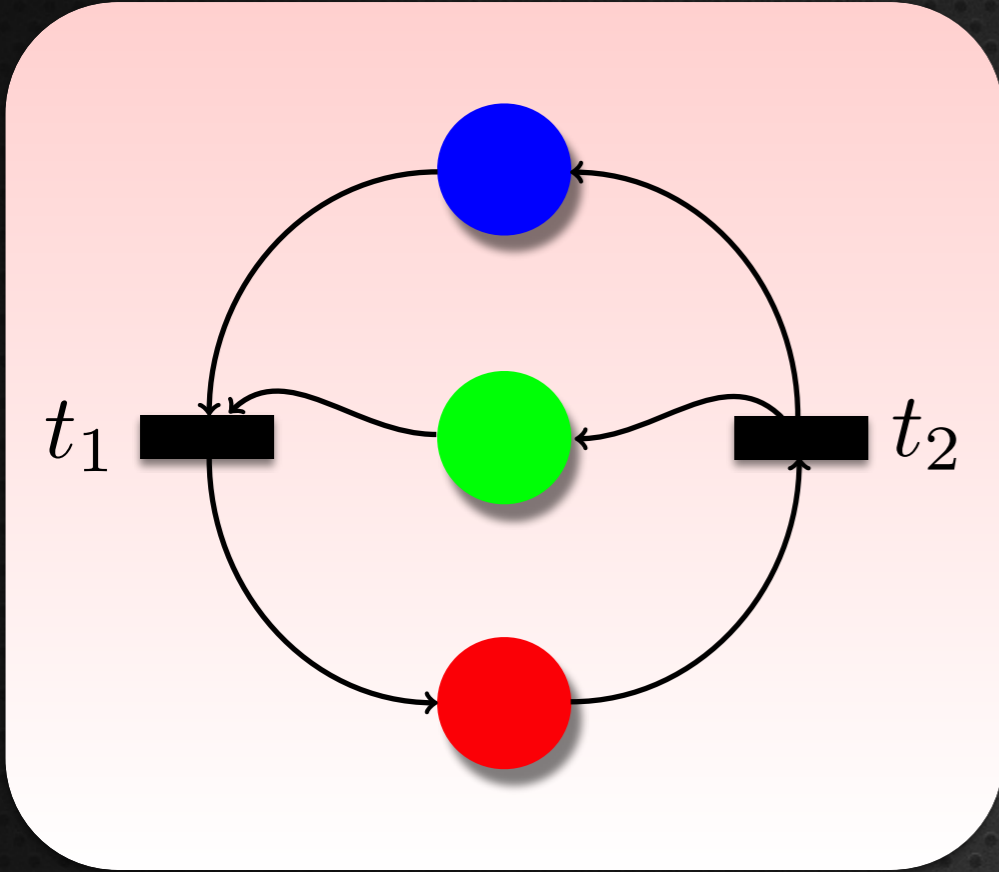
Transitions



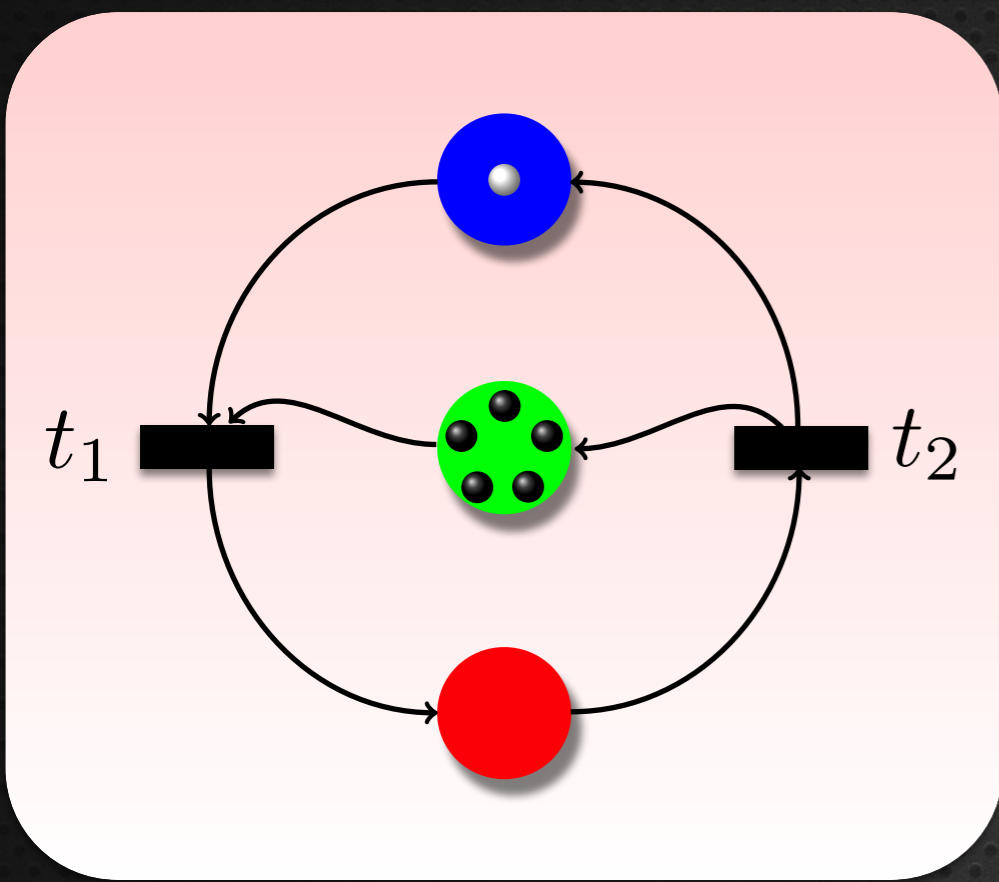
Transitions



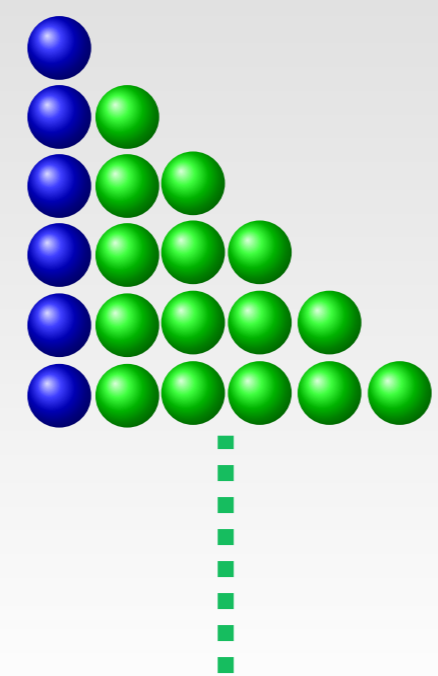
P Safety Properties



Safety Properties

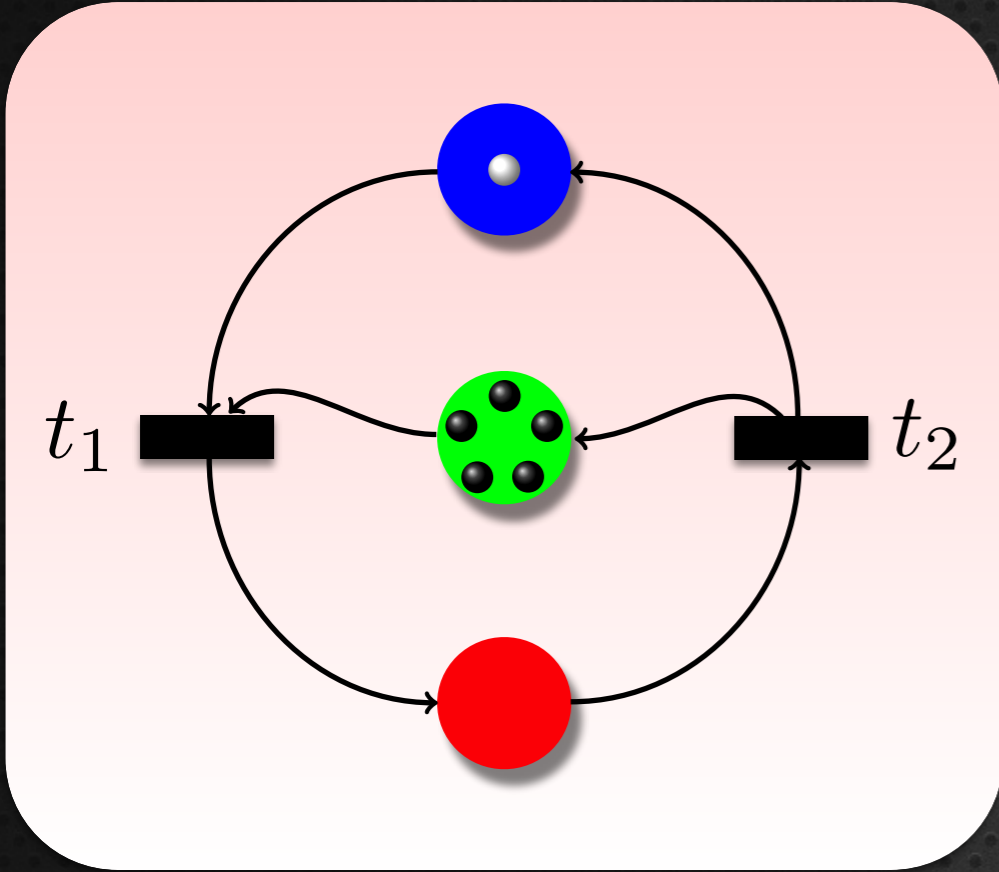


Initial Markings (*Init*)





- one 
- arbitrarily many 

Safety Properties

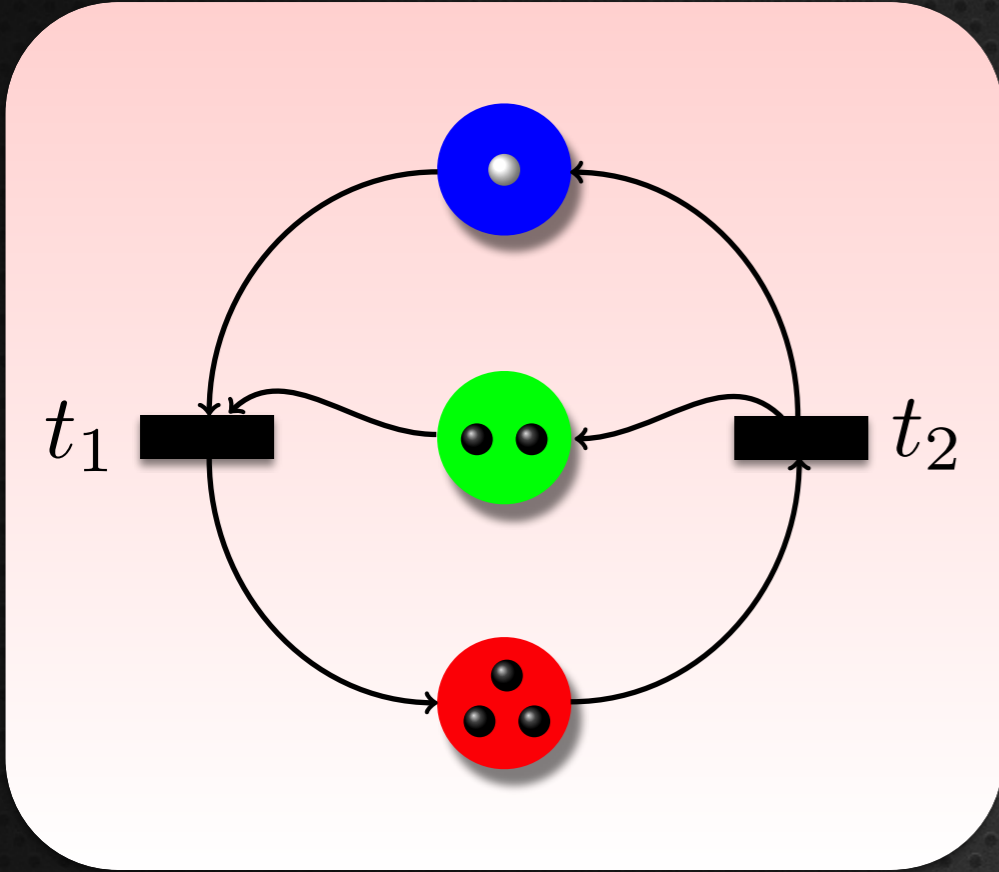


Initial Markings (*Init*)

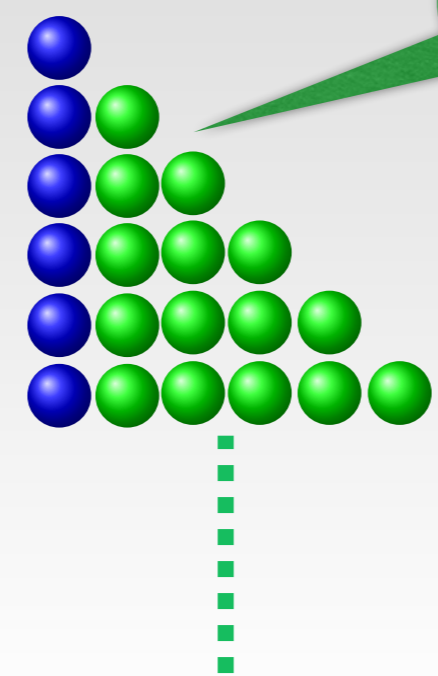
infinitely many

- one 
- arbitrarily many 

Safety Properties



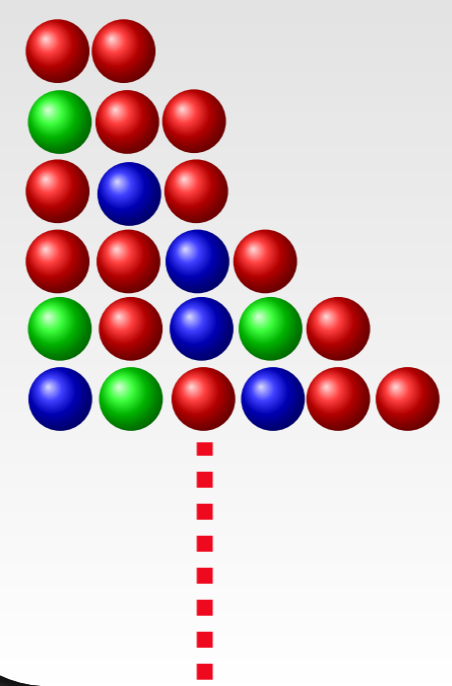
Initial Markings (*Init*)



infinitely many

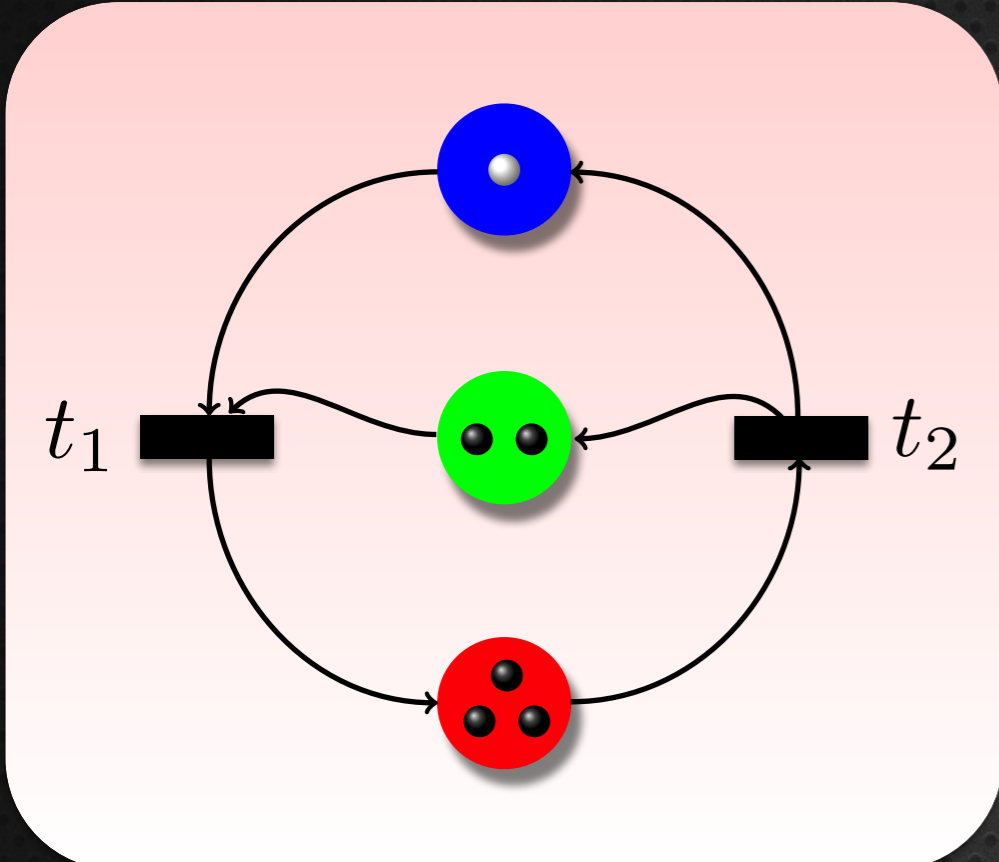
- one 
- arbitrarily many 

Bad Markings (*Bad*)

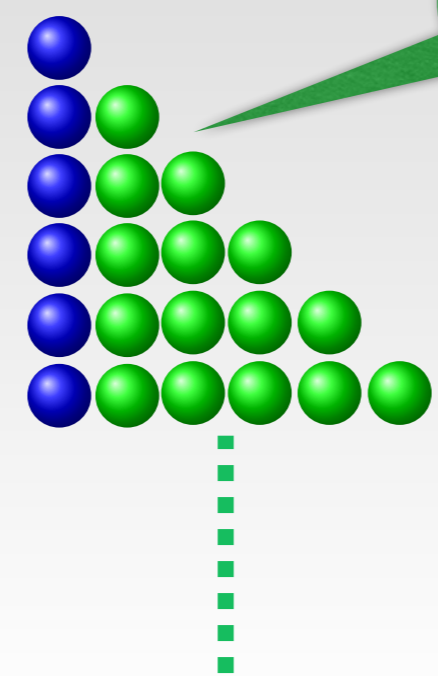


- at least two 

Safety Properties



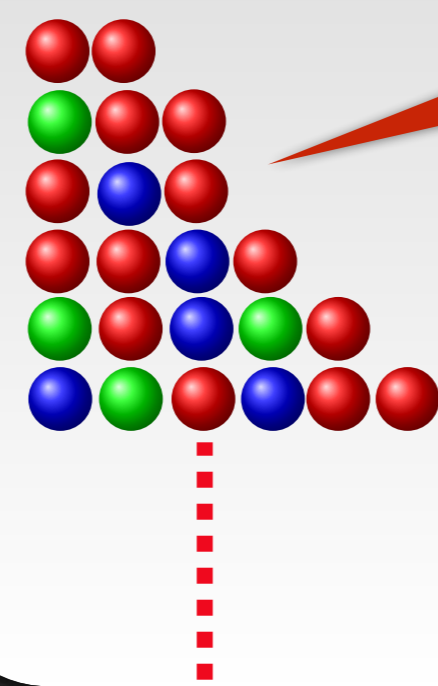
Initial Markings (*Init*)



infinitely many

- one 
- arbitrarily many 

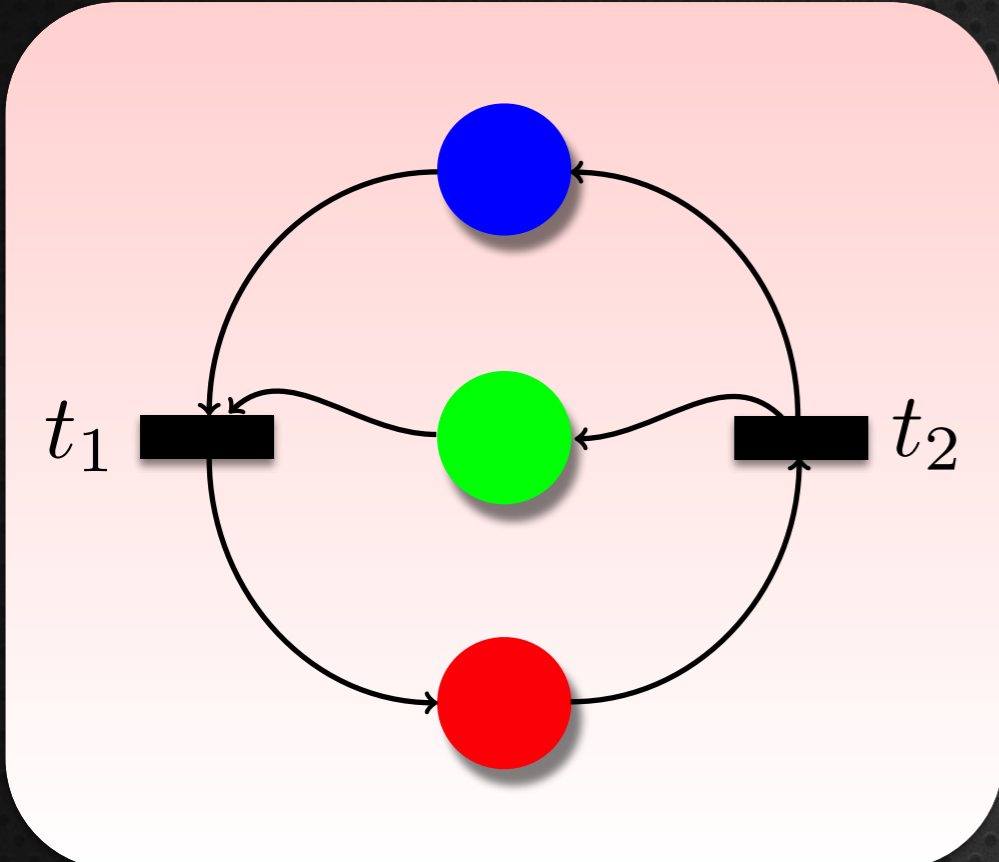
Bad Markings (*Bad*)



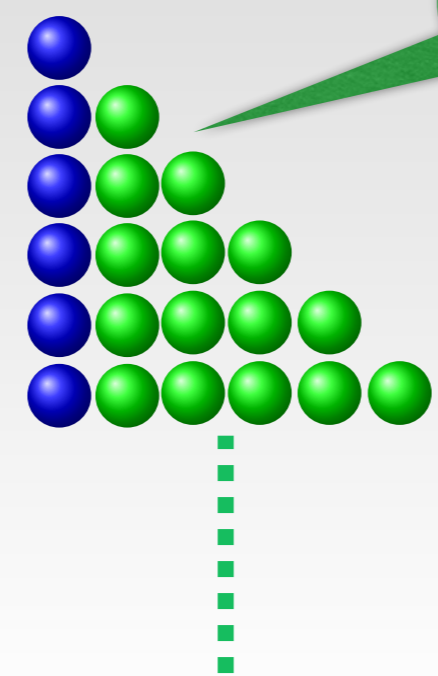
infinitely many

- at least two 

Safety Properties



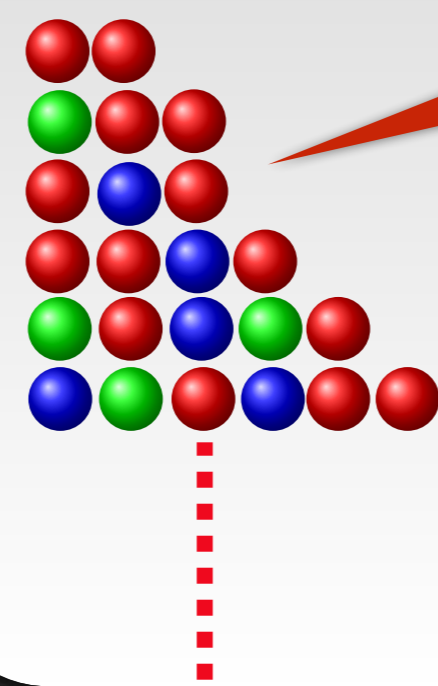
Initial Markings (*Init*)



infinitely many

- one 
- arbitrarily many 

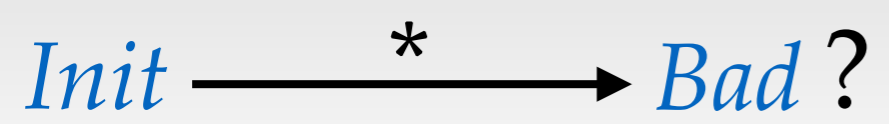
Bad Markings (*Bad*)



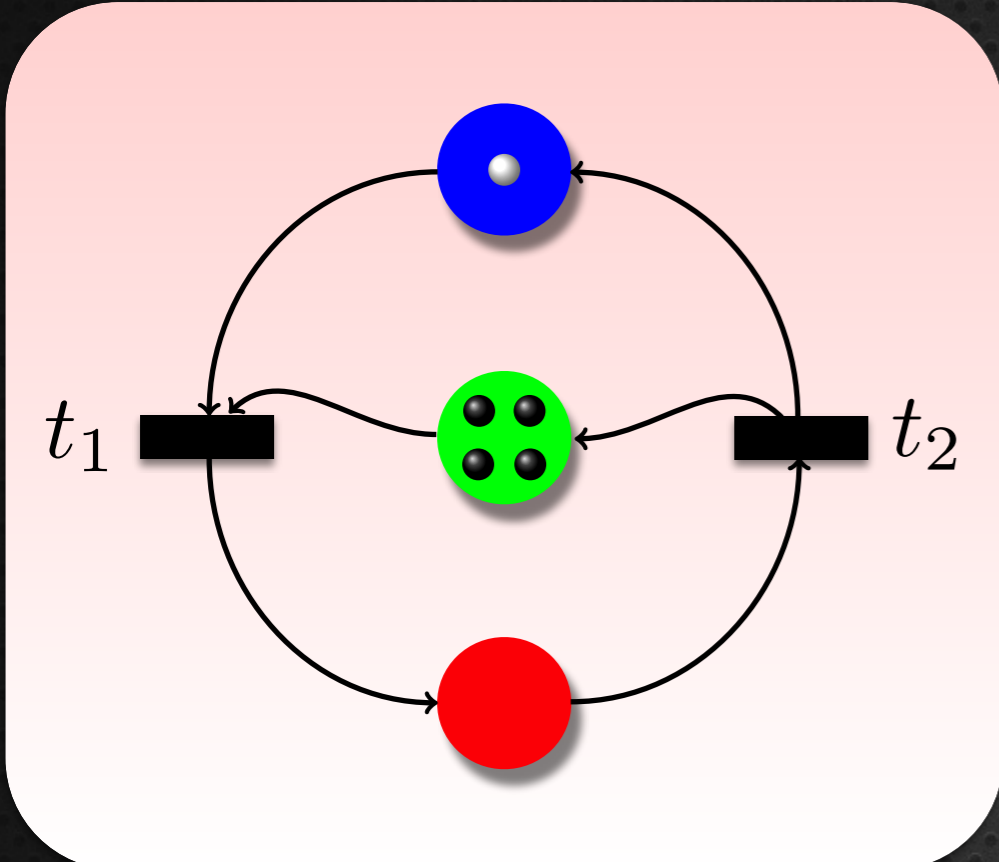
infinitely many

- at least two 

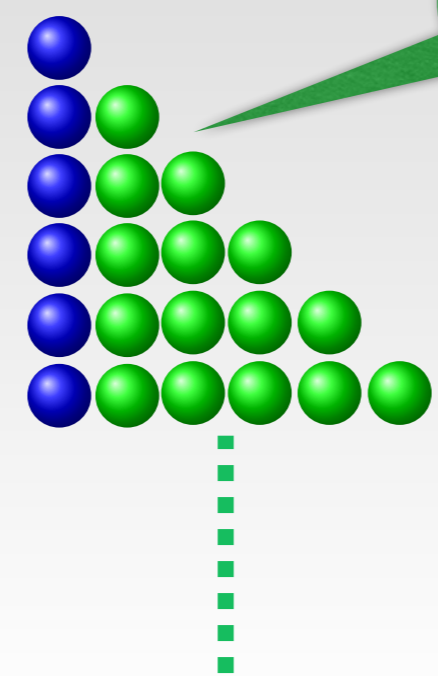
Safety Property



Safety Properties



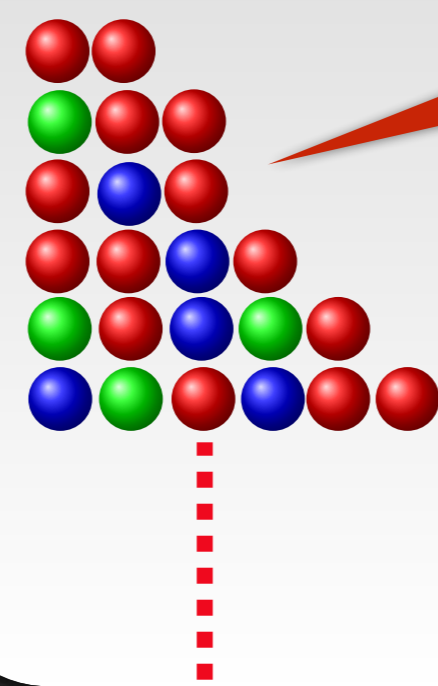
Initial Markings (*Init*)



infinitely many

- one ●
- arbitrarily many ●

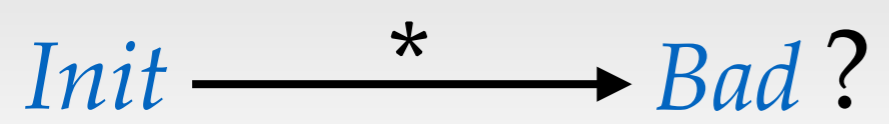
Bad Markings (*Bad*)



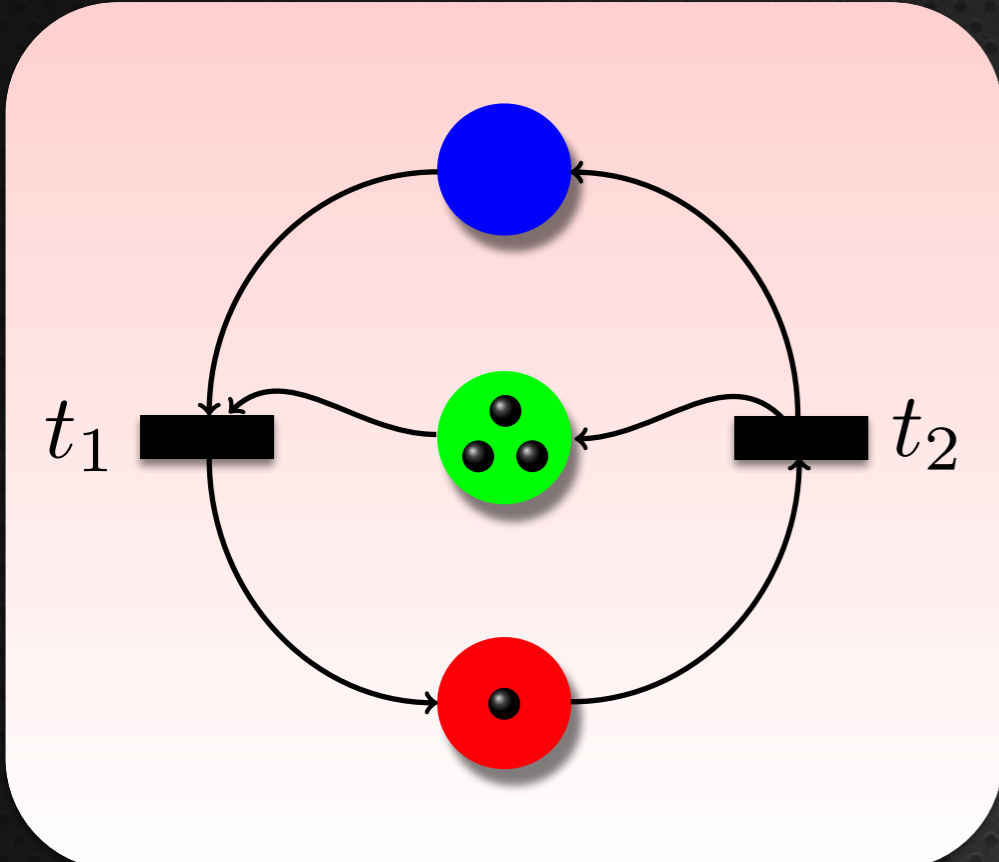
infinitely many

- at least two ●

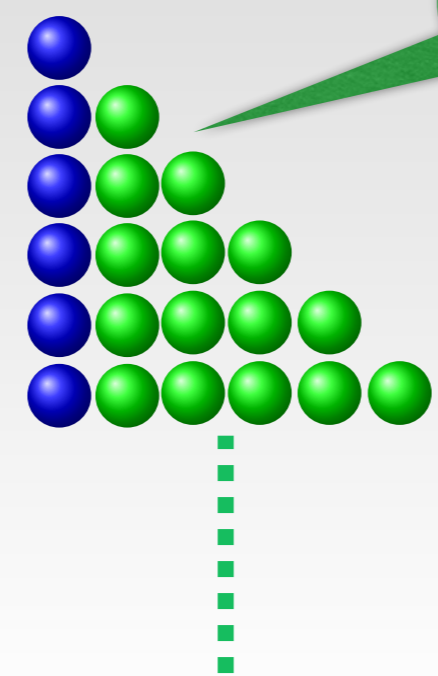
Safety Property



Safety Properties



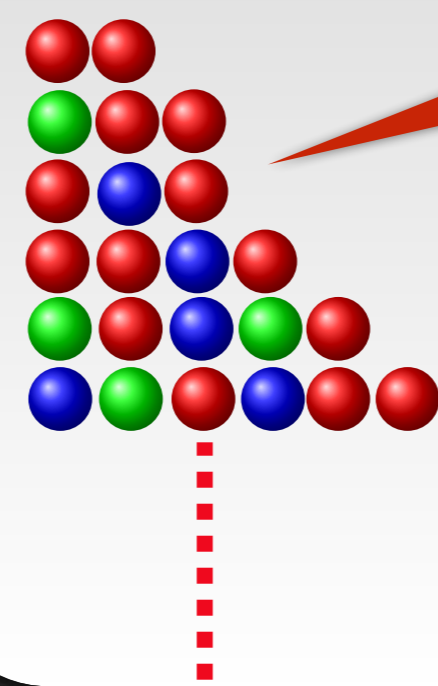
Initial Markings (*Init*)



infinitely many

- one ●
- arbitrarily many ●

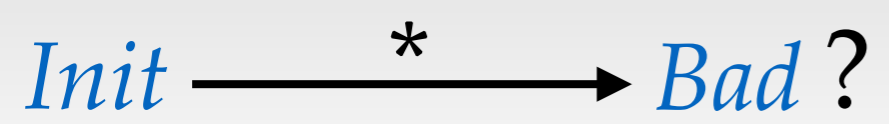
Bad Markings (*Bad*)



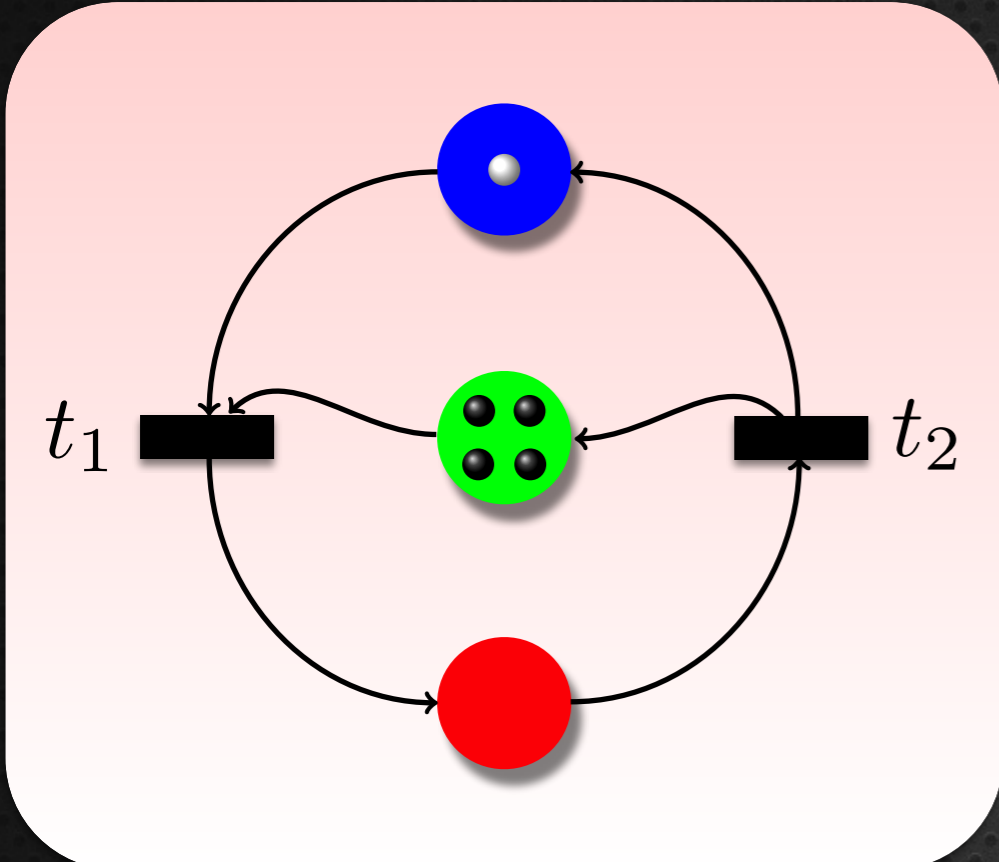
infinitely many

- at least two ●

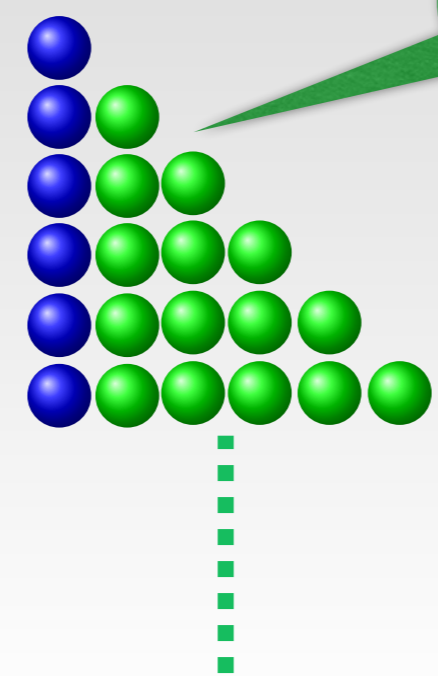
Safety Property



Safety Properties



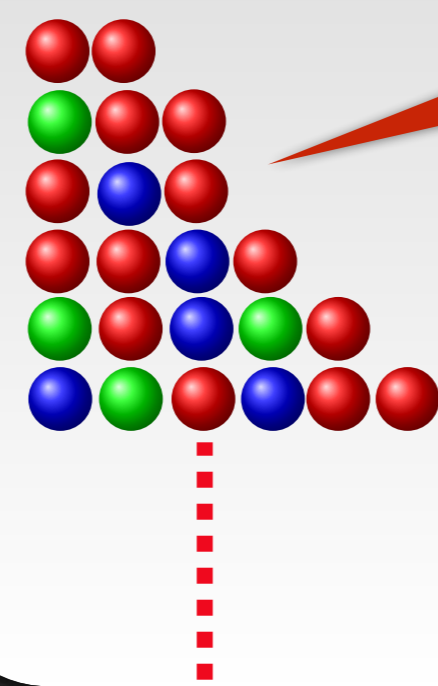
Initial Markings (*Init*)



infinitely many

- one 
- arbitrarily many 

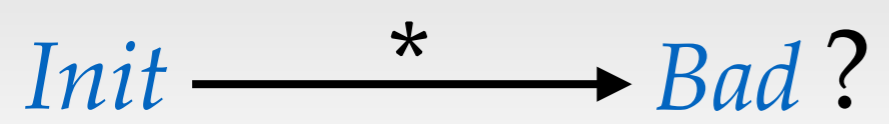
Bad Markings (*Bad*)



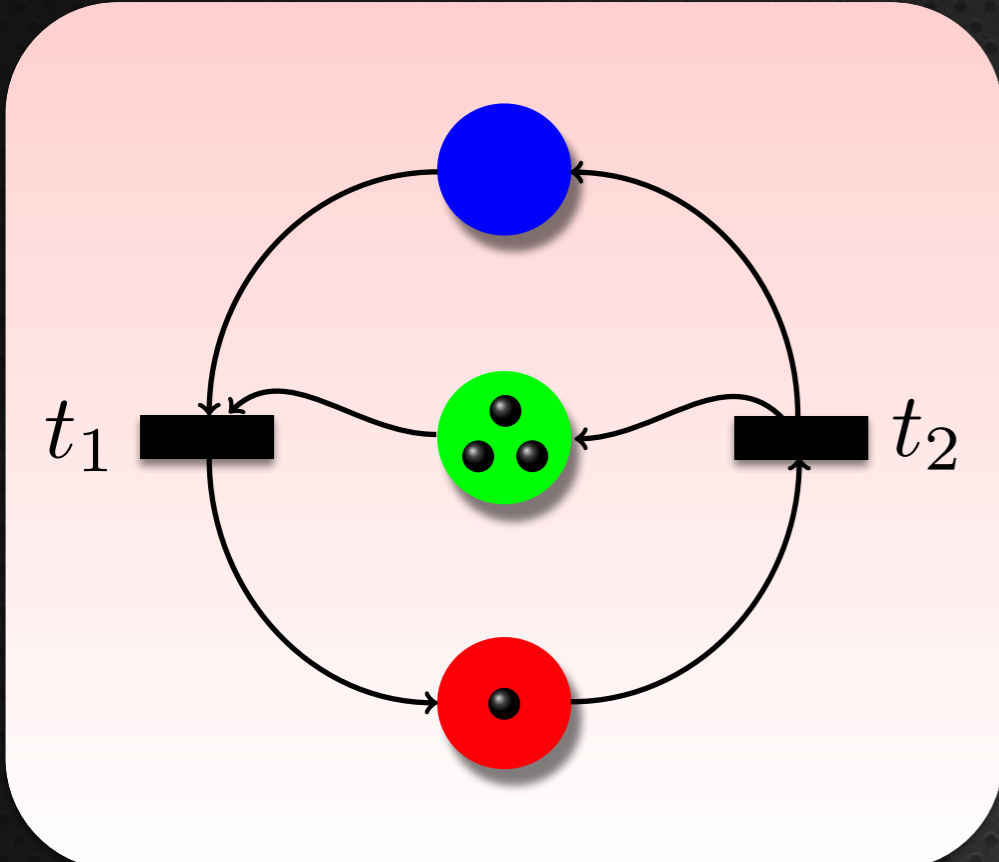
infinitely many

- at least two 

Safety Property





Safety Properties




Initial Markings (*Init*)

Diagram illustrating the initial marking (*Init*). It shows a triangular arrangement of tokens: one blue token at the top left, and a row of green tokens below it. A green speech bubble points to the green tokens with the text "infinitely many". A vertical dashed green line indicates the continuation of the row.

- one 
- arbitrarily many 

Bad Markings (*Bad*)

Diagram illustrating a bad marking (*Bad*). It shows a triangular arrangement of tokens: a row of red tokens at the top, followed by a row of green and red tokens, then a row of red, blue, and red tokens, then a row of red, red, blue, and red tokens, then a row of green, red, blue, green, and red tokens, and finally a row of blue, green, red, blue, red, and red tokens. A red speech bubble points to the top row of red tokens with the text "infinitely many". A vertical dashed red line indicates the continuation of the row.

- at least two 

Safety Property



Init $\xrightarrow{*}$ *Bad* ?

Safety Properties




Initial Markings (*Init*)

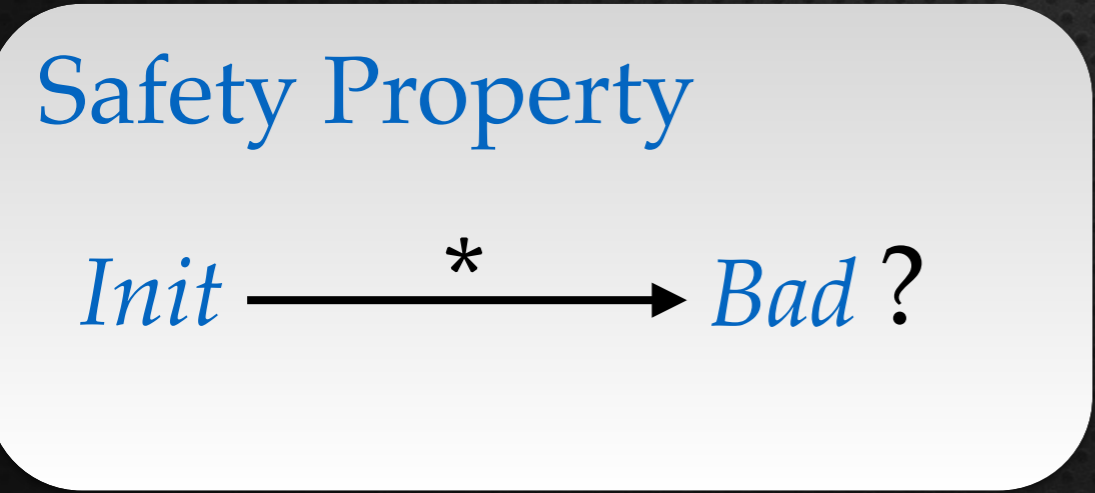
infinitely many

- one 
- arbitrarily many 

Bad Markings (*Bad*)

infinitely many

- at least two 



Safety Properties

Initial Markings (*Init*)

How to check safety properties?

infinitely many

- one 
- arbitrarily many 

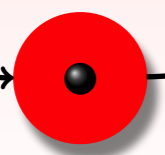
Bad

- Ordering
- Monotonicity
- Upward Closed sets
- Predecessors
- Backward Reachability

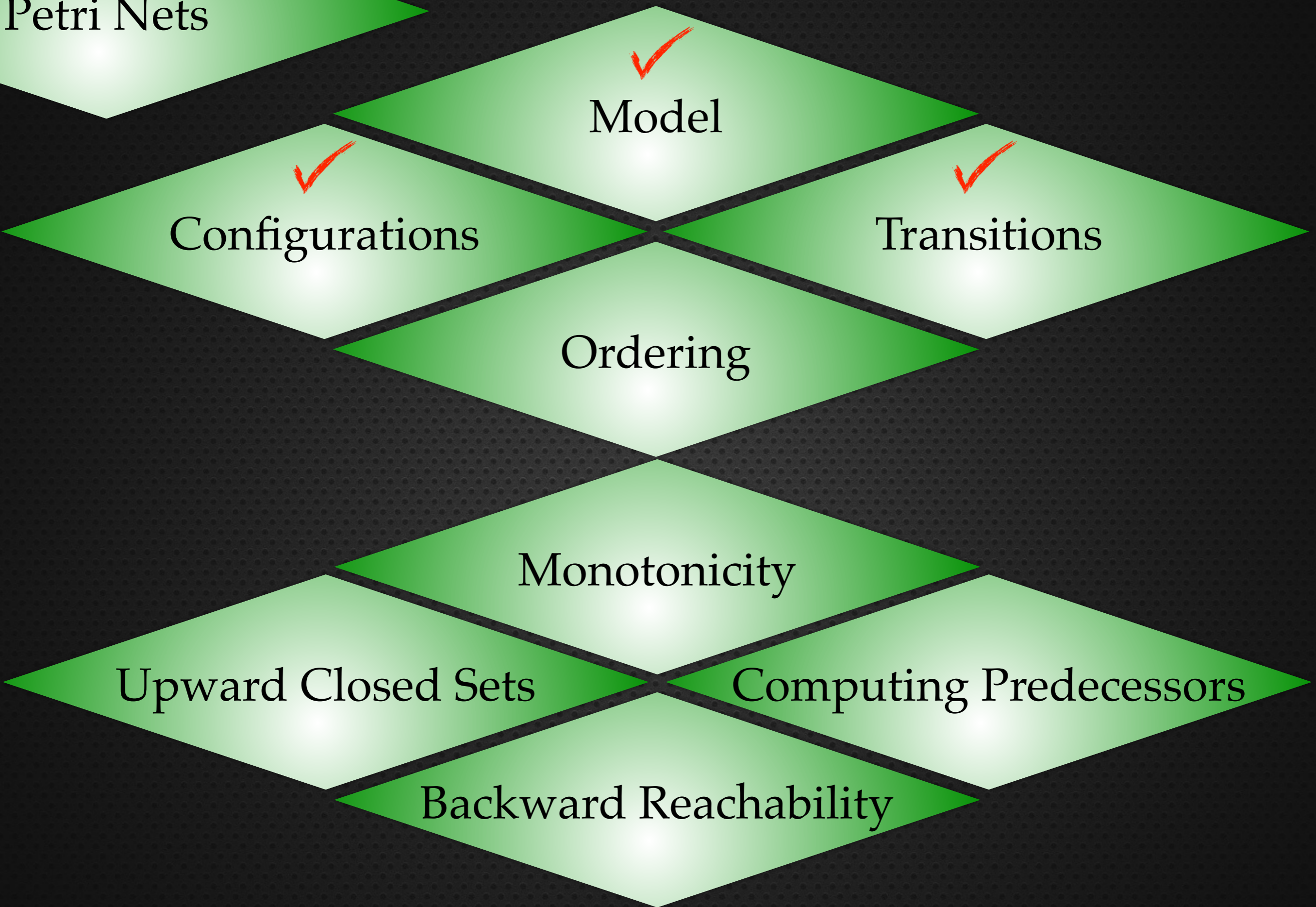
Safety Property



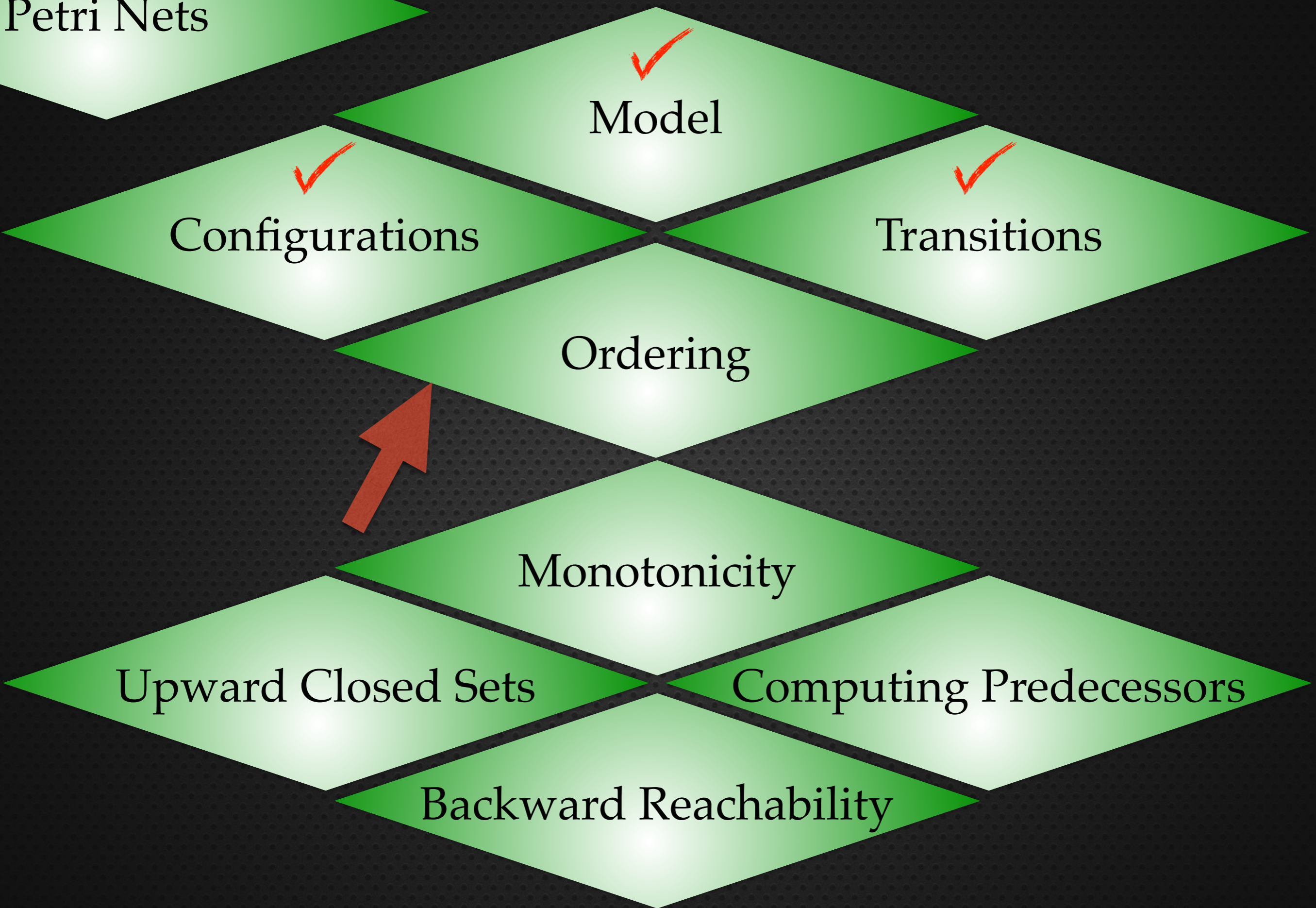
t_1



Petri Nets

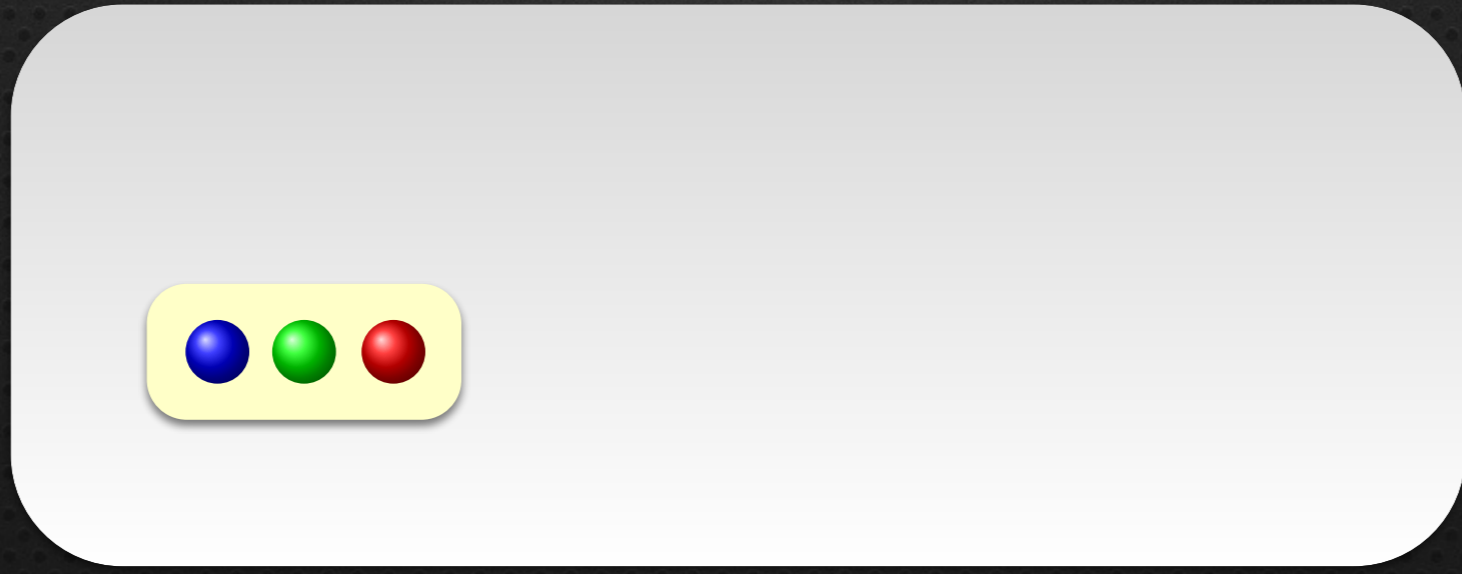
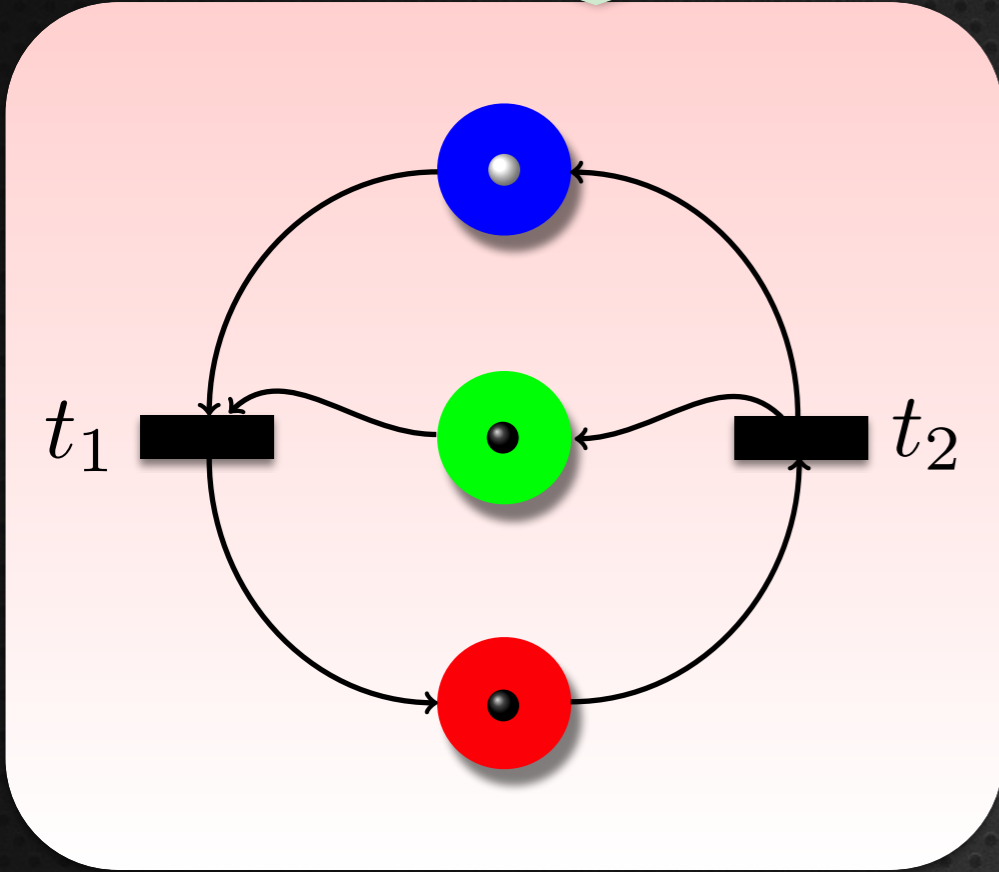


Petri Nets

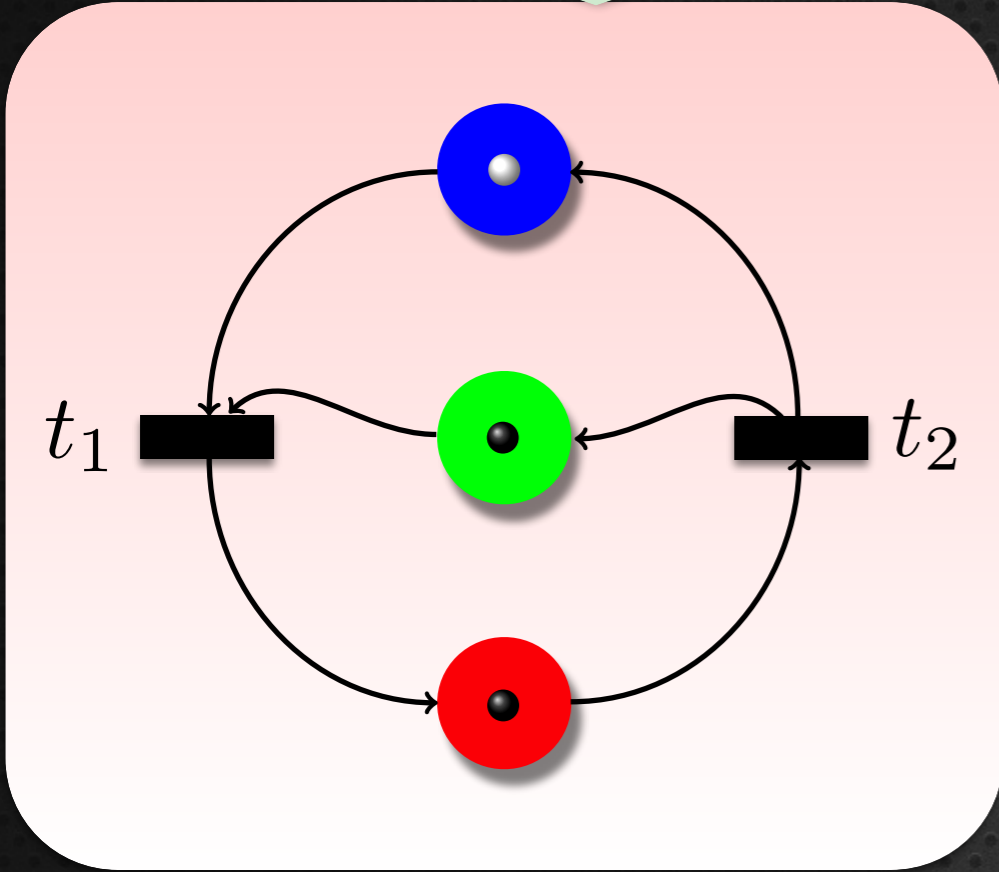


Ordering

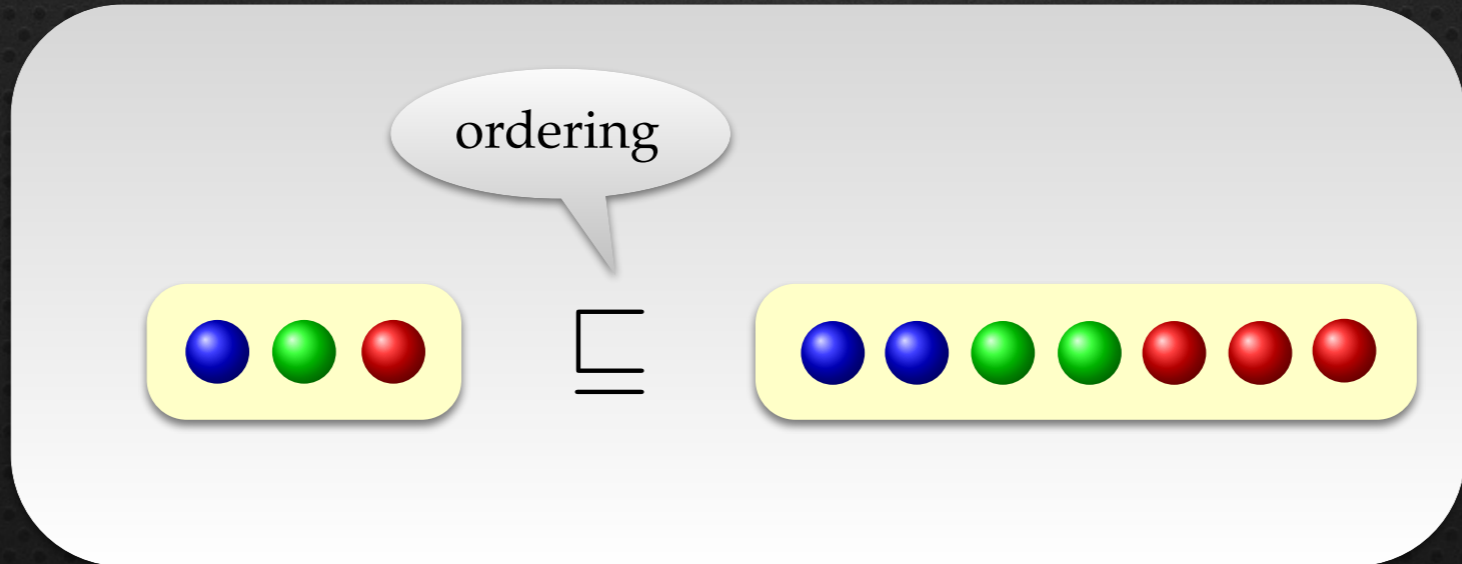
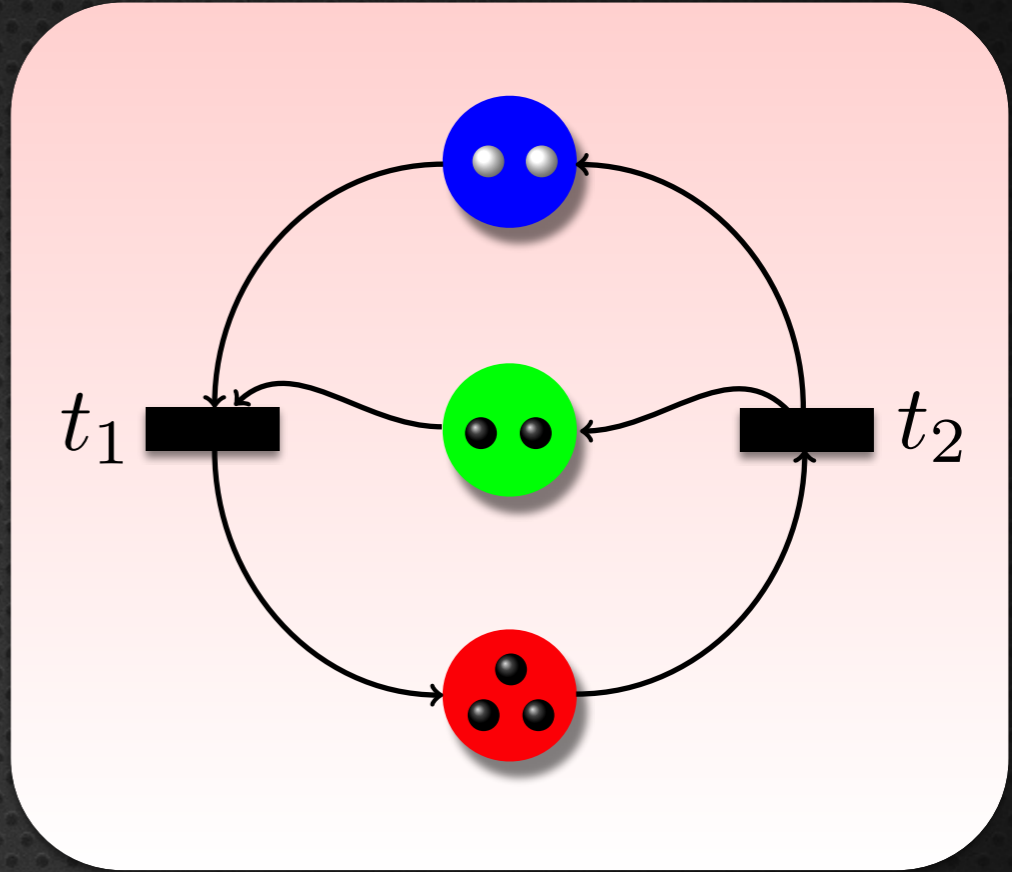
Petri Net Ordering



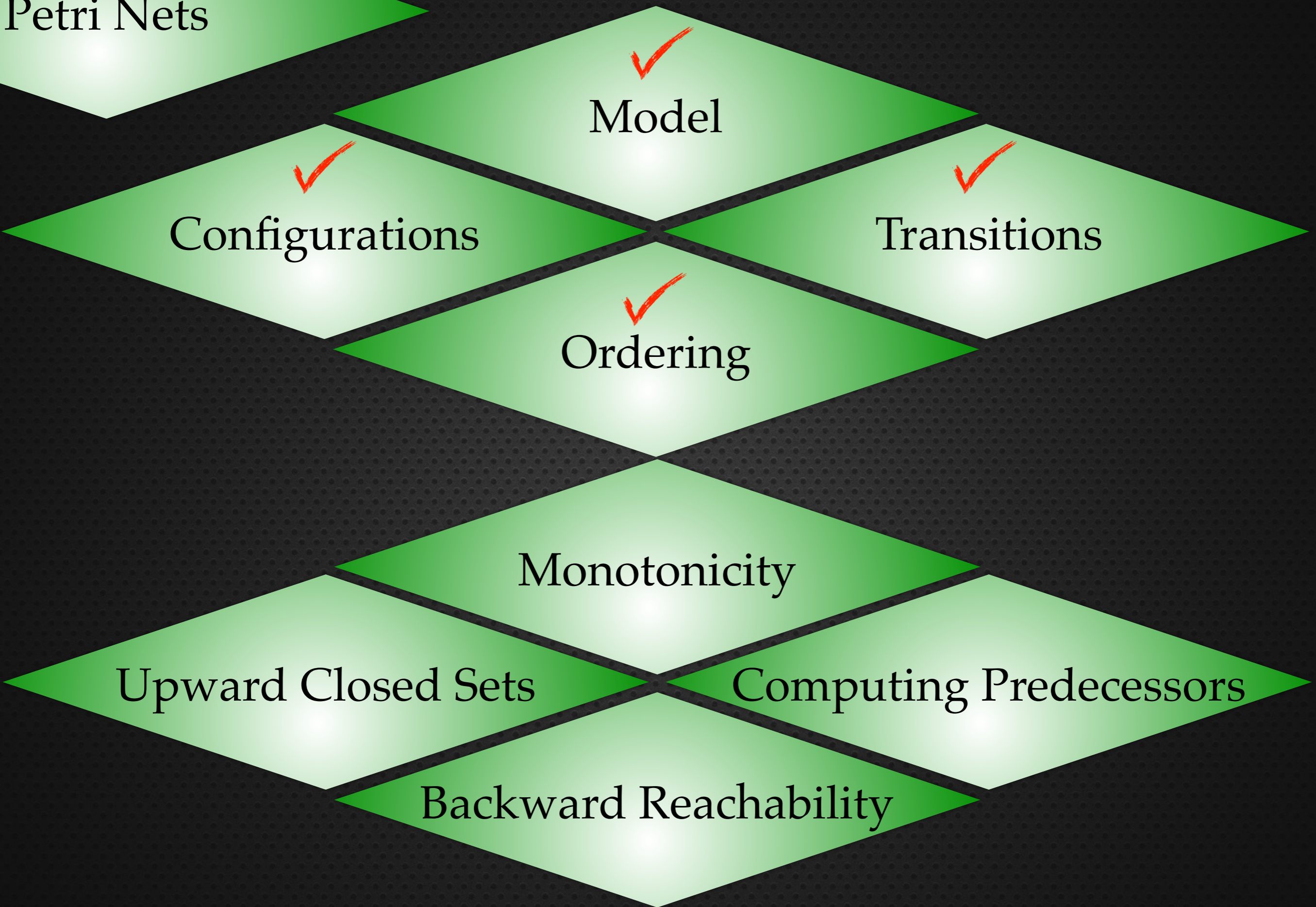
Petri Net Ordering



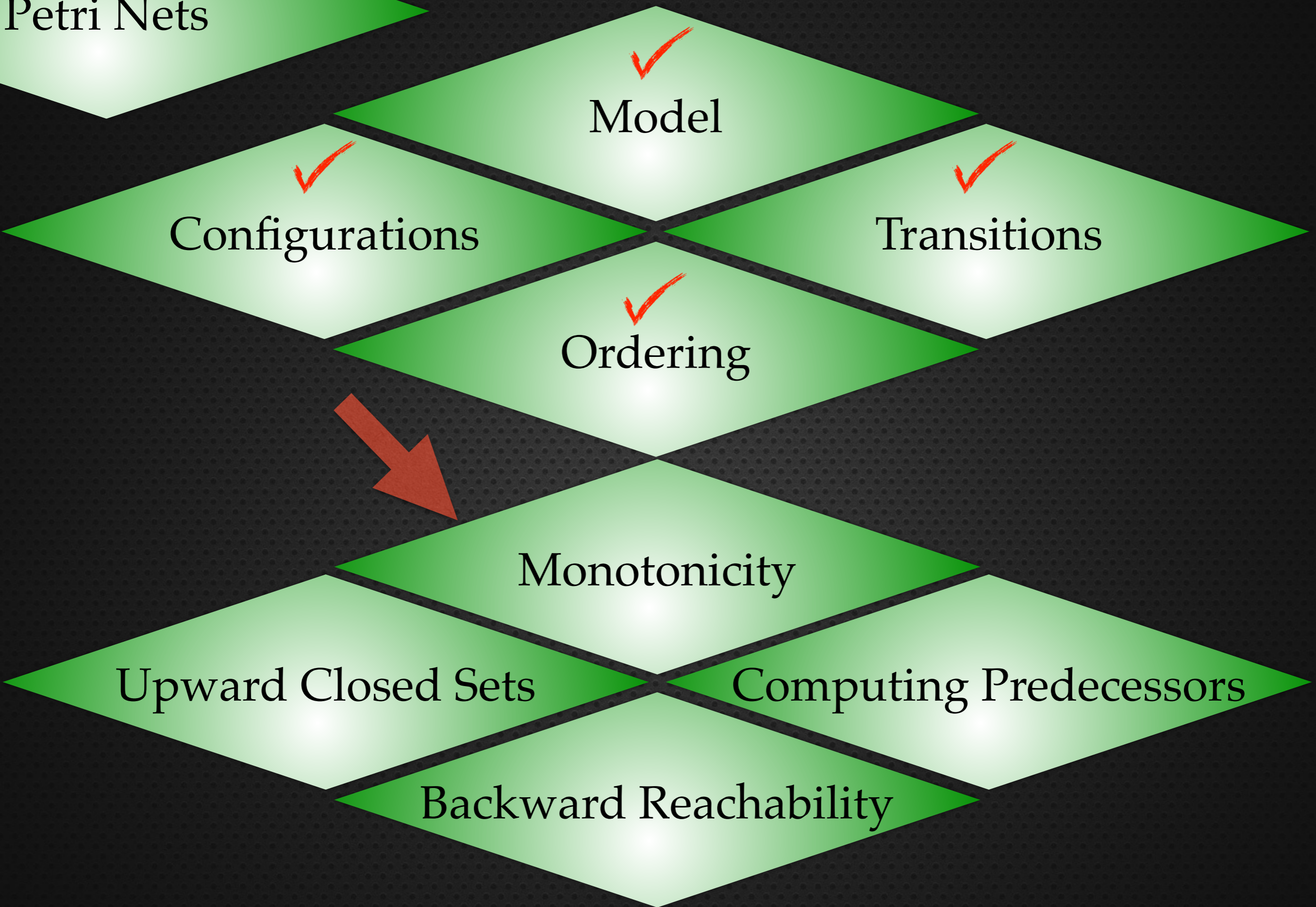
t_1



Petri Nets

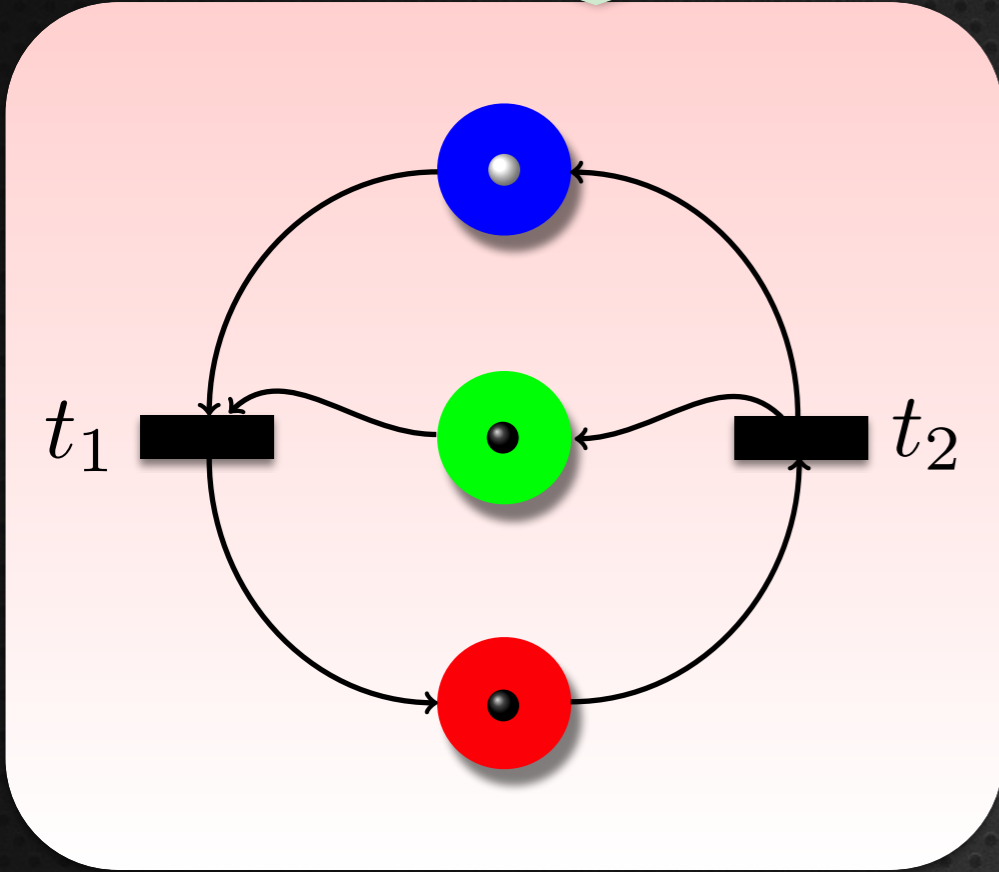


Petri Nets

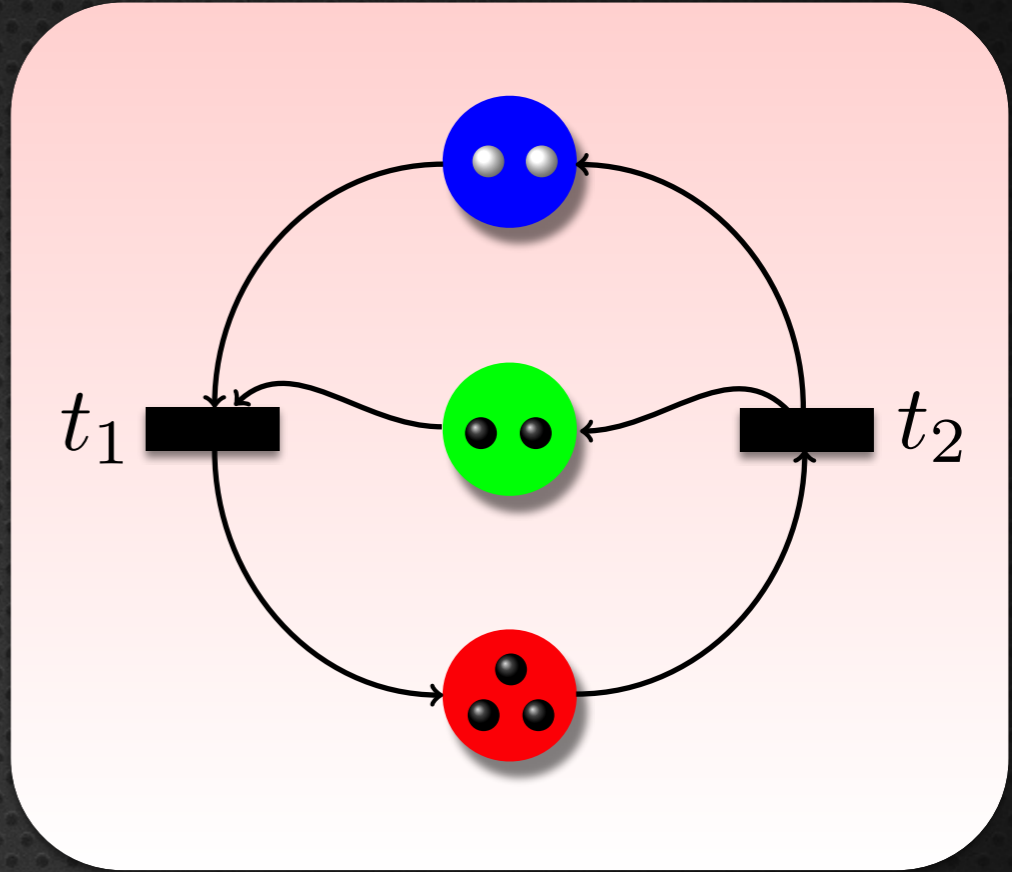


Petri Nets

Monotonicity

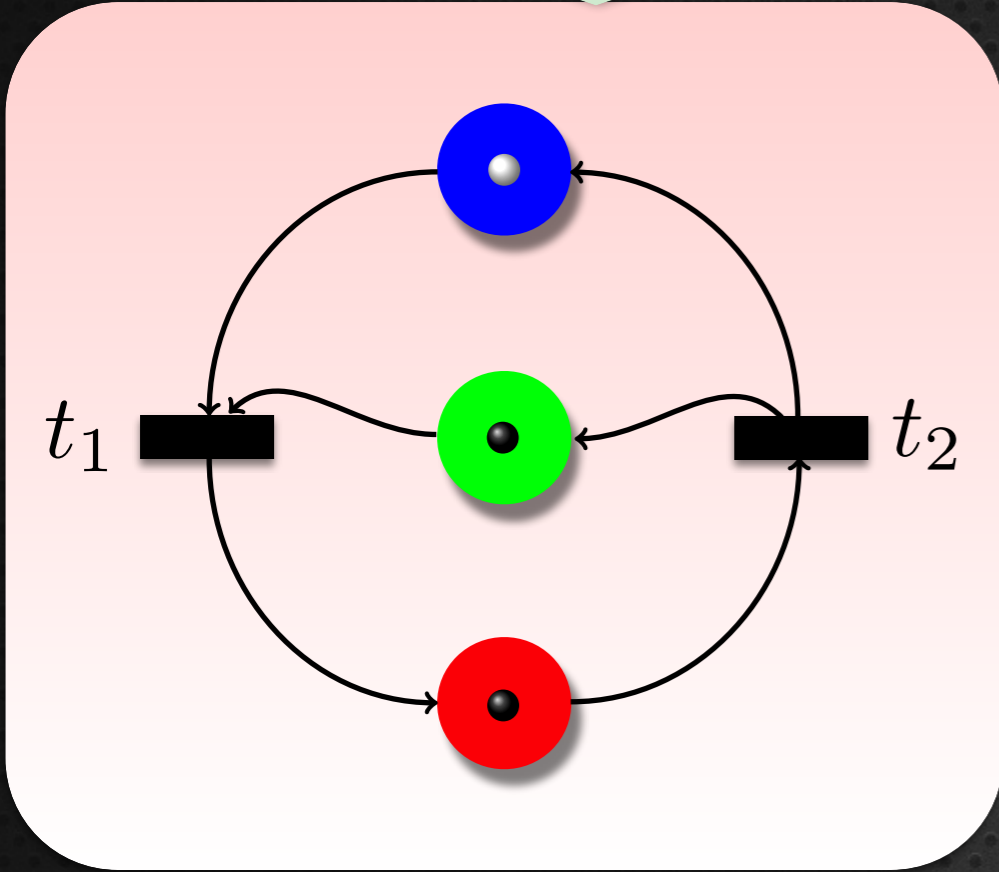


t_1

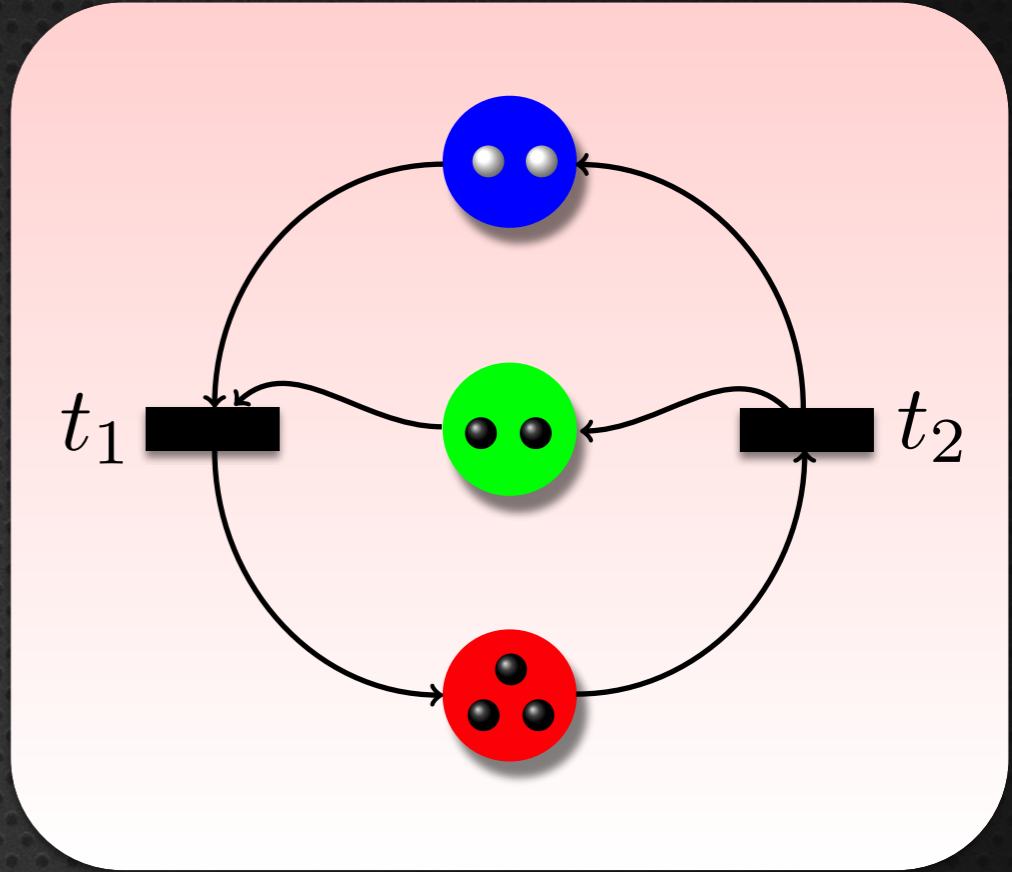


Petri Nets

Monotonicity

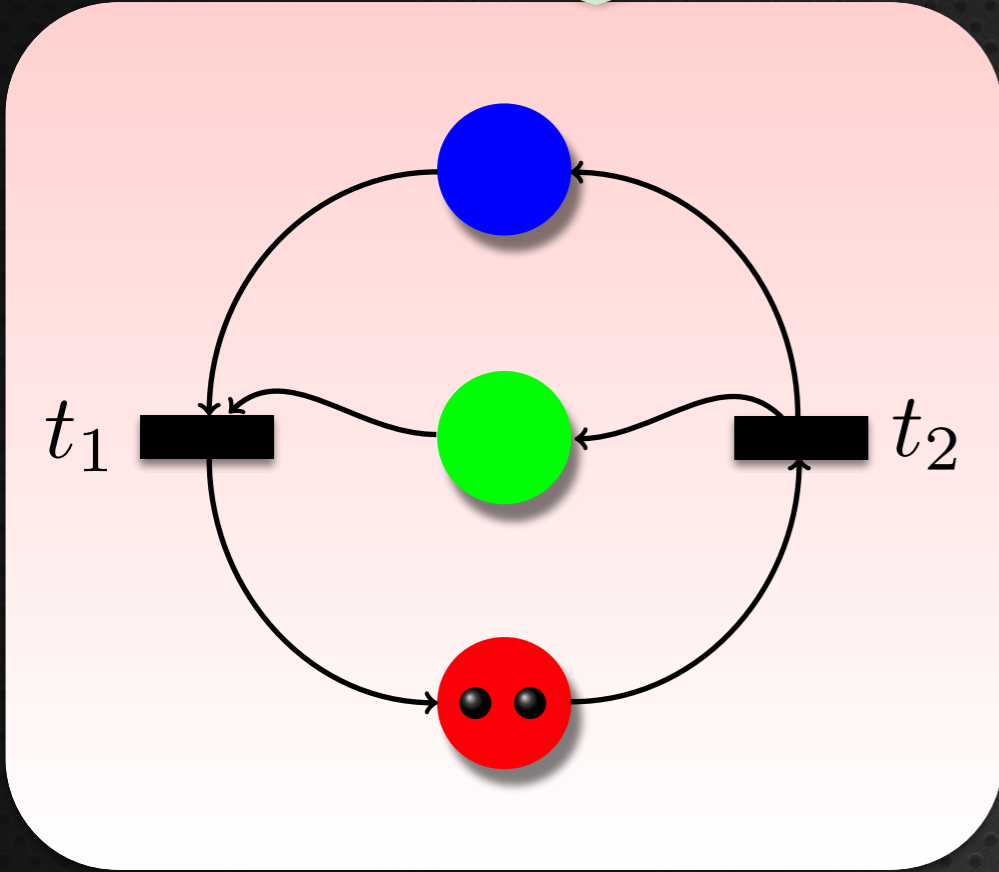


t_1

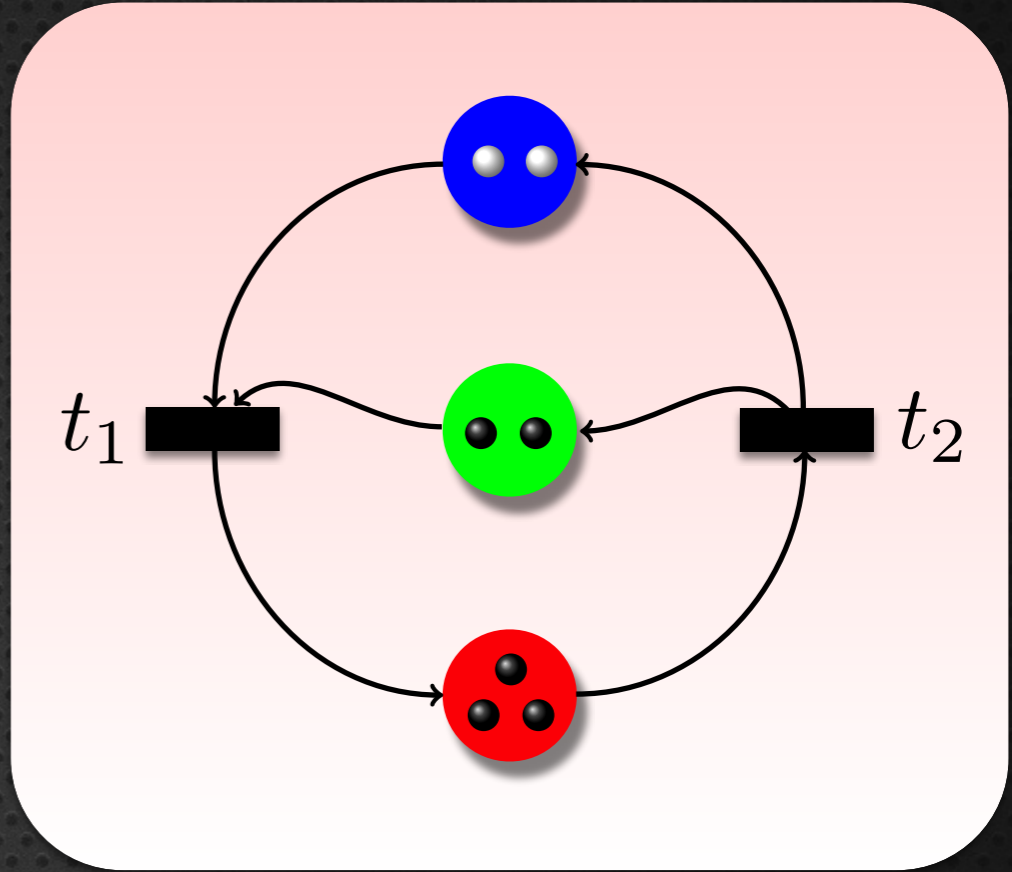


Petri Nets

Monotonicity

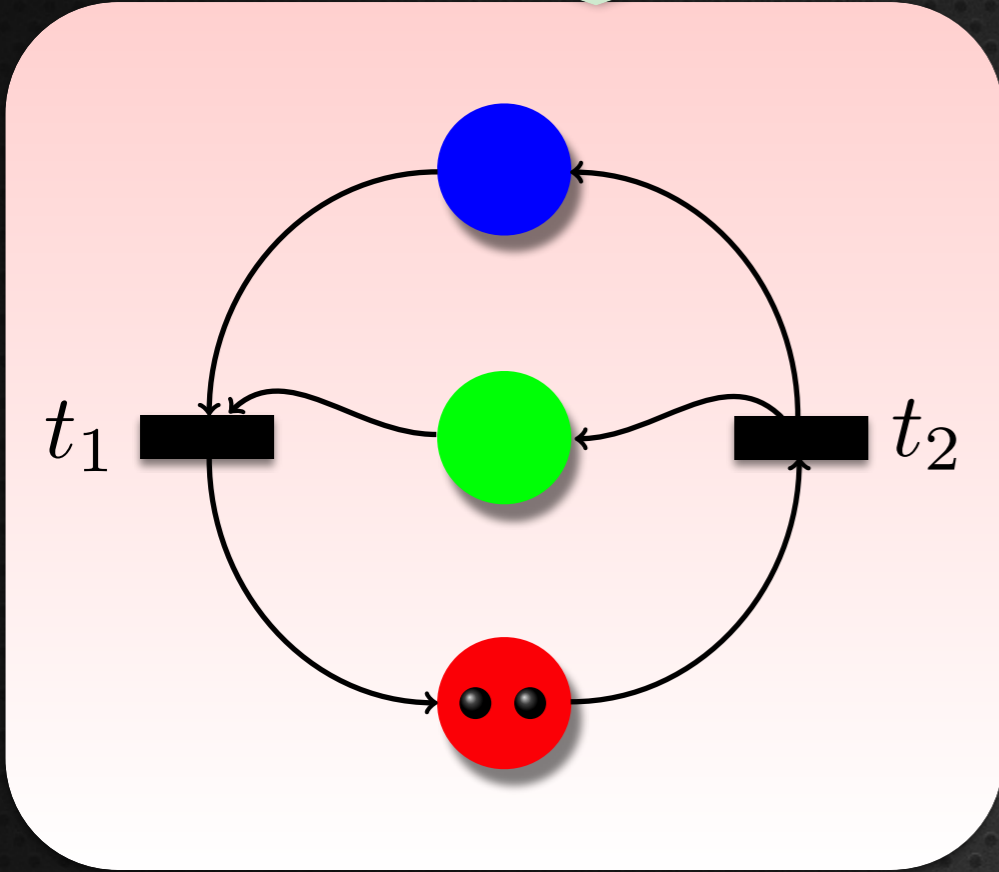


t_1

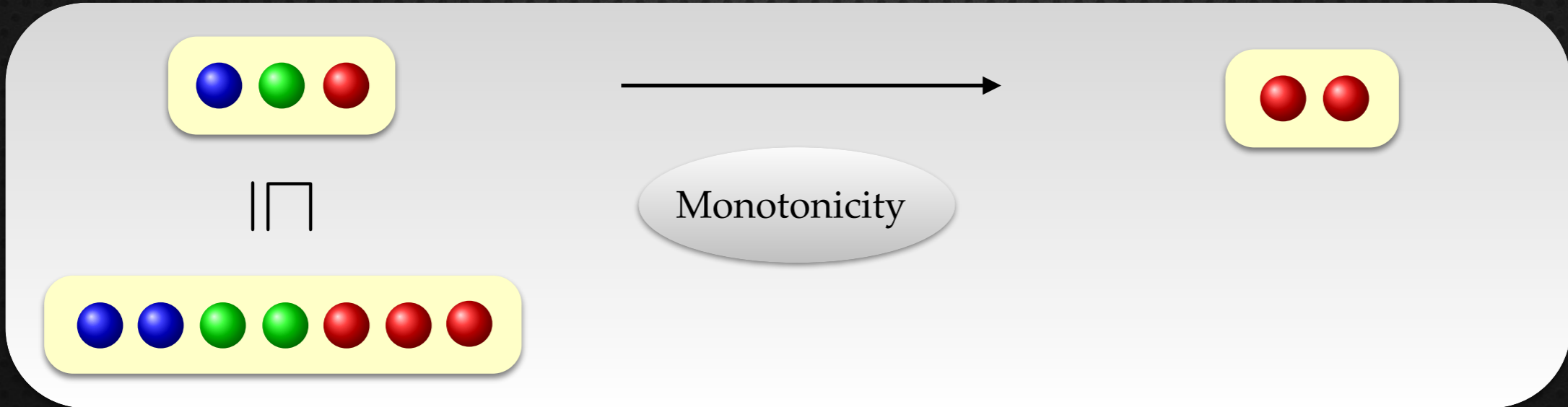
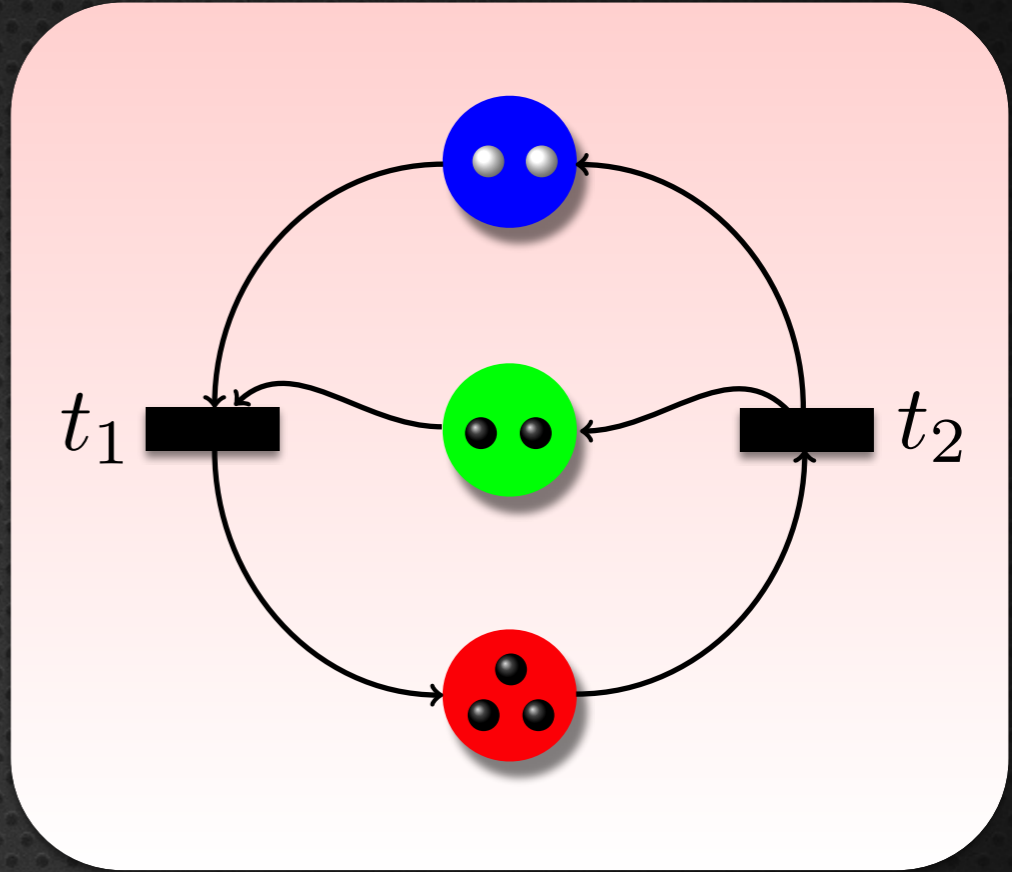


Petri Nets

Monotonicity

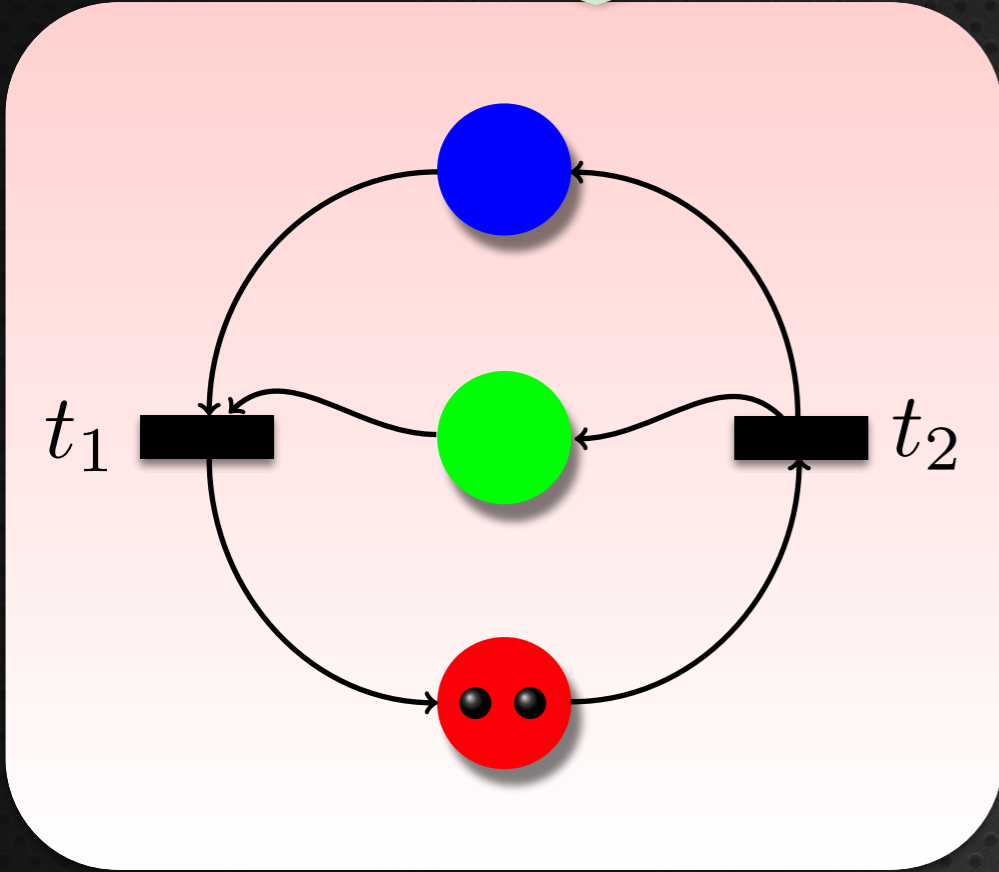


t_1

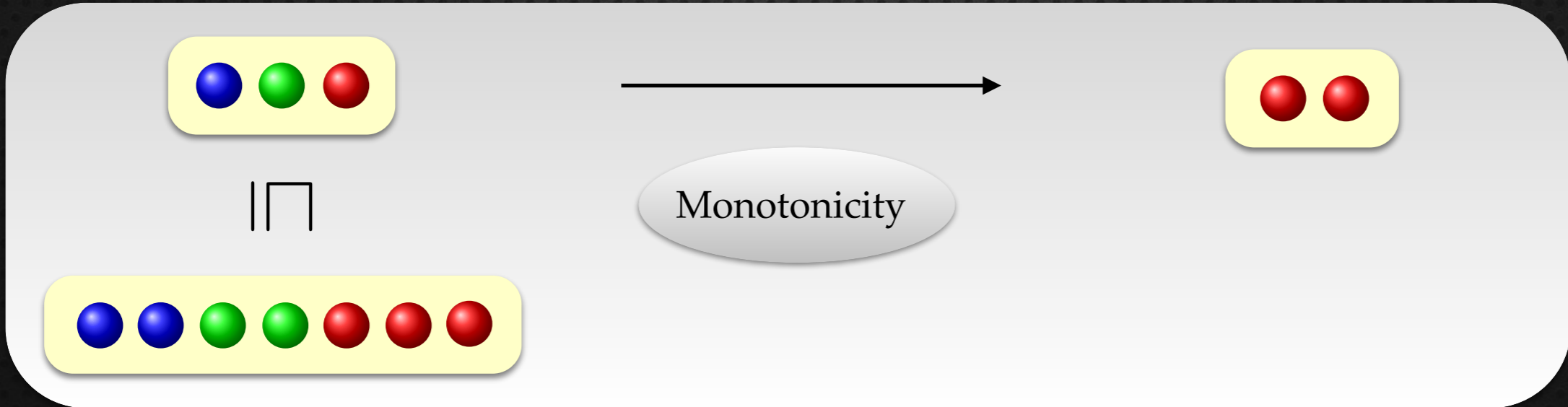
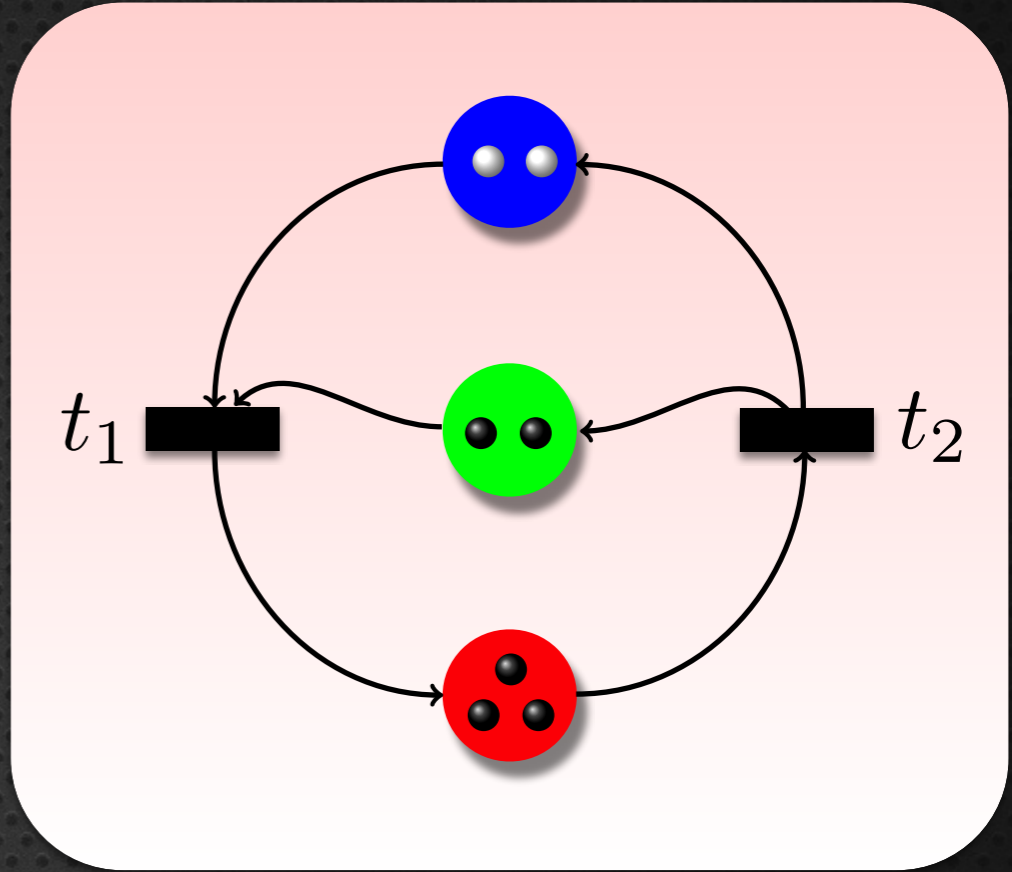


Petri Nets

Monotonicity

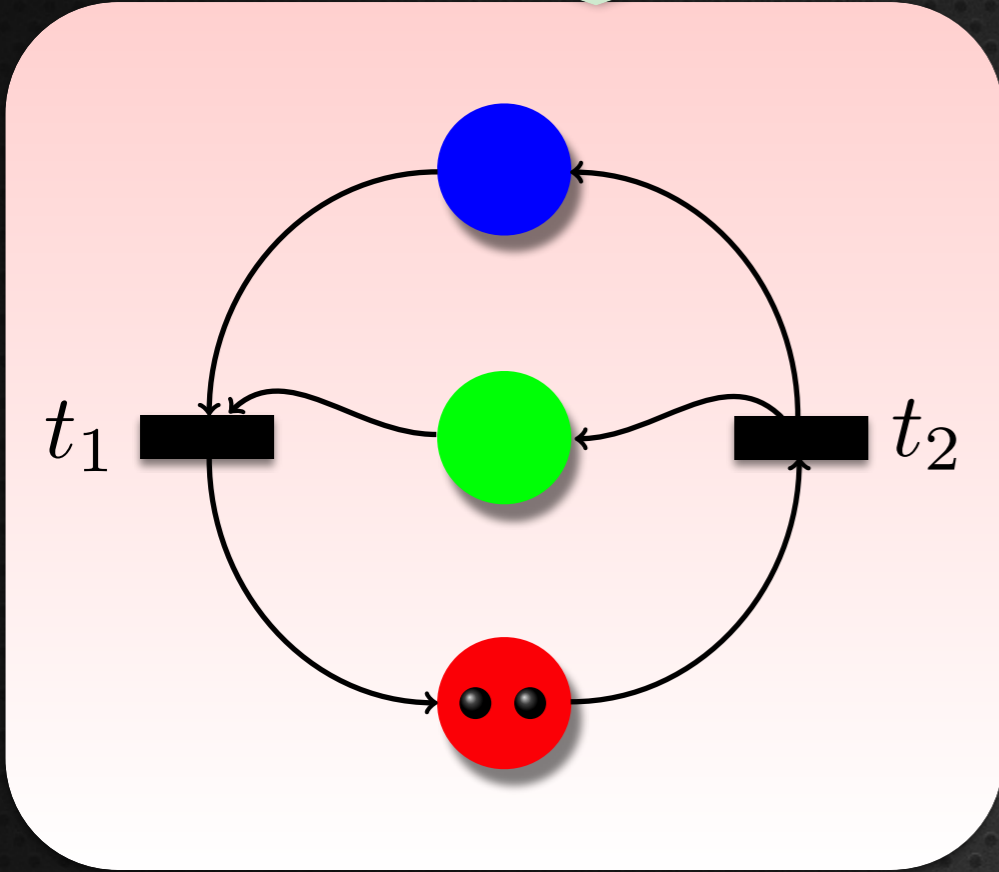


t_1

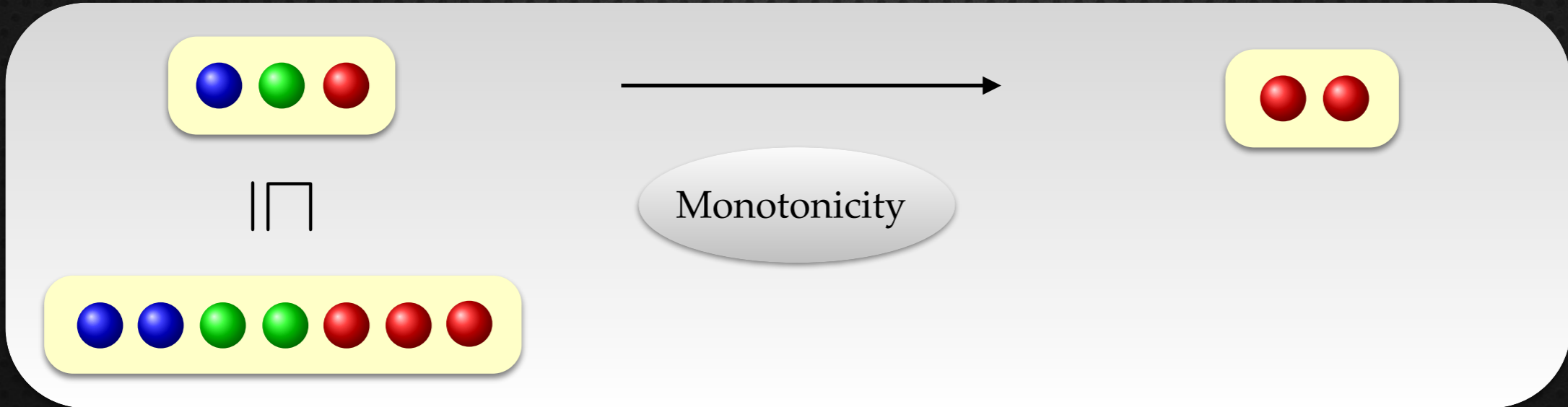
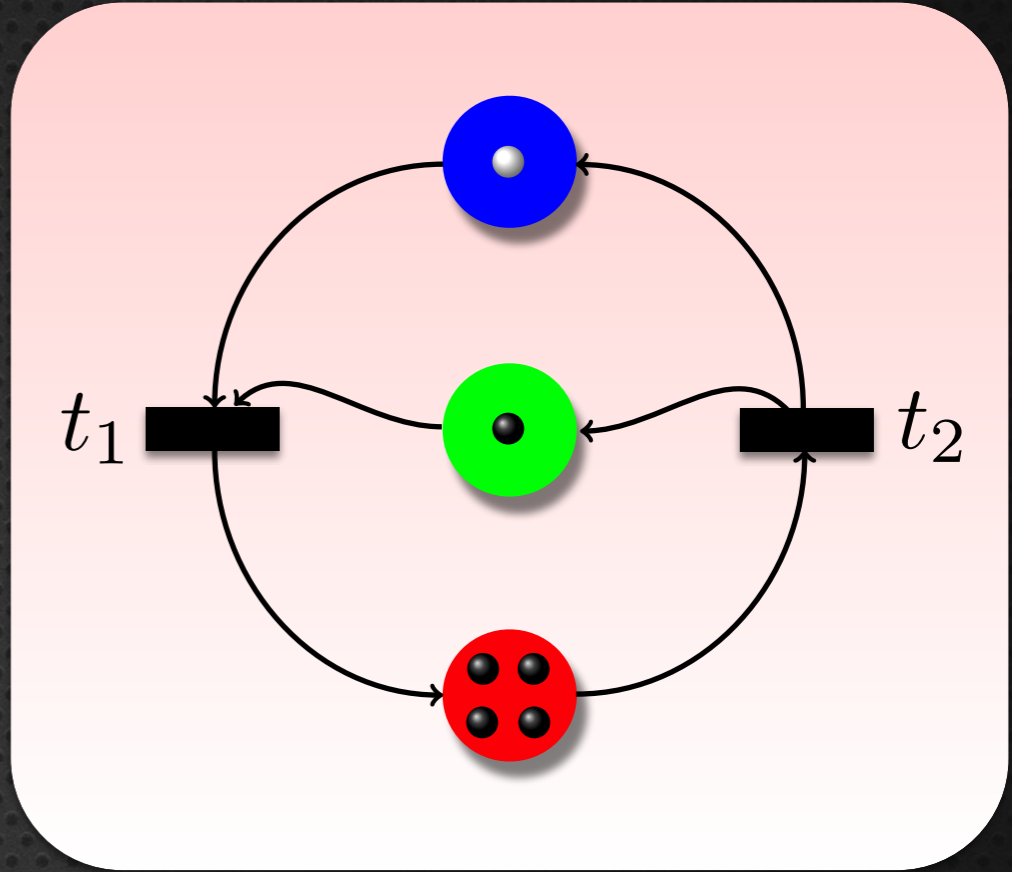


Petri Nets

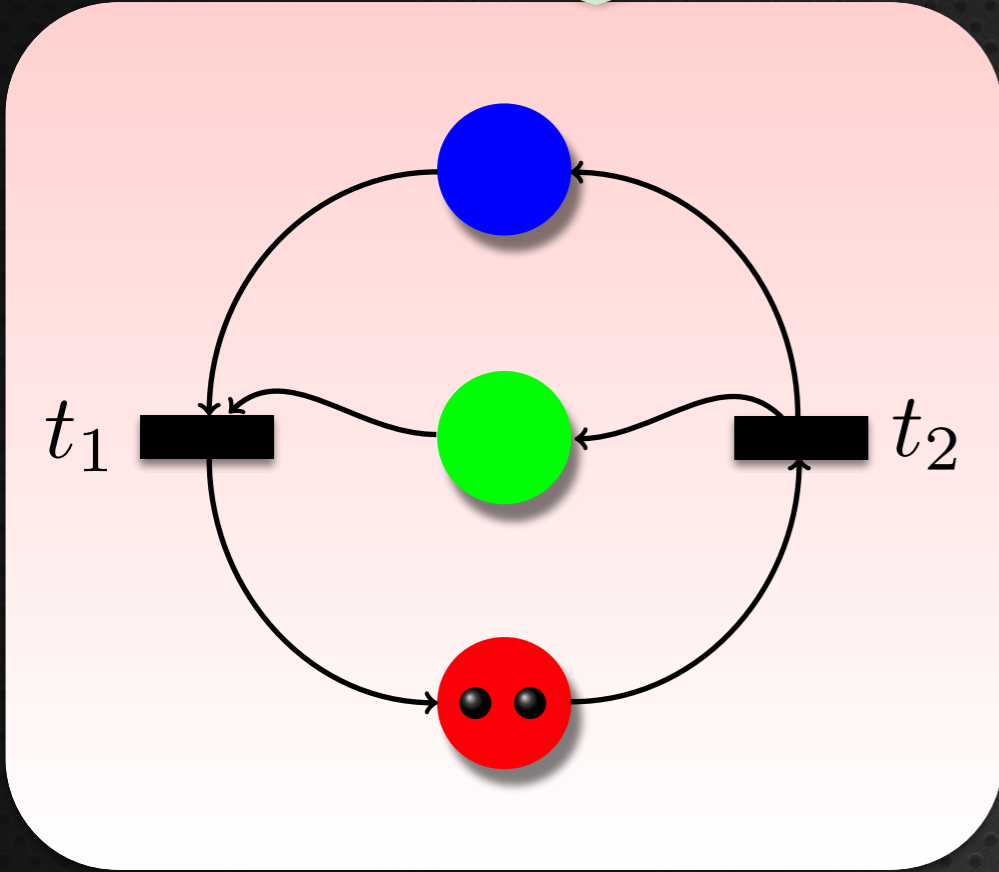
Monotonicity



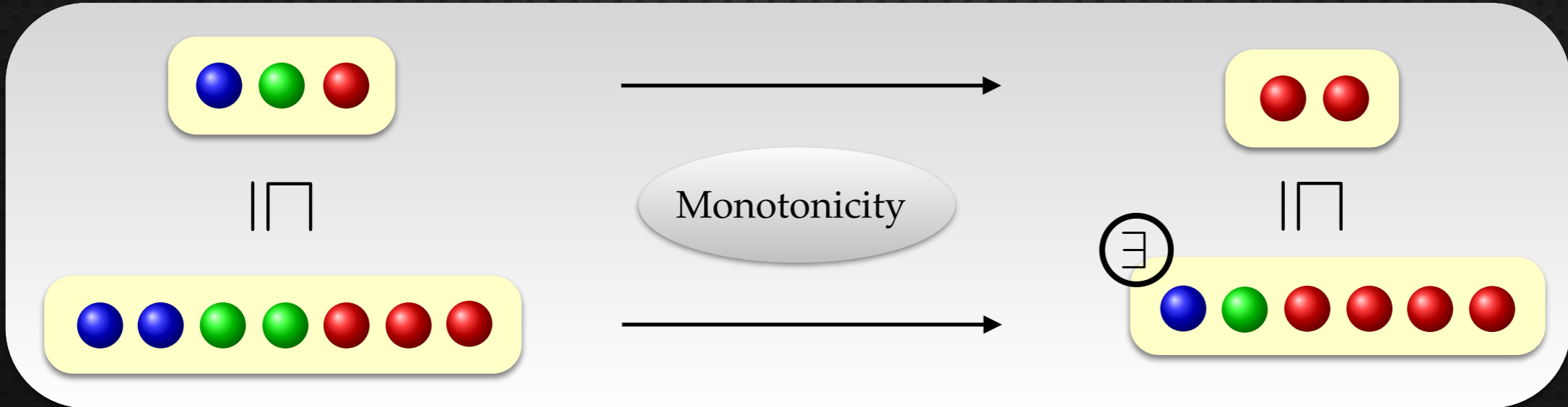
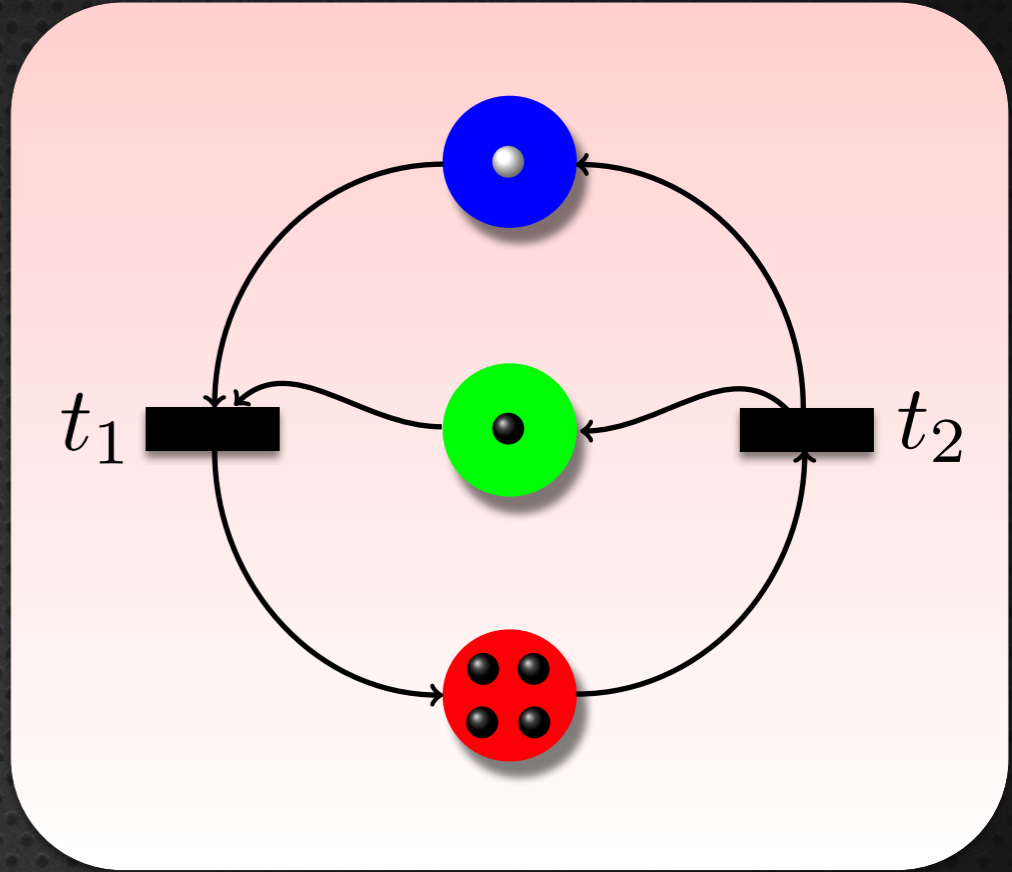
t_1



Petri Nets Monotonicity

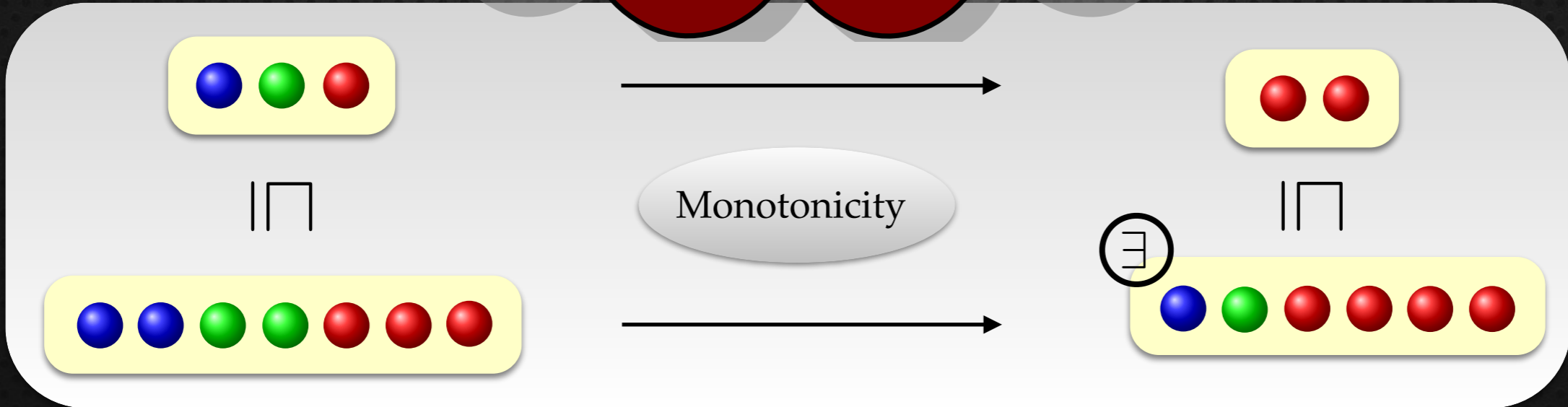
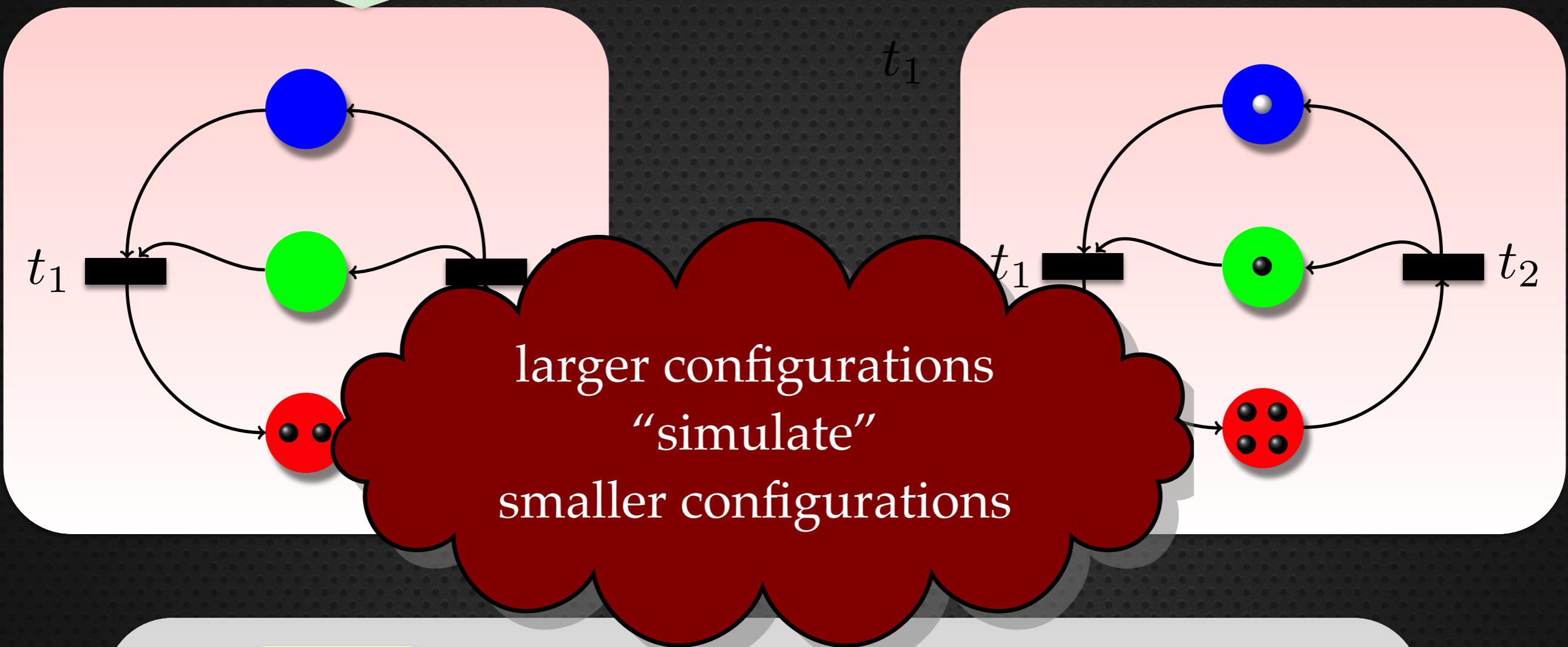


t_1

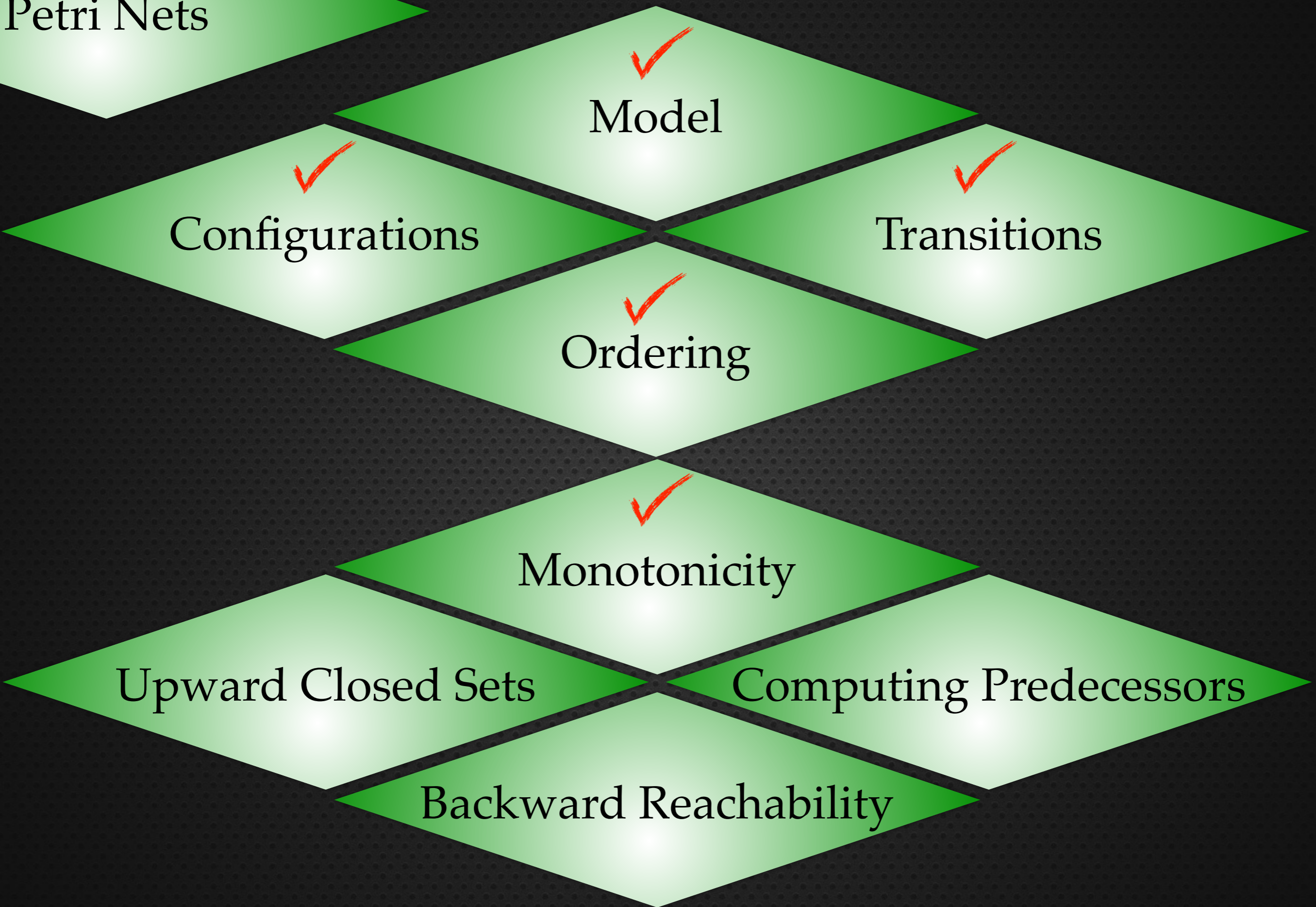


Petri Nets

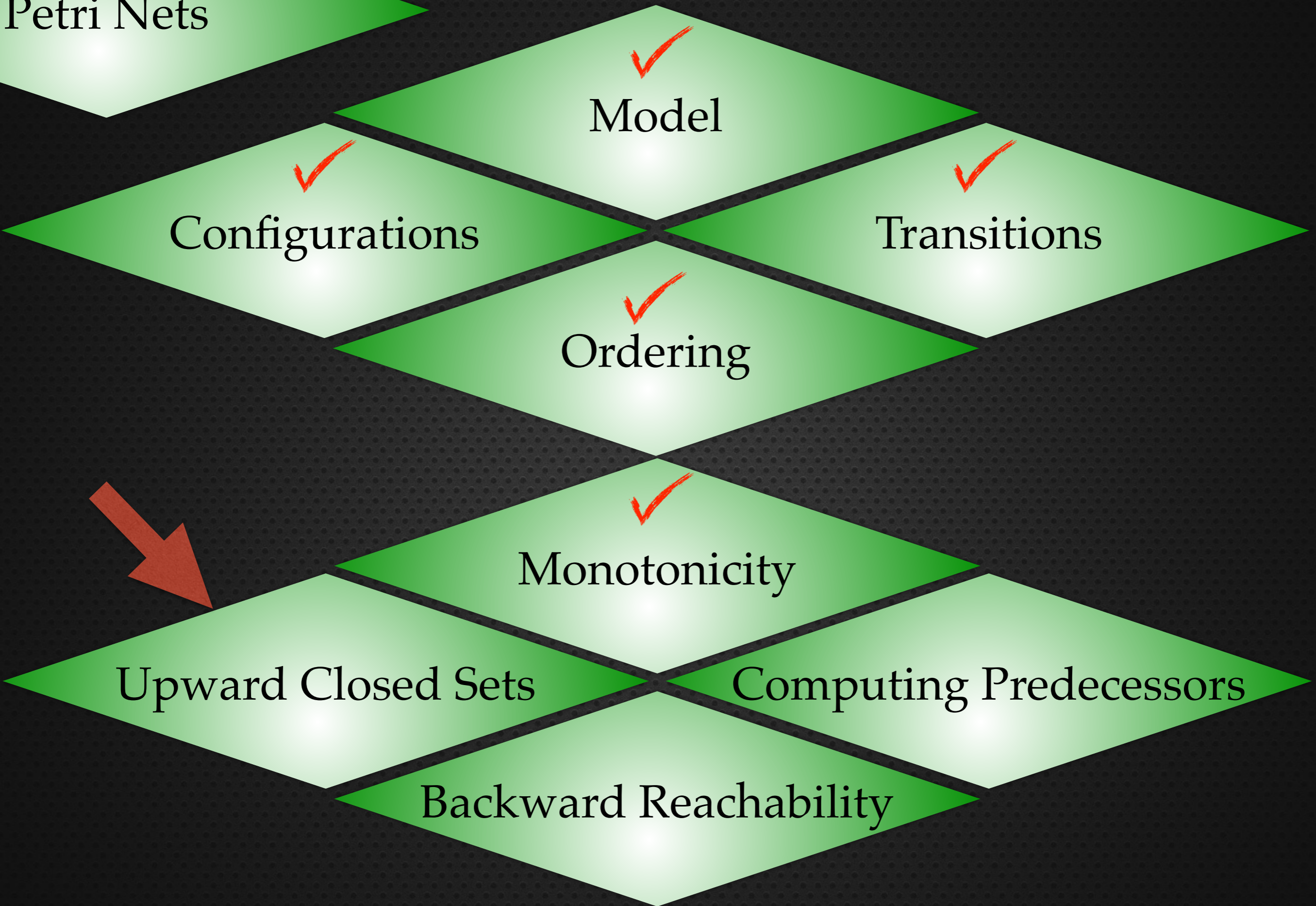
Monotonicity



Petri Nets



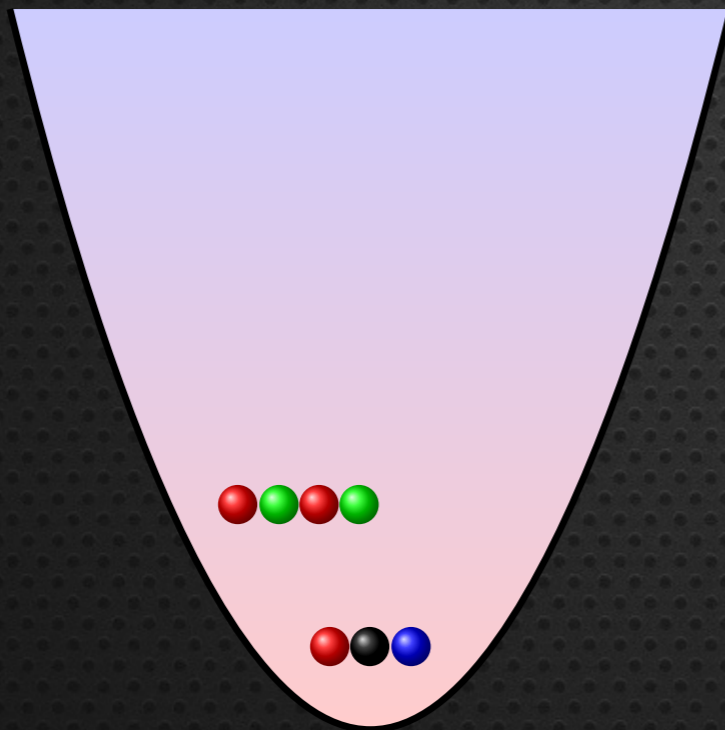
Petri Nets



Petri

Upward Closed Sets

Upward-Closed Set



Upward Closed Set (UC)

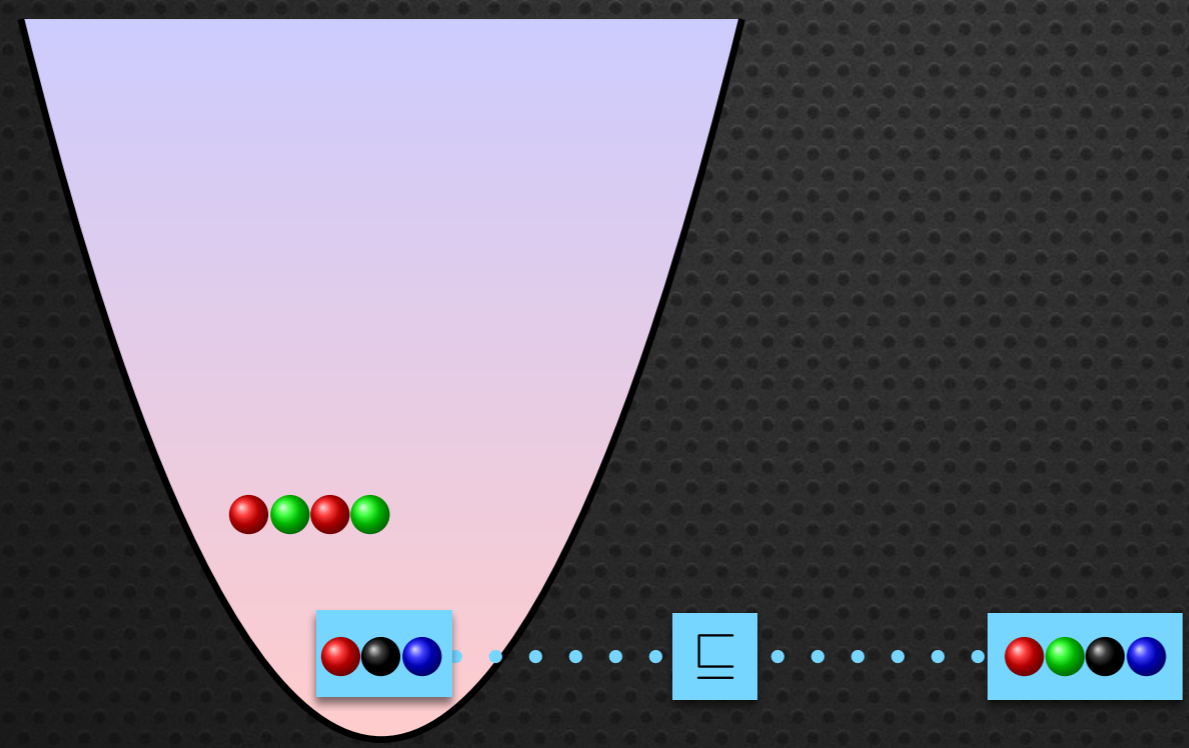
- if $m_1 \in U$ and $m_1 \sqsubseteq m_2$
- then $m_2 \in U$

Upward Closed Sets

Upward-Closed Set

Upward Closed Set (UC)

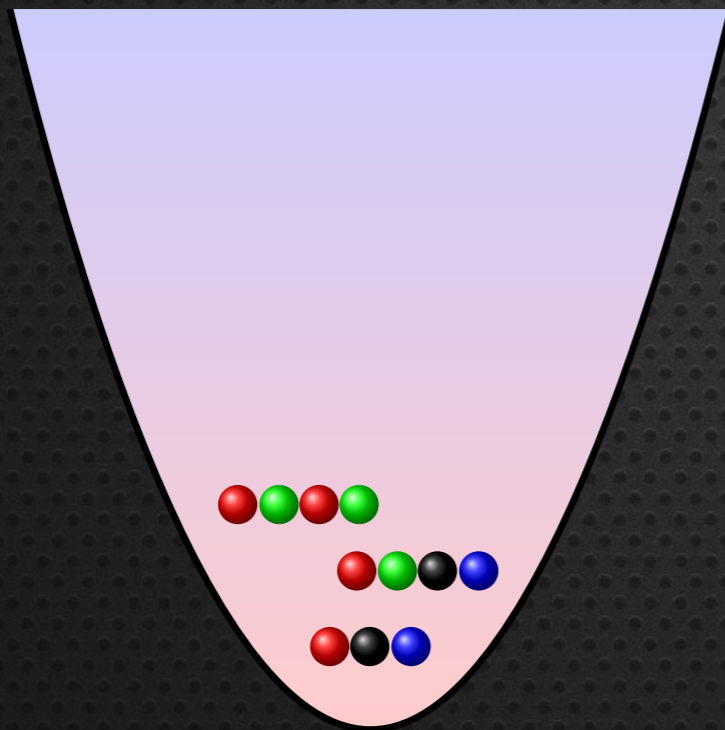
- if $m_1 \in U$ and $m_1 \sqsubseteq m_2$
- then $m_2 \in U$



Petri

Upward Closed Sets

Upward-Closed Set



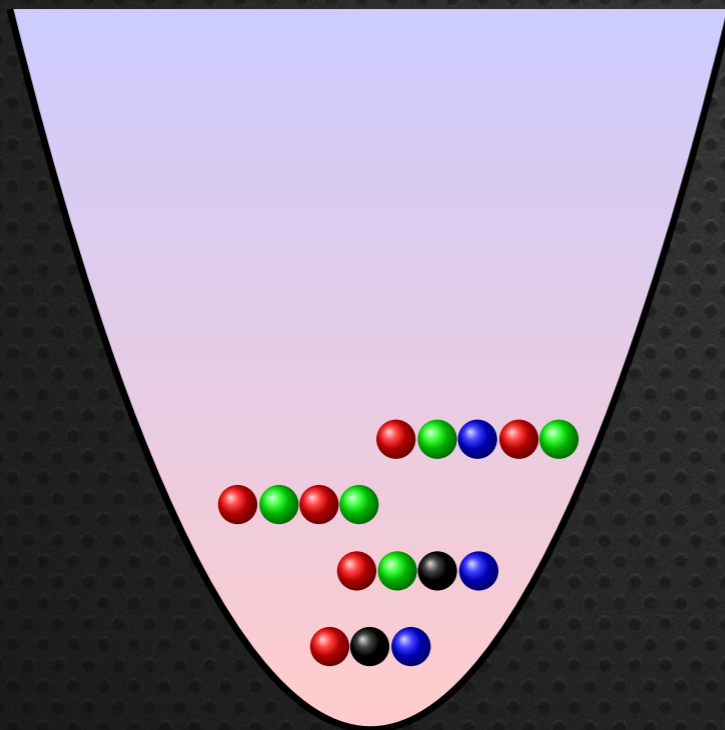
Upward Closed Set (UC)

- if $m_1 \in U$ and $m_1 \sqsubseteq m_2$
- then $m_2 \in U$

Petri

Upward Closed Sets

Upward-Closed Set



Upward Closed Set (UC)

- if $m_1 \in U$ and $m_1 \sqsubseteq m_2$
- then $m_2 \in U$

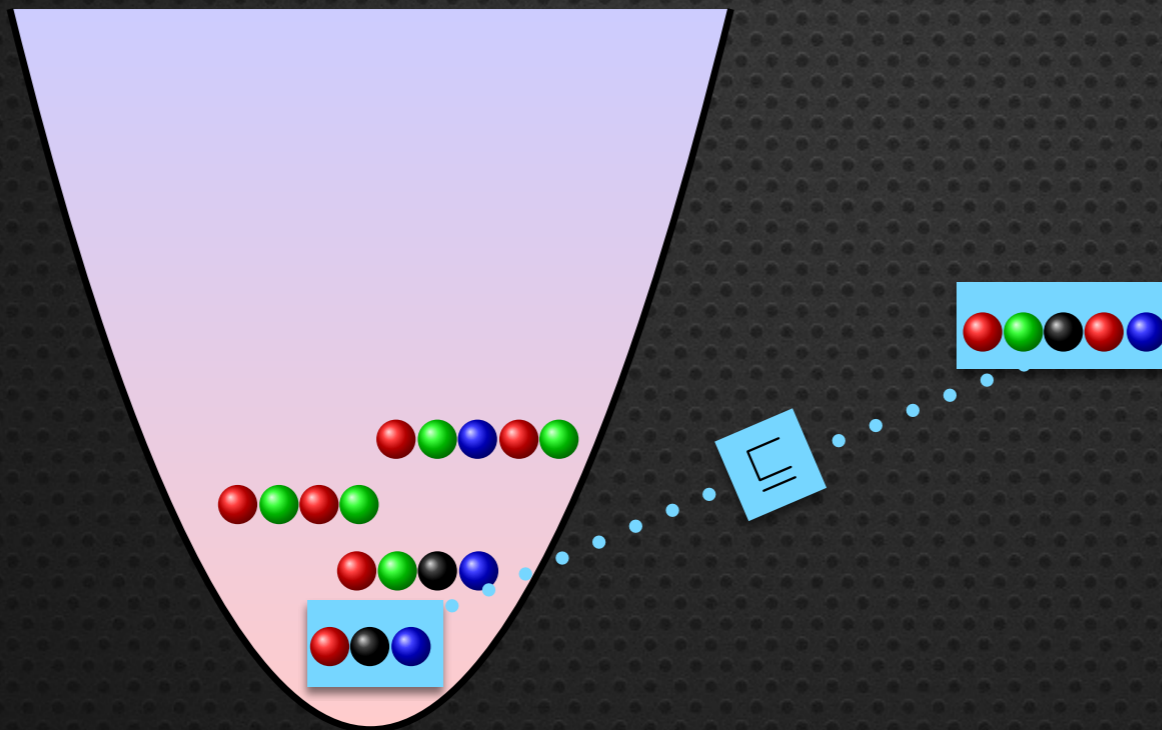
Petri

Upward Closed Sets

Upward-Closed Set

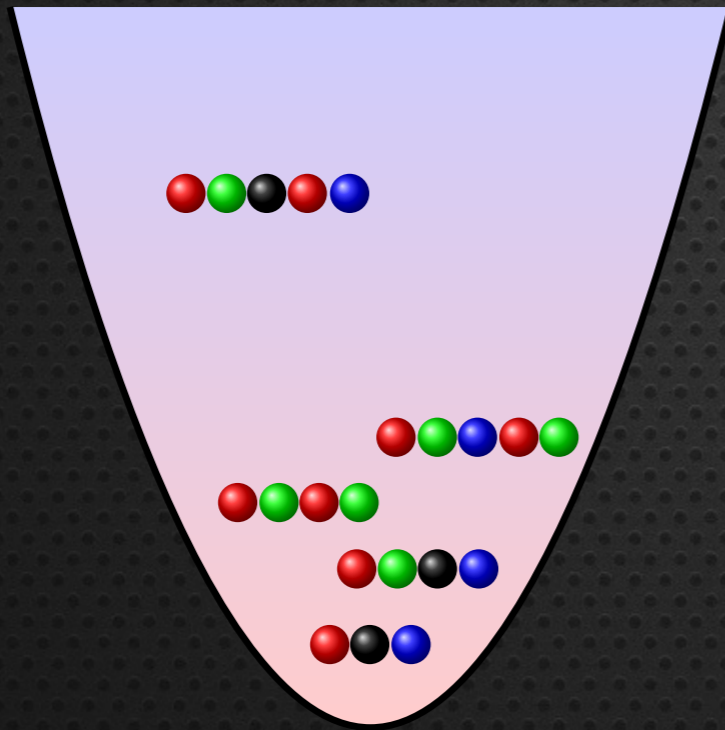
Upward Closed Set (UC)

- if $m_1 \in U$ and $m_1 \sqsubseteq m_2$
- then $m_2 \in U$



Upward Closed Sets

Upward-Closed Set



Upward Closed Set (UC)

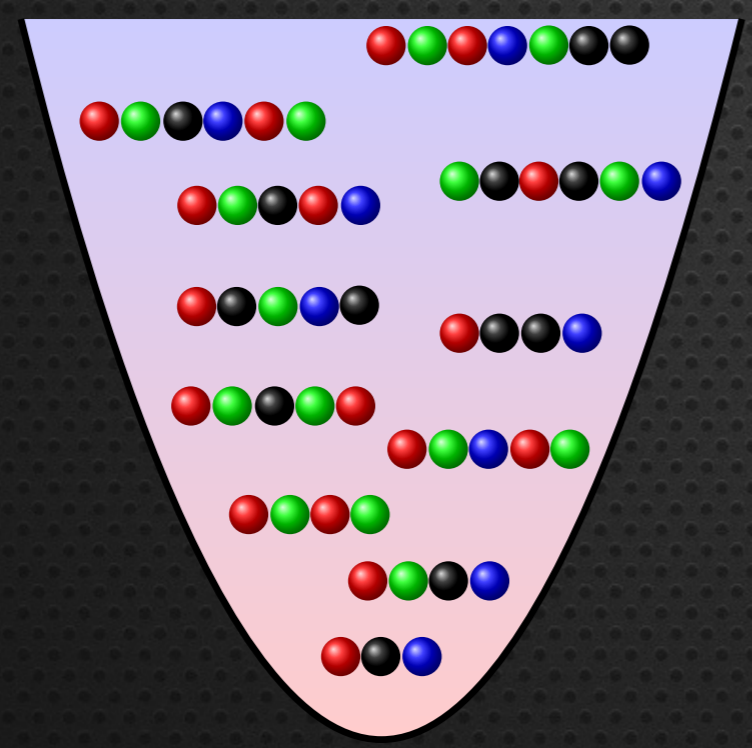
- if $m_1 \in U$ and $m_1 \sqsubseteq m_2$
- then $m_2 \in U$

Upward Closed Sets

Upward-Closed Set

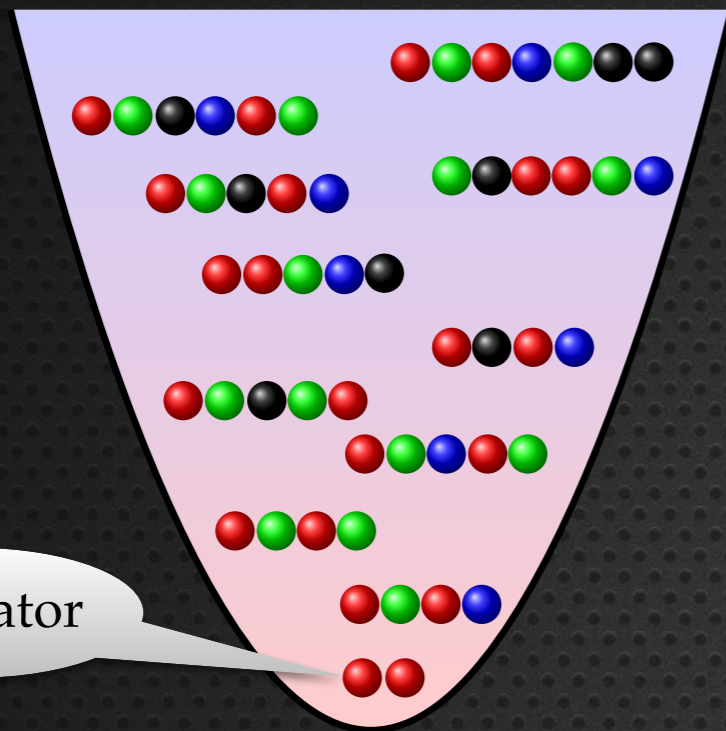
Upward Closed Set (UC)

- if $m_1 \in U$ and $m_1 \sqsubseteq m_2$
- then $m_2 \in U$



Petri Nets

Upward Closed Sets



generator

Upward Closed Set (UC)

- if $m_1 \in U$ and $m_1 \sqsubseteq m_2$
- then $m_2 \in U$

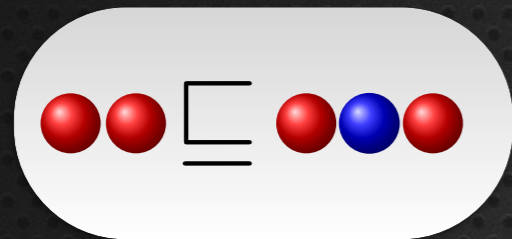
critical section



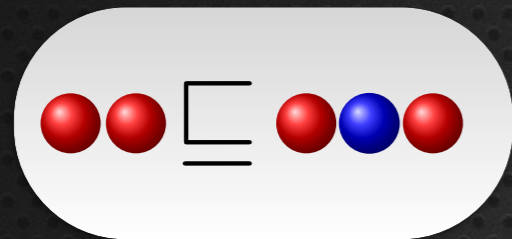
Why UC?

- Bad sets of markings are UC
 - checking safety properties = reachability of bad markings
- Uniquely characterized by generator
 - simple representation = finite multiset

Upward Closed Sets

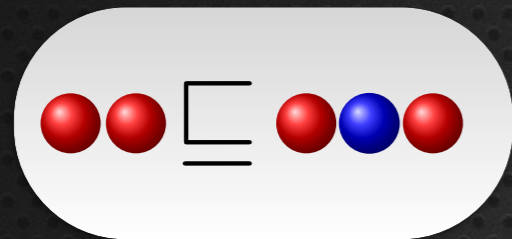


Upward Closed Sets

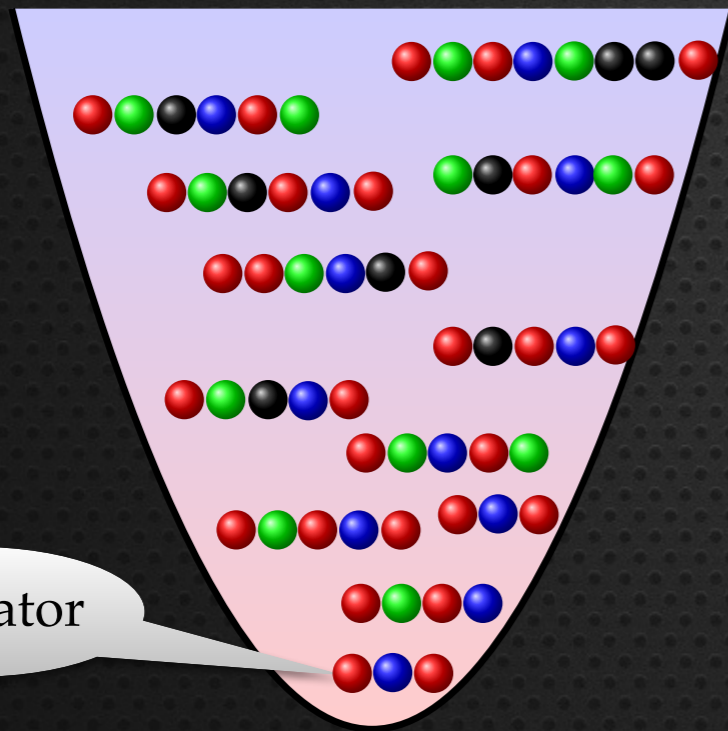


implies

Upward Closed Sets

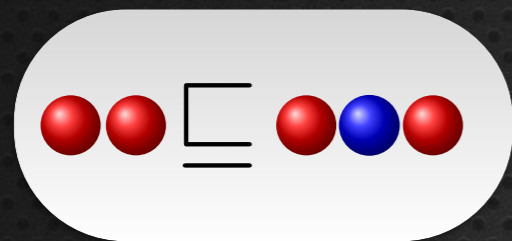


implies

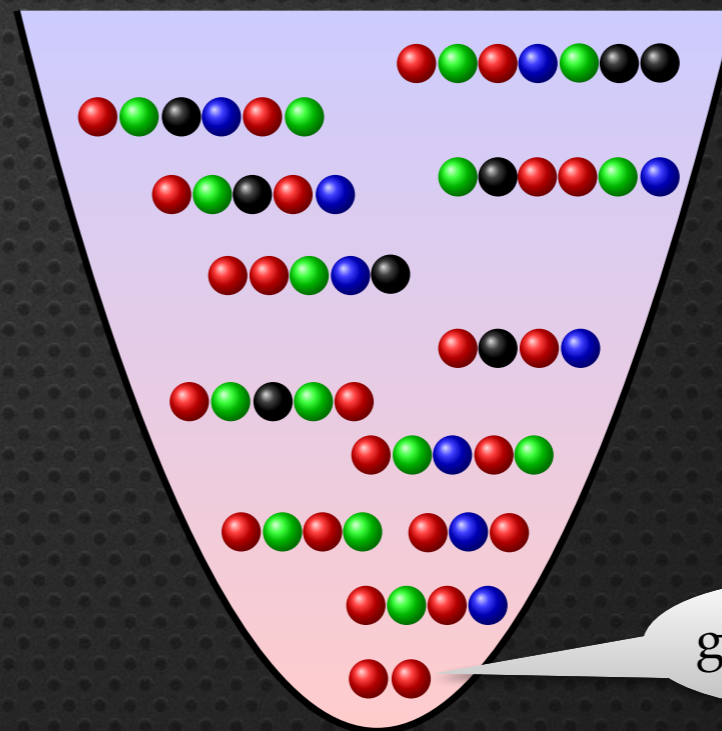
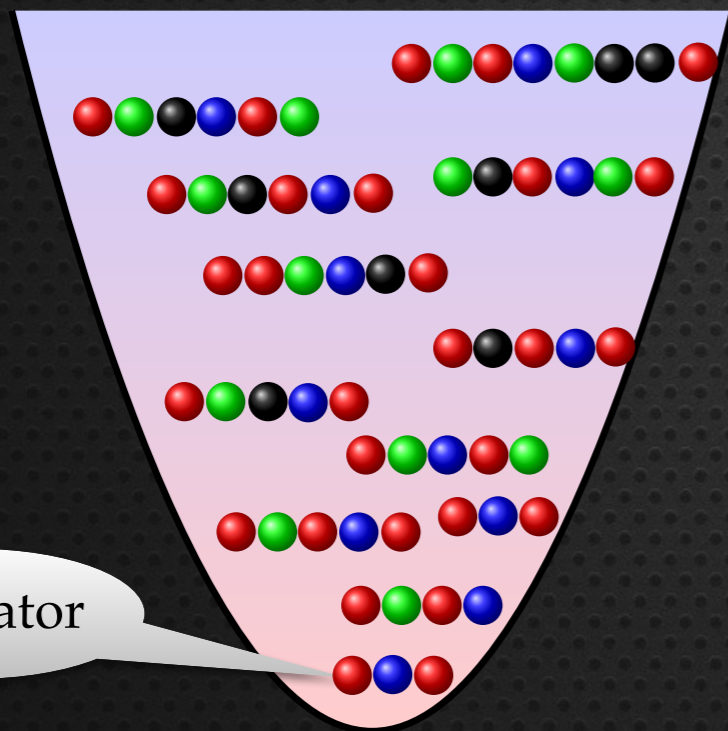


generator

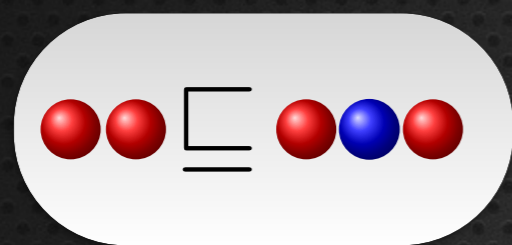
Upward Closed Sets



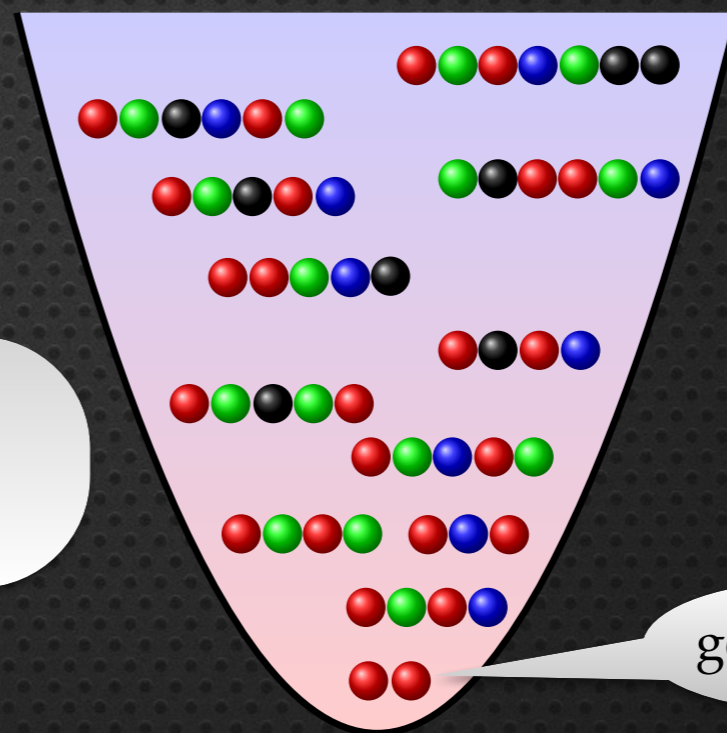
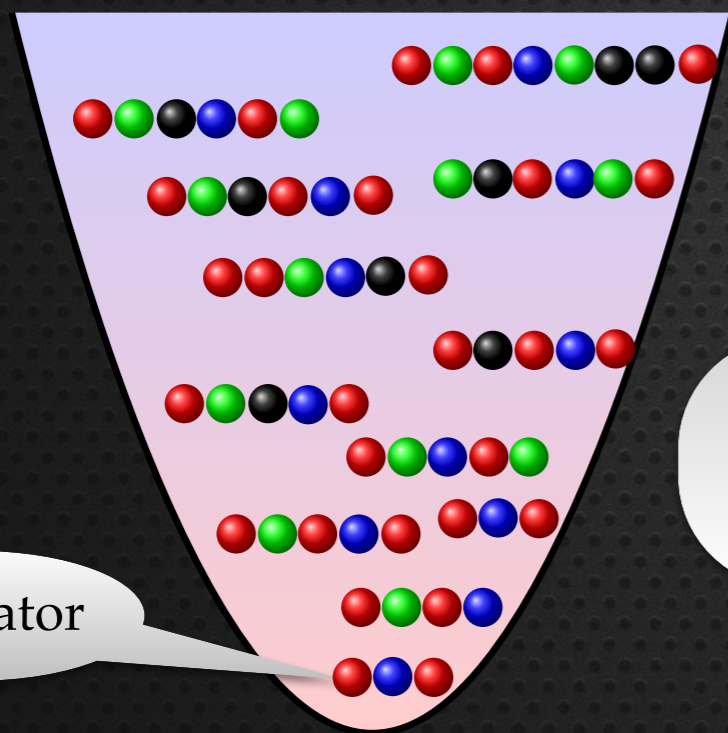
implies



Upward Closed Sets



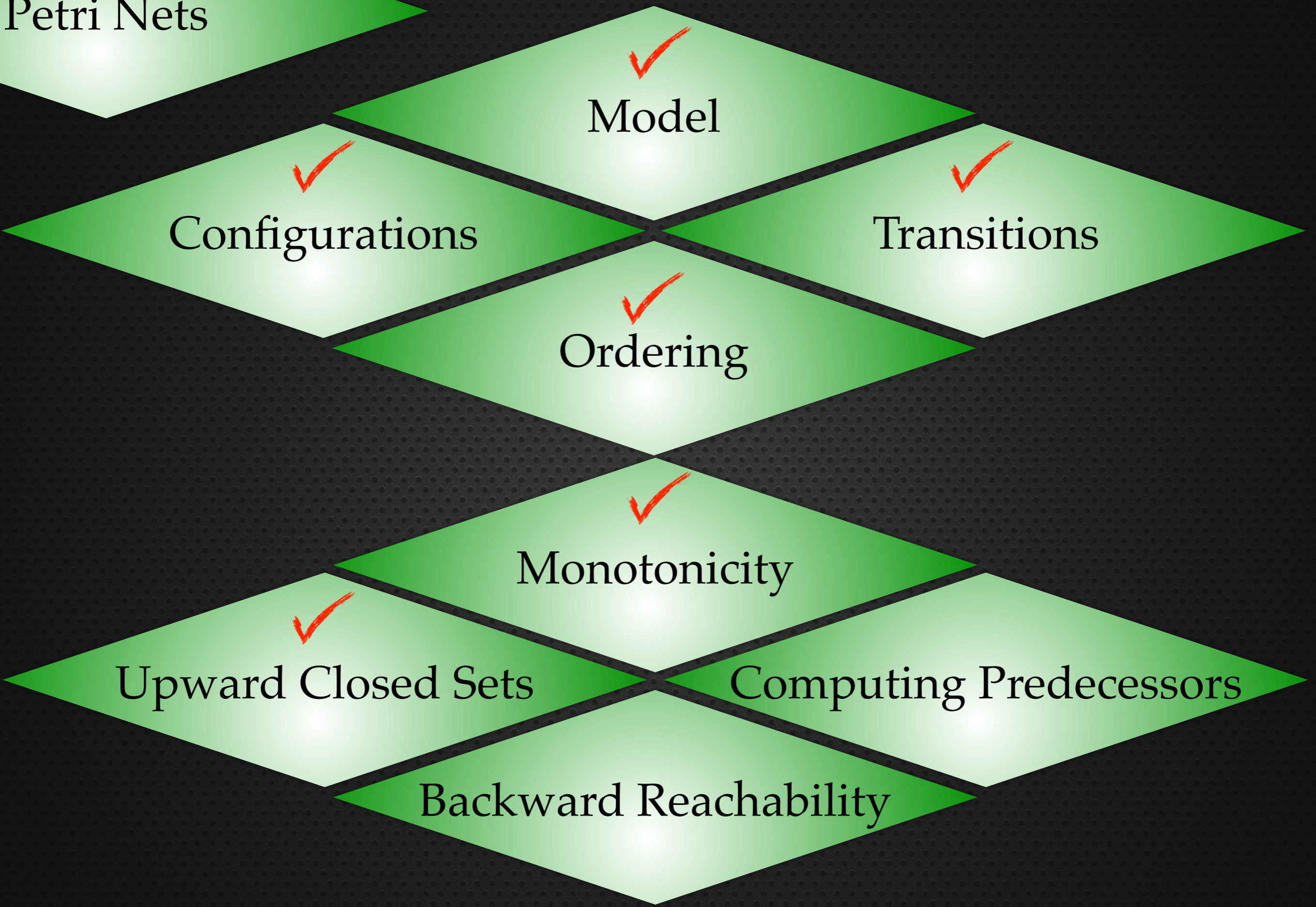
implies



generator

generator

Petri Nets



Model

Configurations

Transitions

Ordering

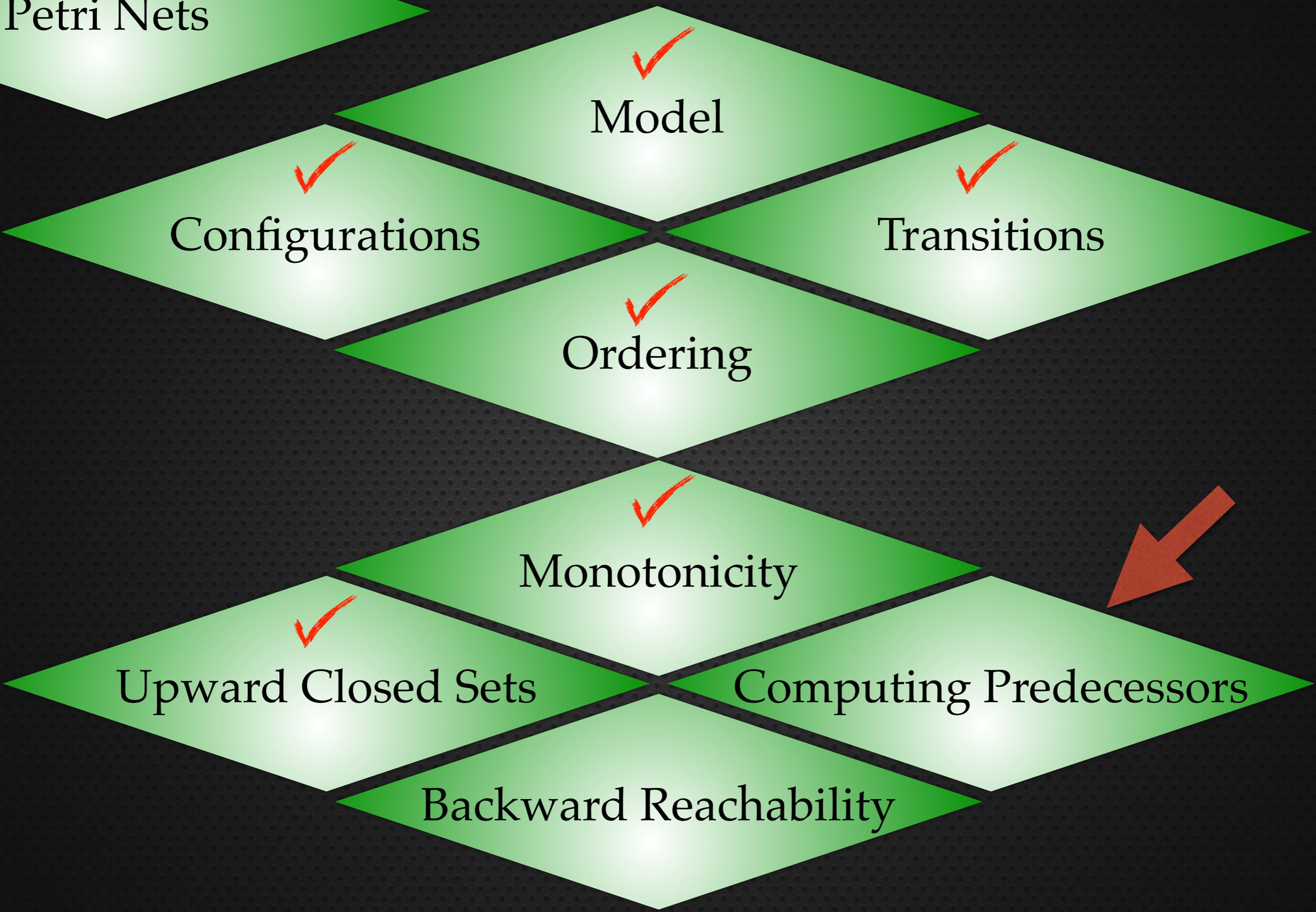
Monotonicity

Upward Closed Sets

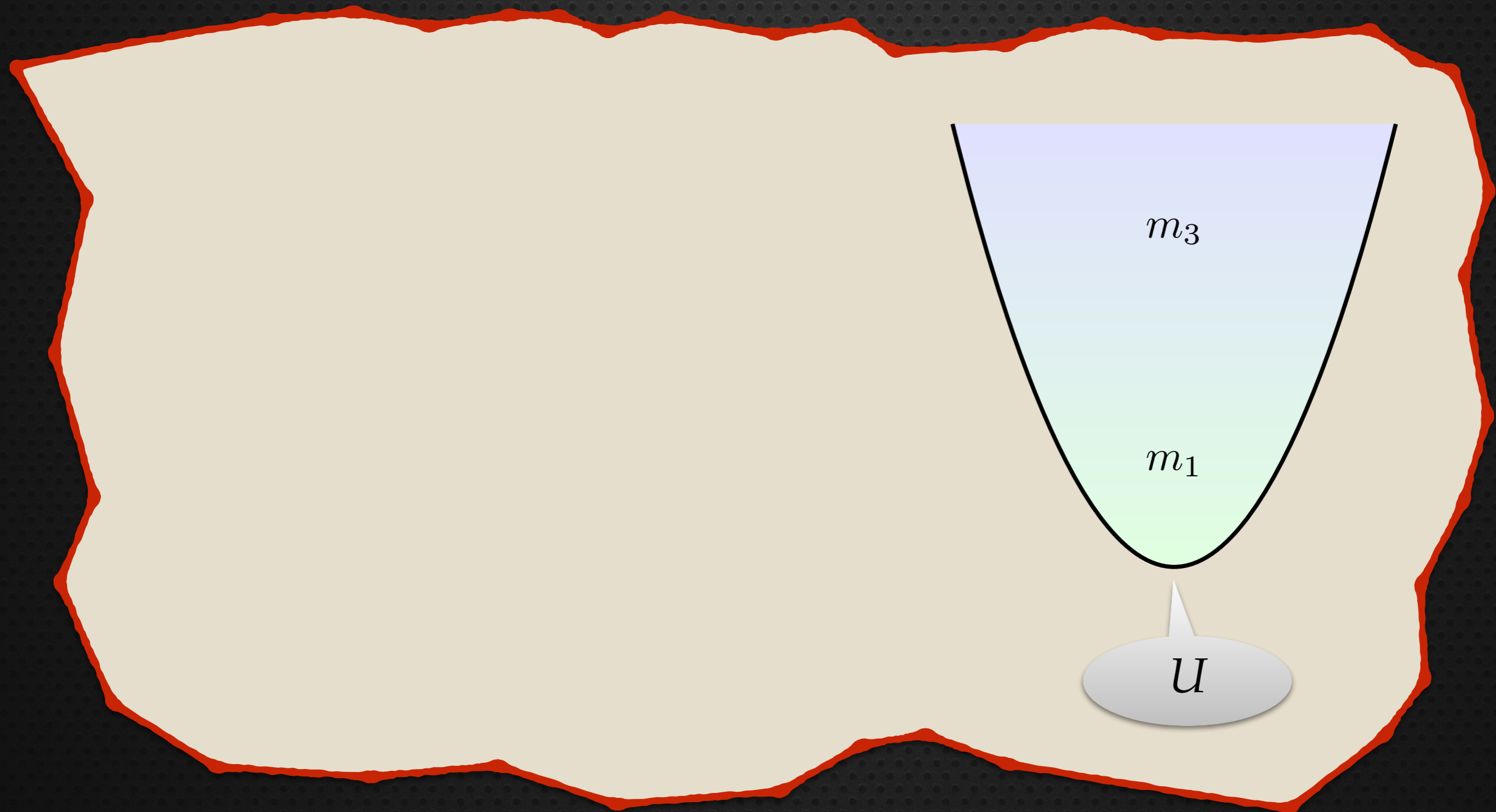
Computing Predecessors

Backward Reachability

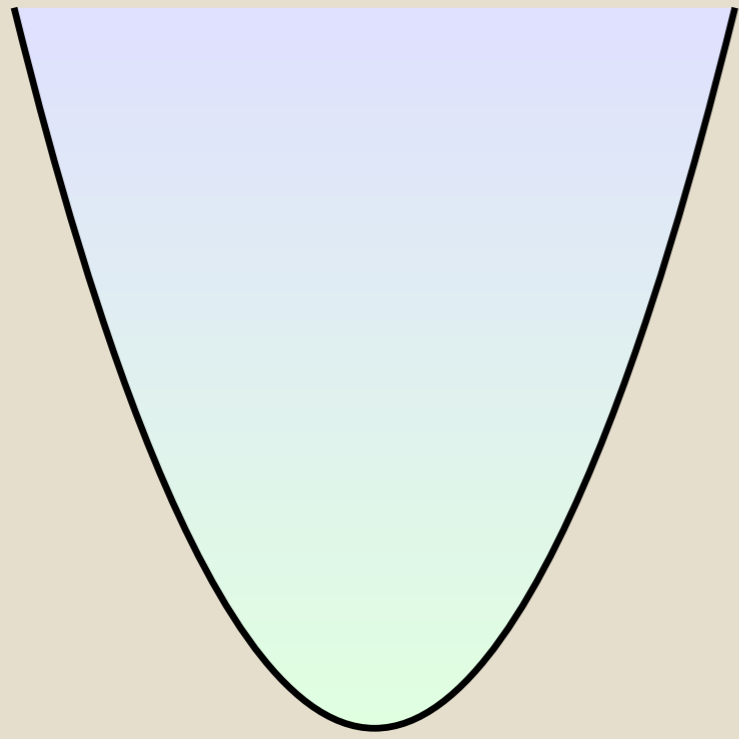
Petri Nets



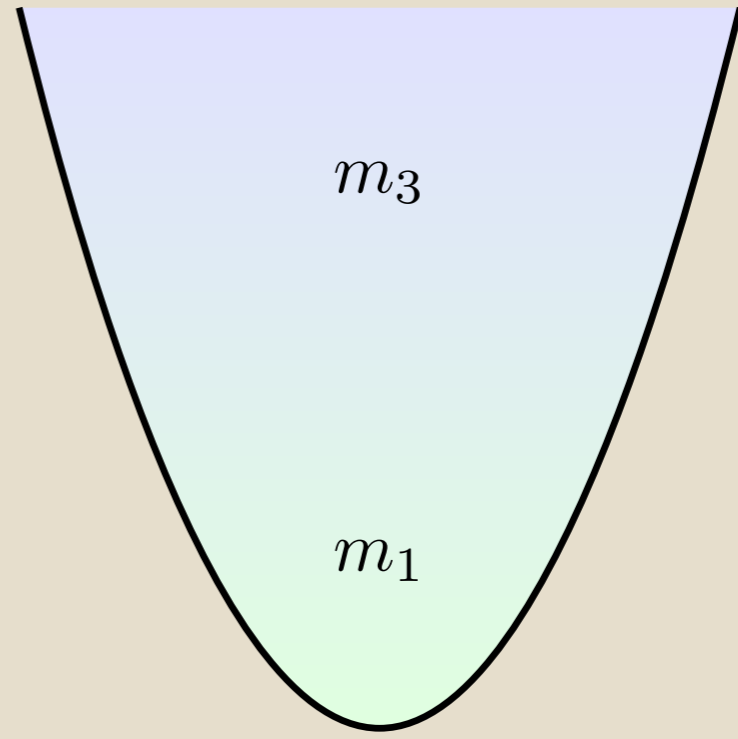
Predecessors



Predecessors

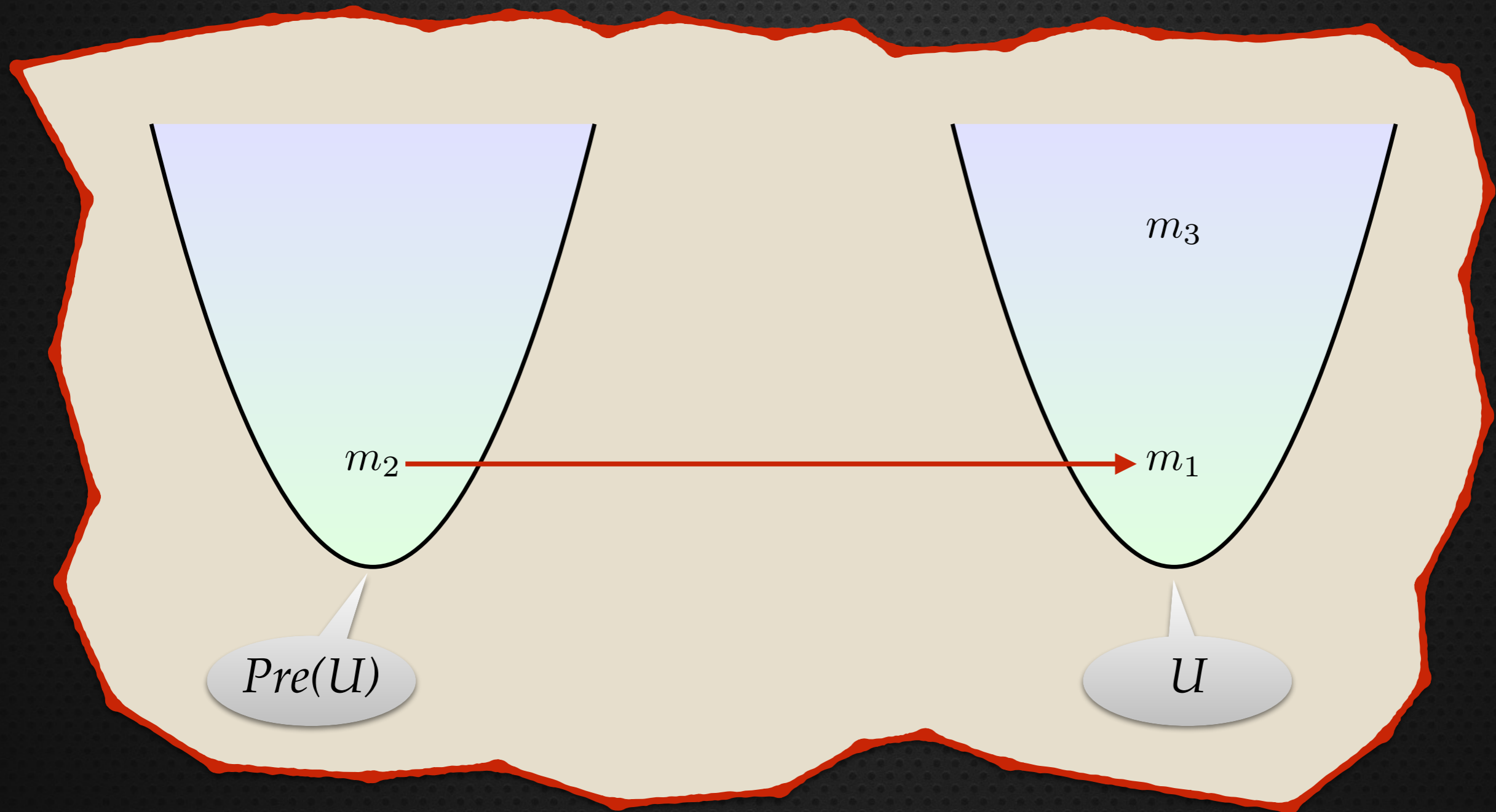


$Pre(U)$

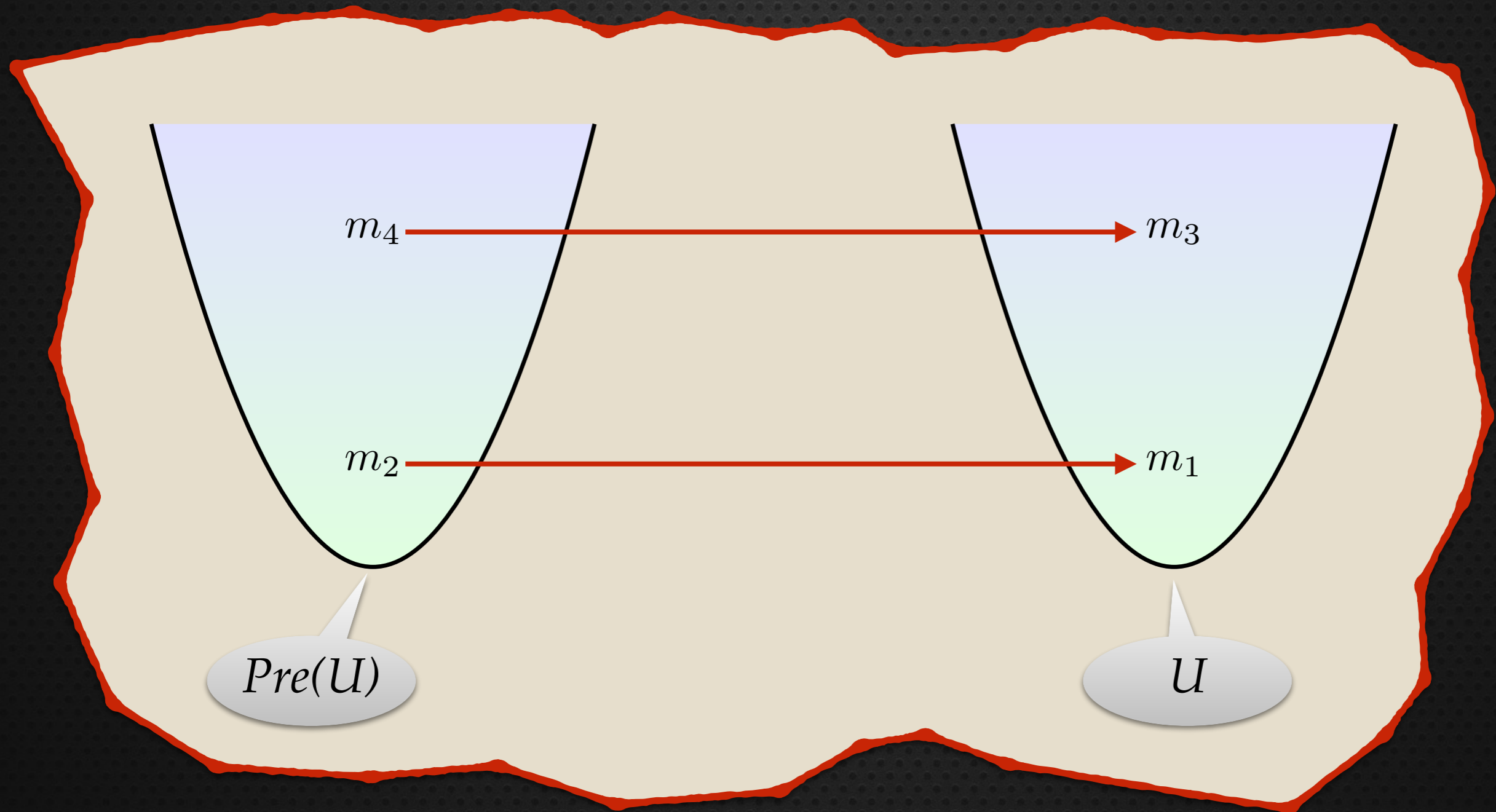


U

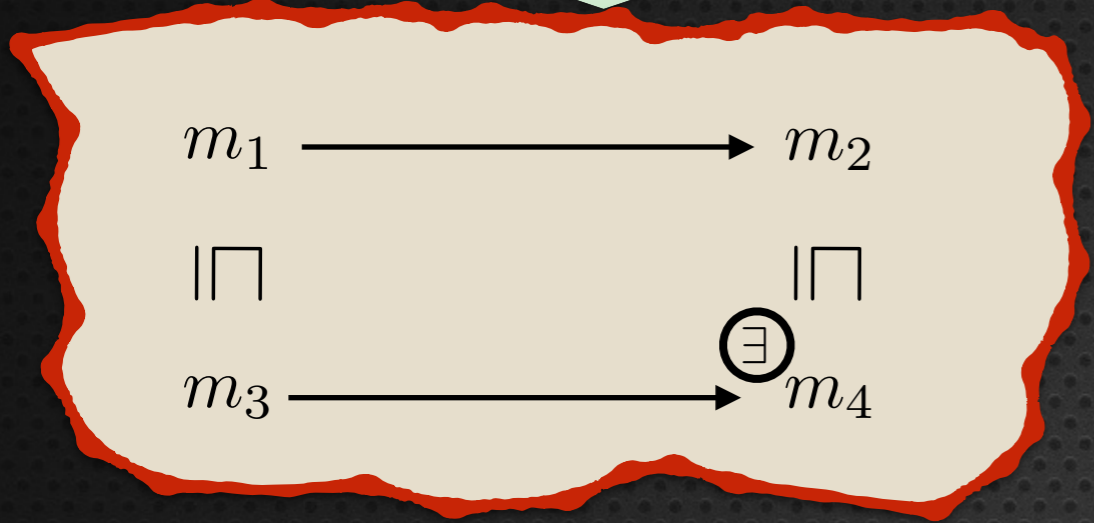
Predecessors



Predecessors



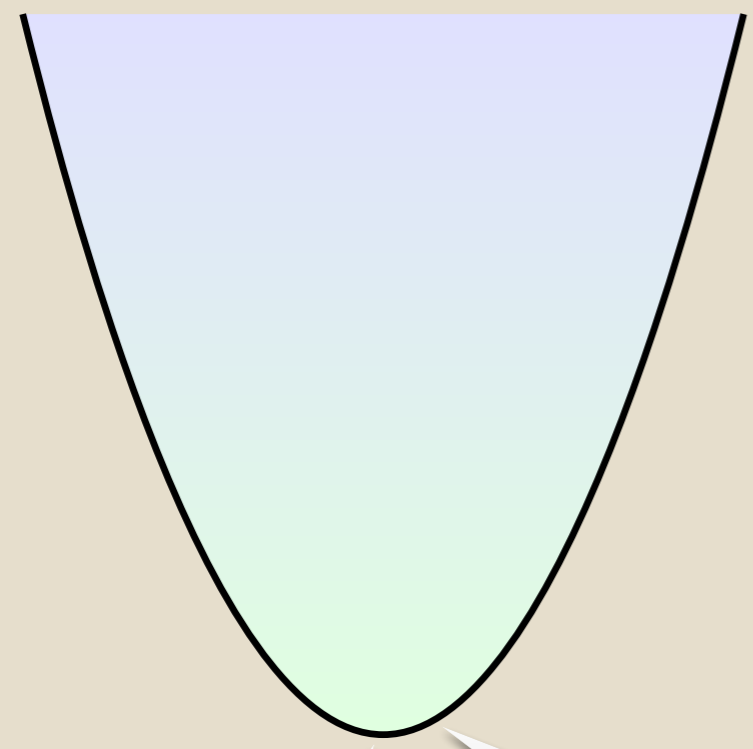
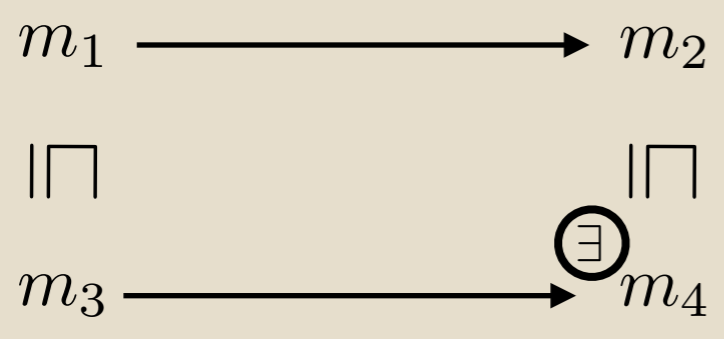
Predecessors



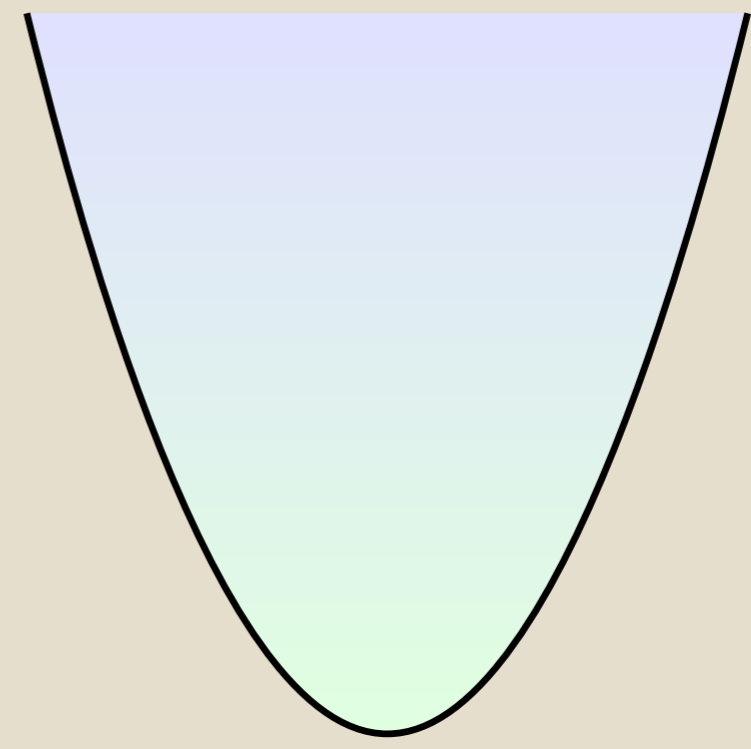
Monotonicity: UC persevered by *Pre*

Predecessors

Monotonicity: UC persevered by *Pre*



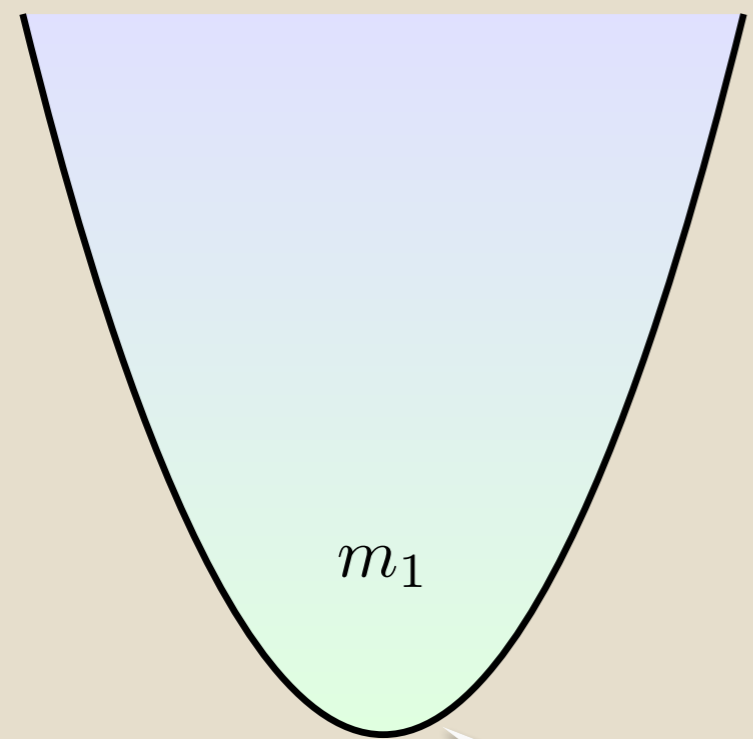
$Pre(U)$ upward closed?



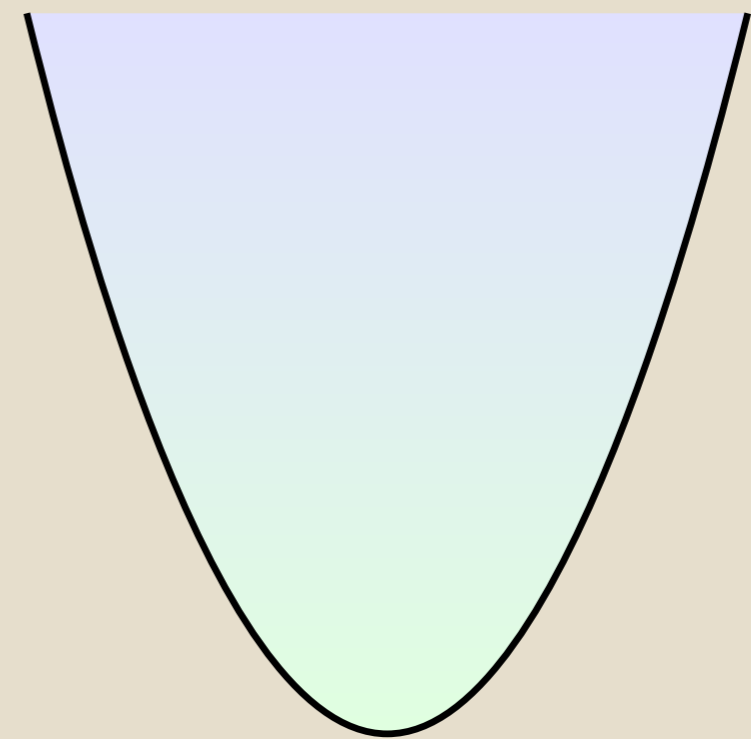
U upward closed

Predecessors

Monotonicity: UC persevered by *Pre*



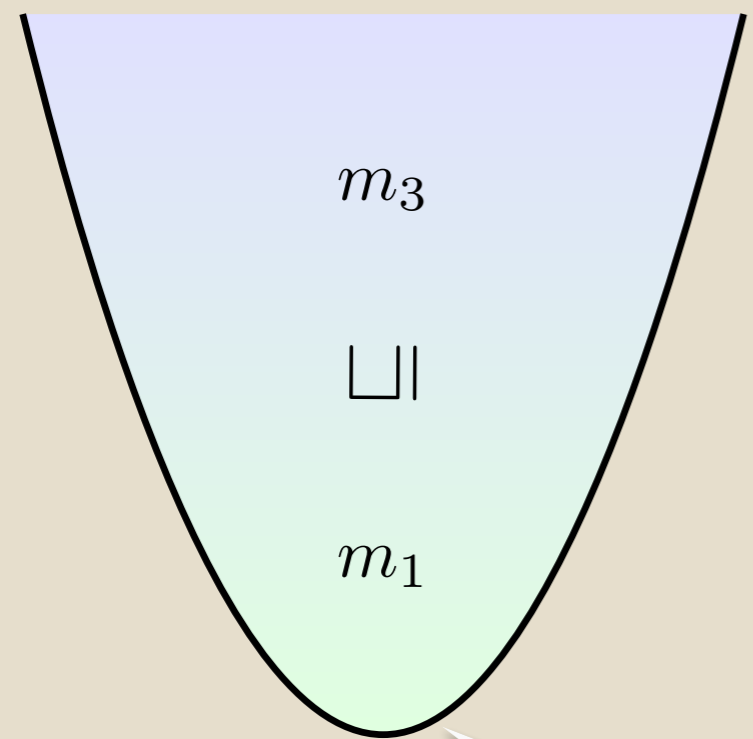
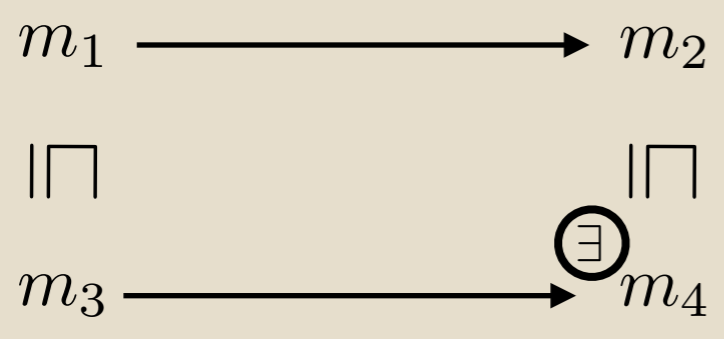
Pre(U) *upward closed?*



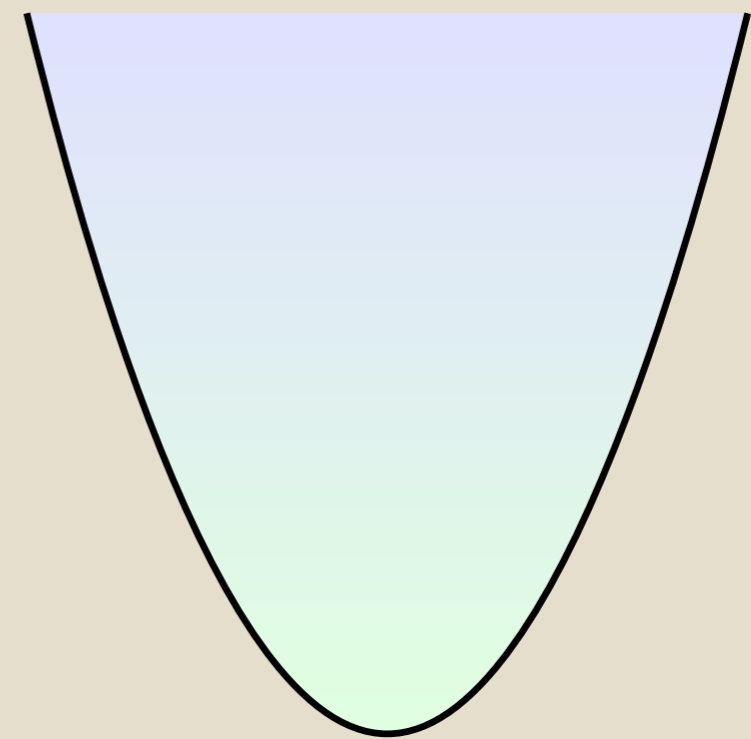
U *upward closed*

Predecessors

Monotonicity: UC persevered by *Pre*



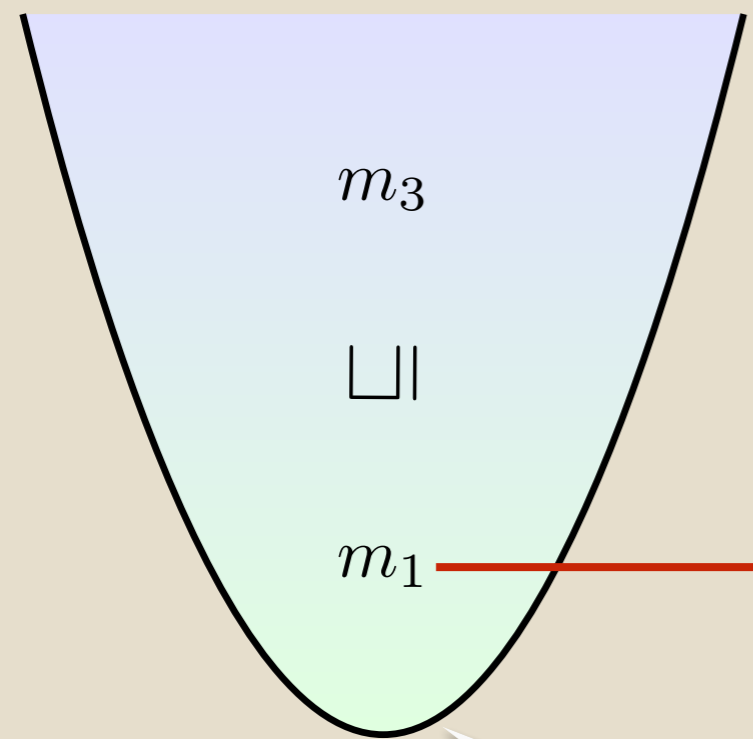
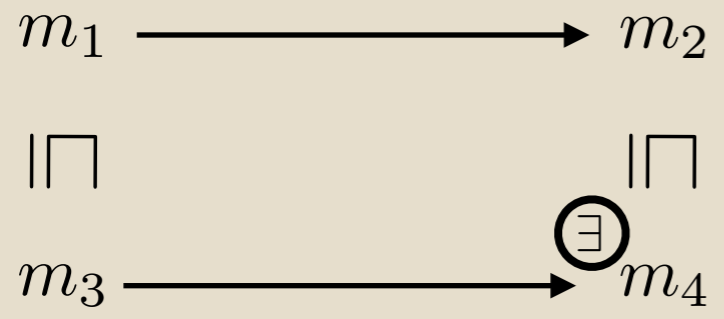
$Pre(U)$ upward closed?



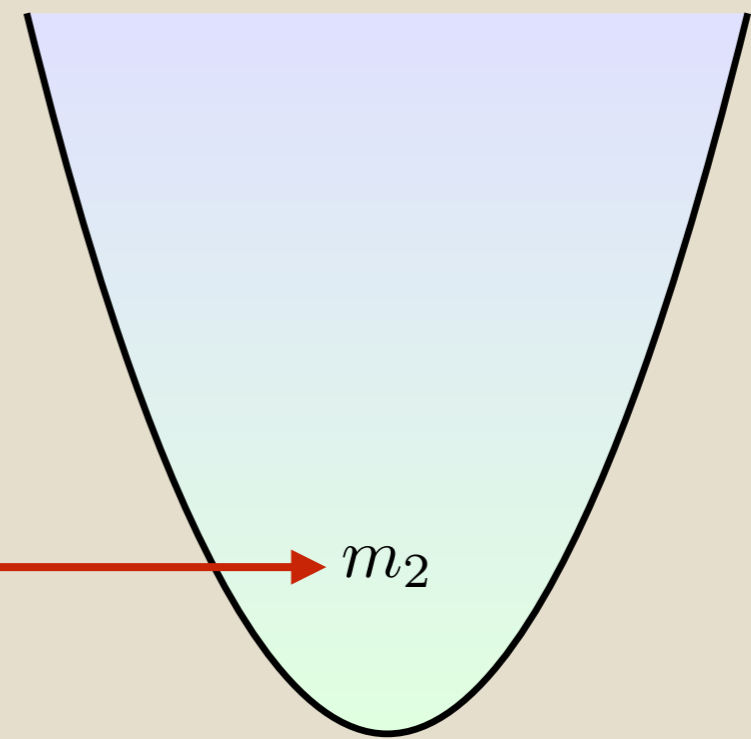
U upward closed

Predecessors

Monotonicity: UC persevered by *Pre*



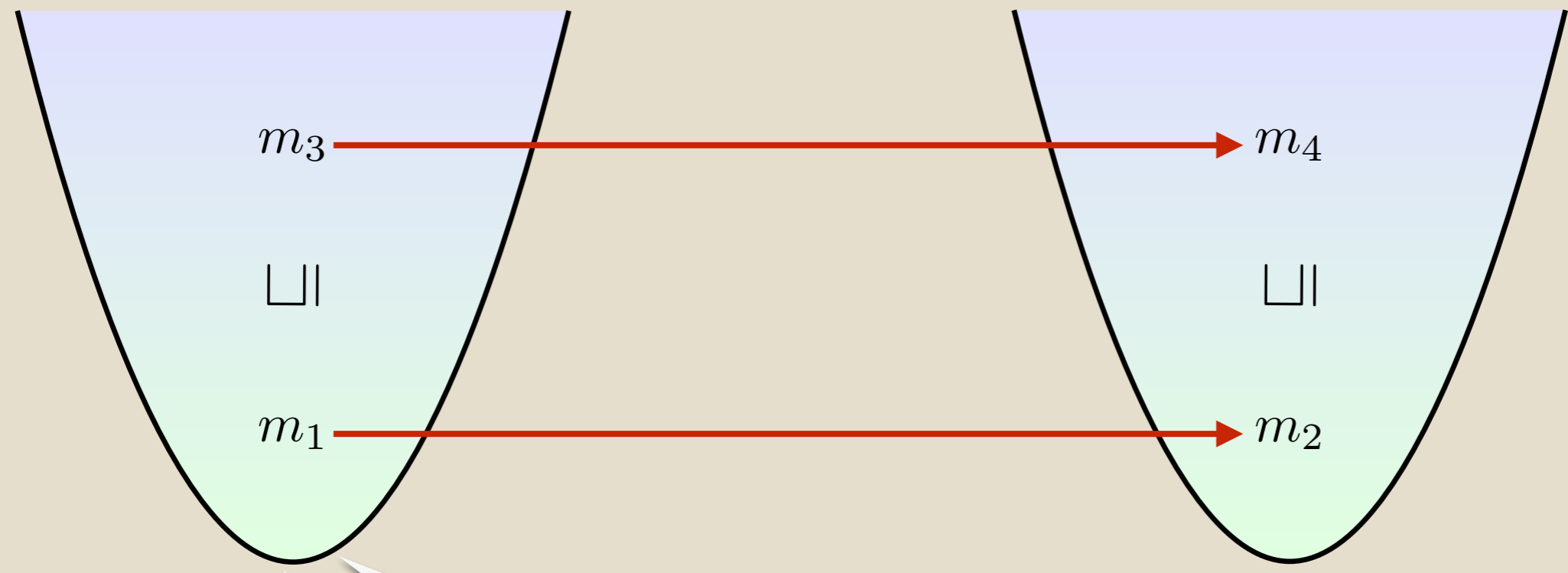
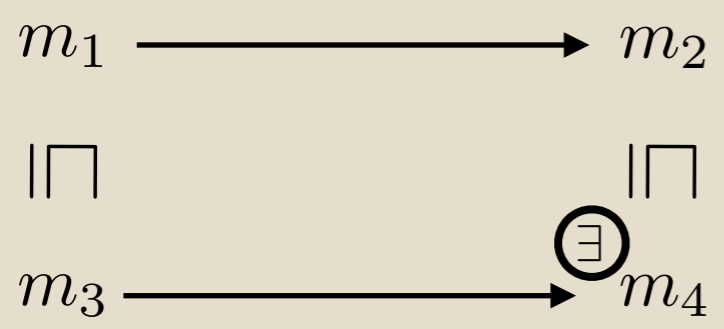
Pre(U) *upward closed?*



U *upward closed*

Predecessors

Monotonicity: UC persevered by *Pre*

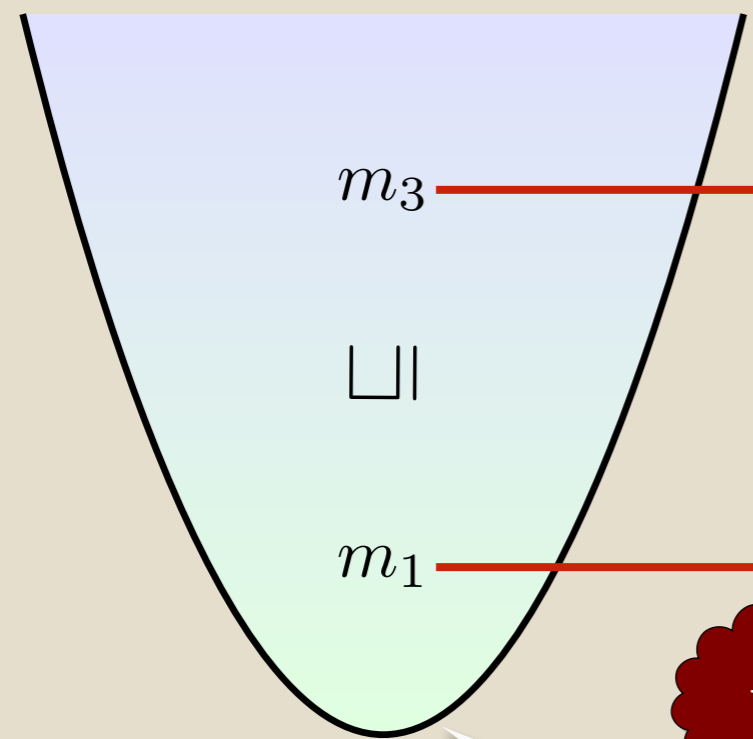


Pre(U) upward closed?

U upward closed

Predecessors

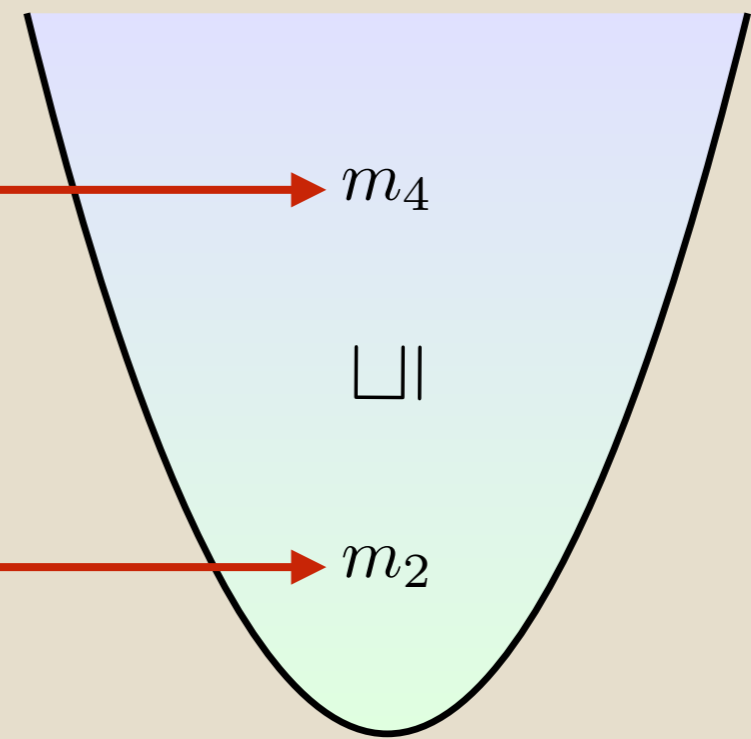
Monotonicity: UC persevered by *Pre*



Pre(U)

upward closed?

yes

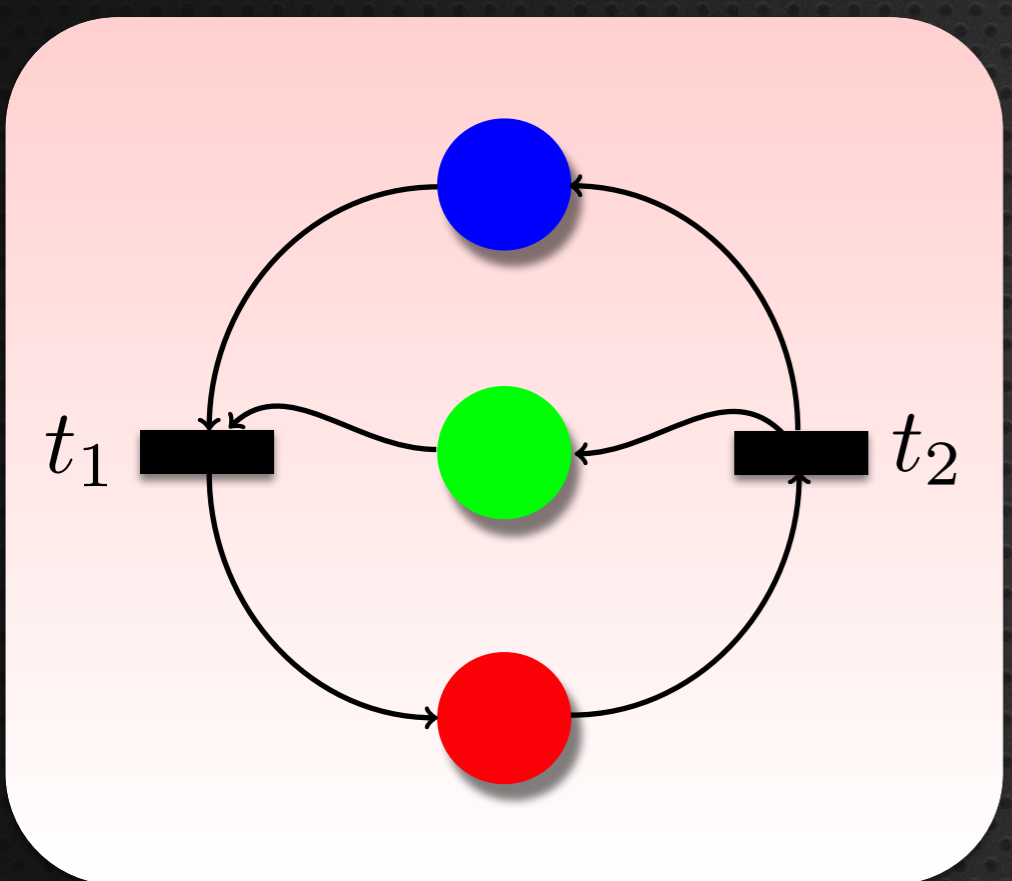


U

upward closed

Petri Nets

Computing Predecessors

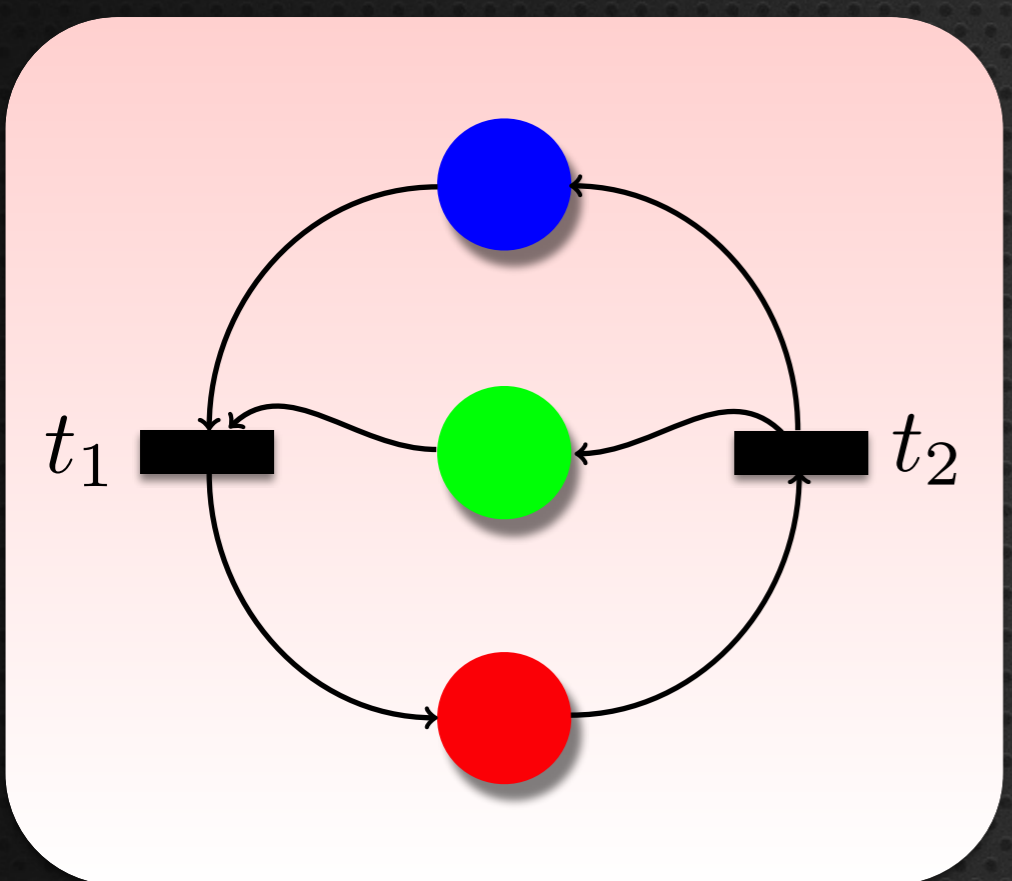


$$Pre_{t_1} \left[\begin{array}{c} \text{yellow bowl} \\ \text{2 red tokens} \end{array} \right] = \begin{array}{c} \text{light green bowl} \\ \text{1 red, 1 green, 1 blue token} \end{array}$$

$$Pre_{t_2} \left[\begin{array}{c} \text{yellow bowl} \\ \text{2 red tokens} \end{array} \right] =$$

Petri Nets

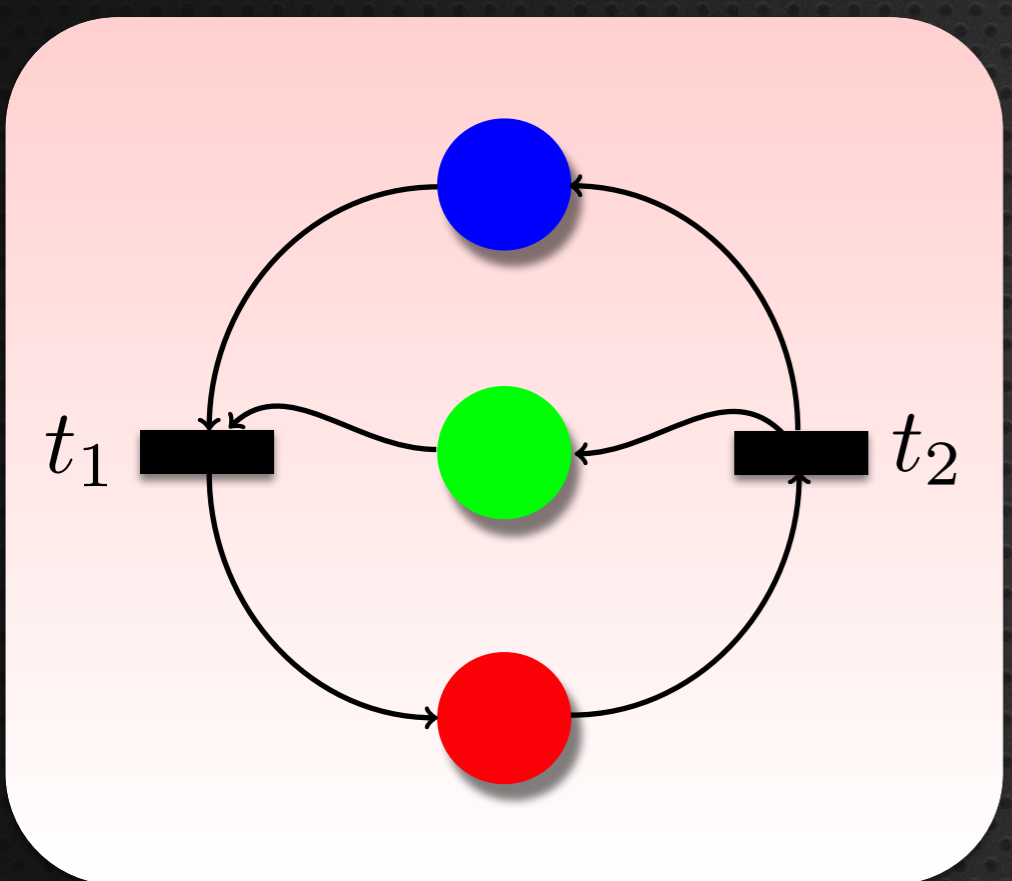
Computing Predecessors



$$Pre_{t_1} \left[\begin{array}{c} \text{Yellow bowl} \\ \text{2 red balls} \end{array} \right] = \begin{array}{c} \text{Light green bowl} \\ \text{1 red, 1 green, 1 blue ball} \end{array}$$

$$Pre_{t_2} \left[\begin{array}{c} \text{Yellow bowl} \\ \text{2 red balls} \end{array} \right] = \begin{array}{c} \text{Light green bowl} \\ \text{3 red balls} \end{array}$$

Petri Nets Computing Predecessors



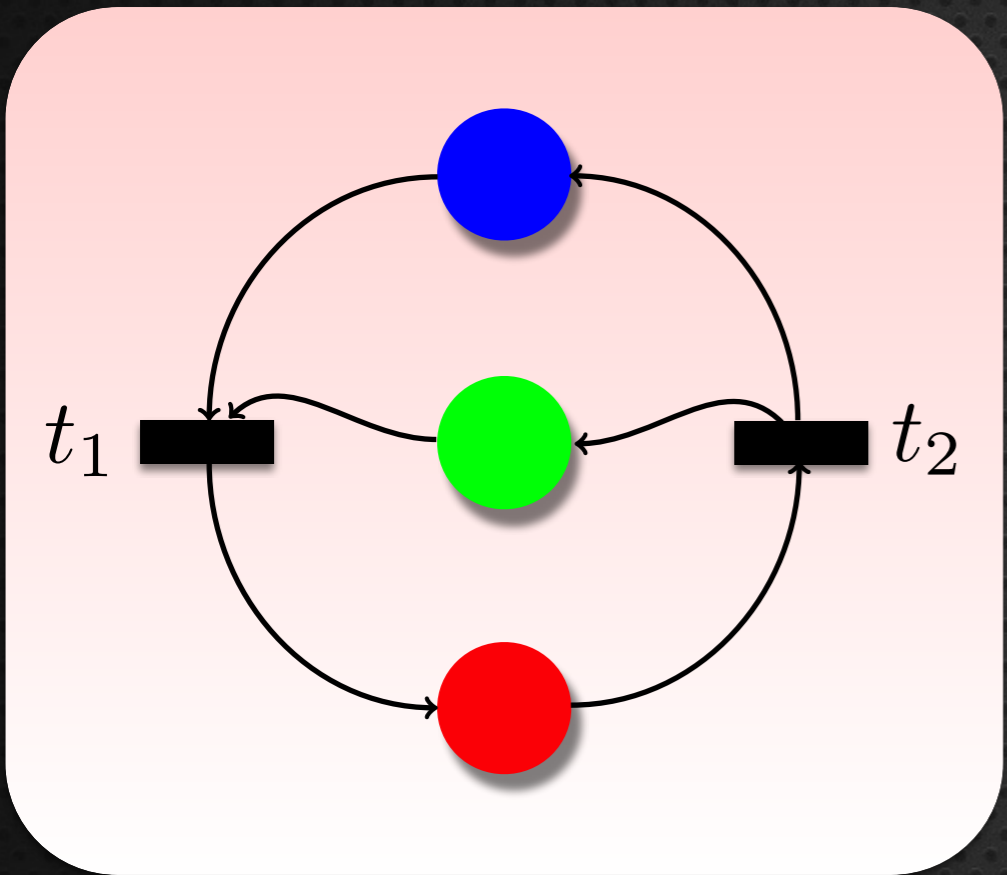
$$Pre_{t_1} \left[\begin{array}{c} \text{Yellow bowl} \\ \text{2 red tokens} \end{array} \right] = \begin{array}{c} \text{Light green bowl} \\ \text{1 red, 1 green, 1 blue token} \end{array}$$

$$Pre_{t_2} \left[\begin{array}{c} \text{Yellow bowl} \\ \text{2 red tokens} \end{array} \right] = \begin{array}{c} \text{Light green bowl} \\ \text{3 red tokens} \end{array}$$

$$Pre_{t_1} \left[\begin{array}{c} \text{Yellow bowl} \\ \text{1 red, 1 green, 1 blue token} \end{array} \right] =$$

Petri Nets

Computing Predecessors

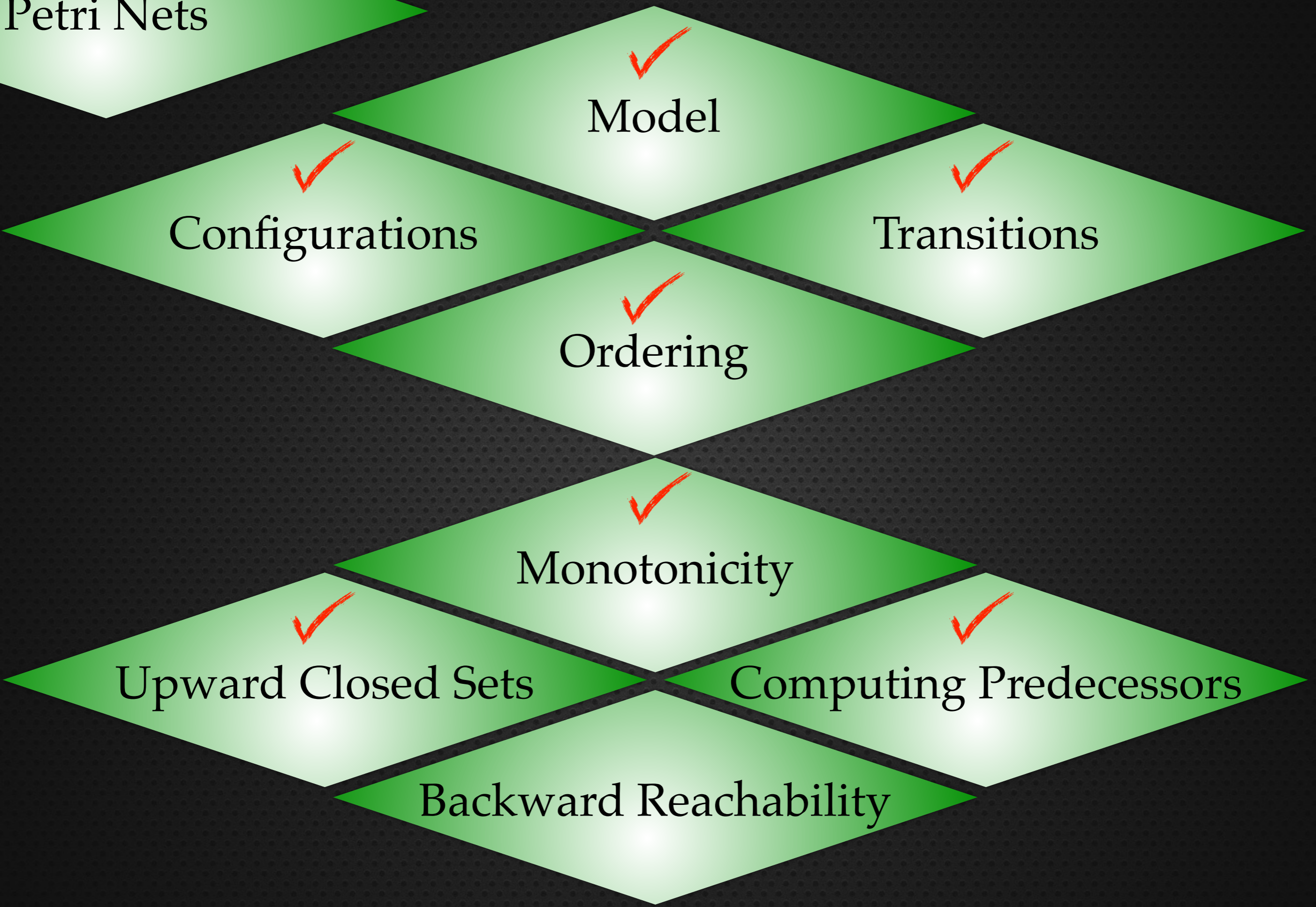


$$Pre_{t_1} \left[\begin{array}{c} \text{Yellow bowl} \\ \text{2 red tokens} \end{array} \right] = \begin{array}{c} \text{Green bowl} \\ \text{1 red, 1 green, 1 blue token} \end{array}$$

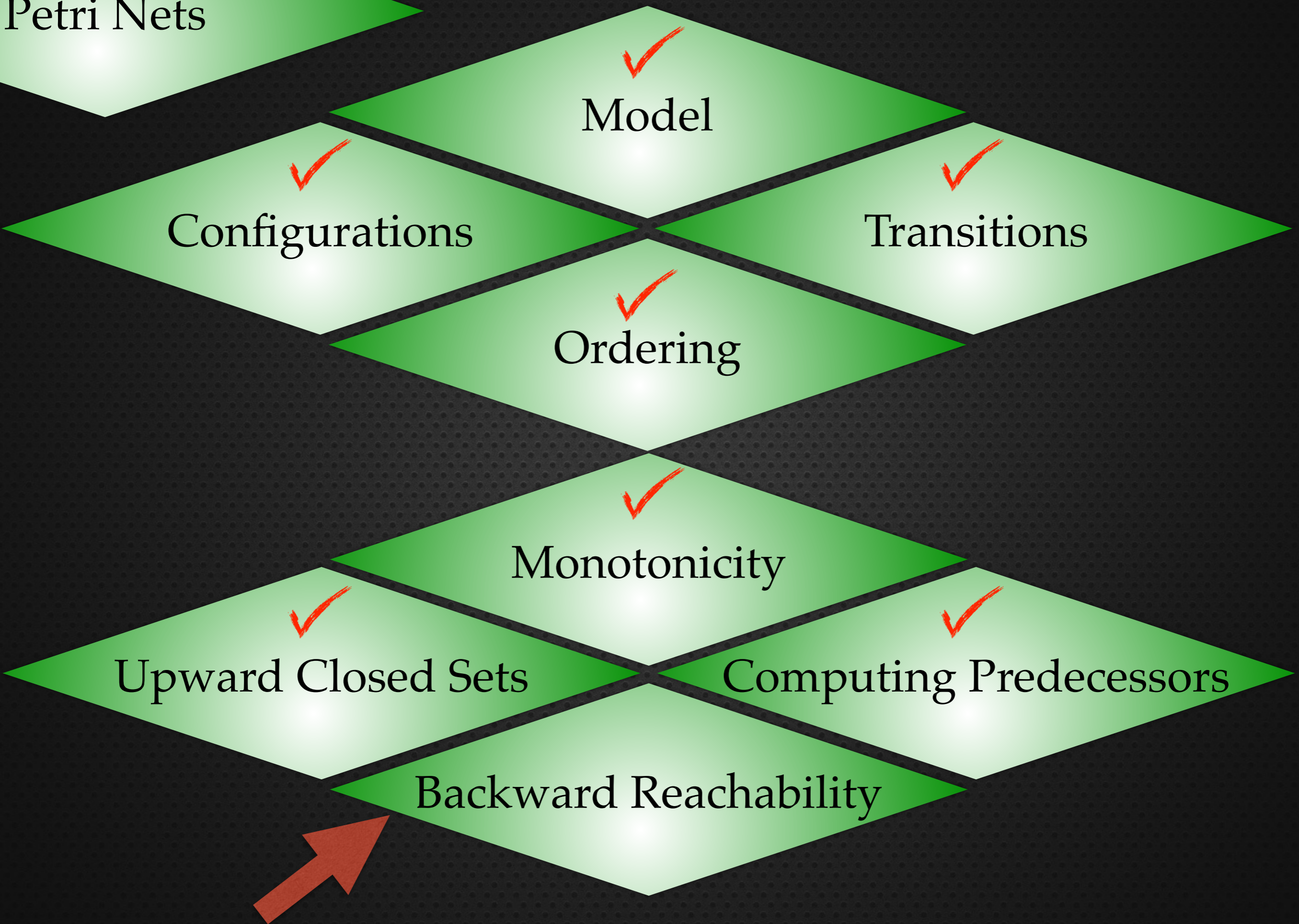
$$Pre_{t_2} \left[\begin{array}{c} \text{Yellow bowl} \\ \text{2 red tokens} \end{array} \right] = \begin{array}{c} \text{Green bowl} \\ \text{3 red tokens} \end{array}$$

$$Pre_{t_1} \left[\begin{array}{c} \text{Yellow bowl} \\ \text{1 red, 1 green, 1 blue token} \end{array} \right] = \begin{array}{c} \text{Green bowl} \\ \text{2 green, 1 blue, 1 red token} \end{array}$$

Petri Nets

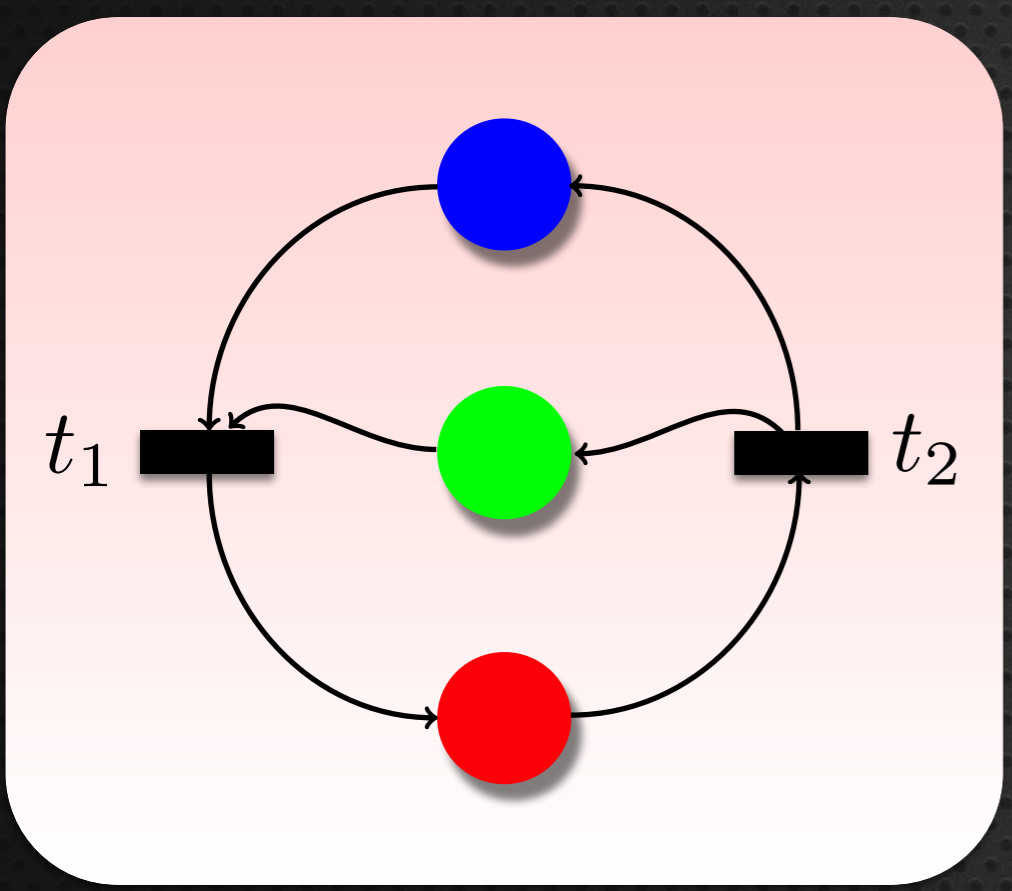


Petri Nets



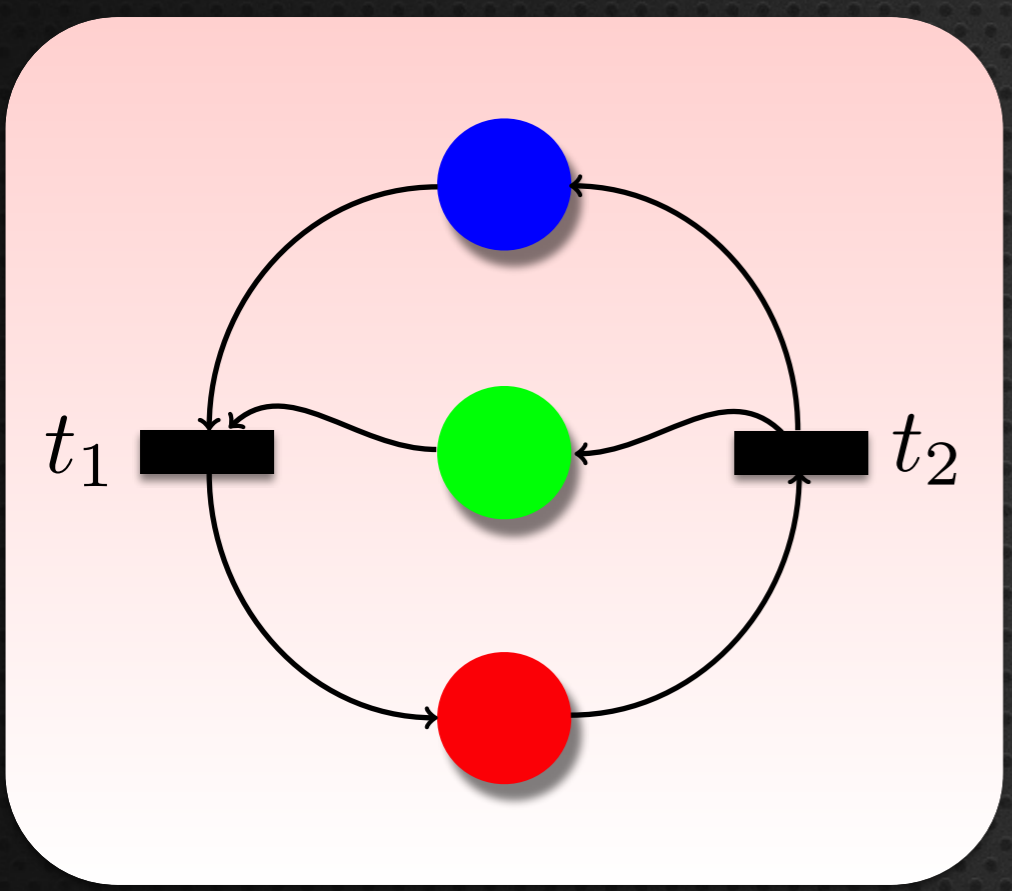
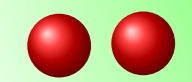
Petri Net

Backward Reachability



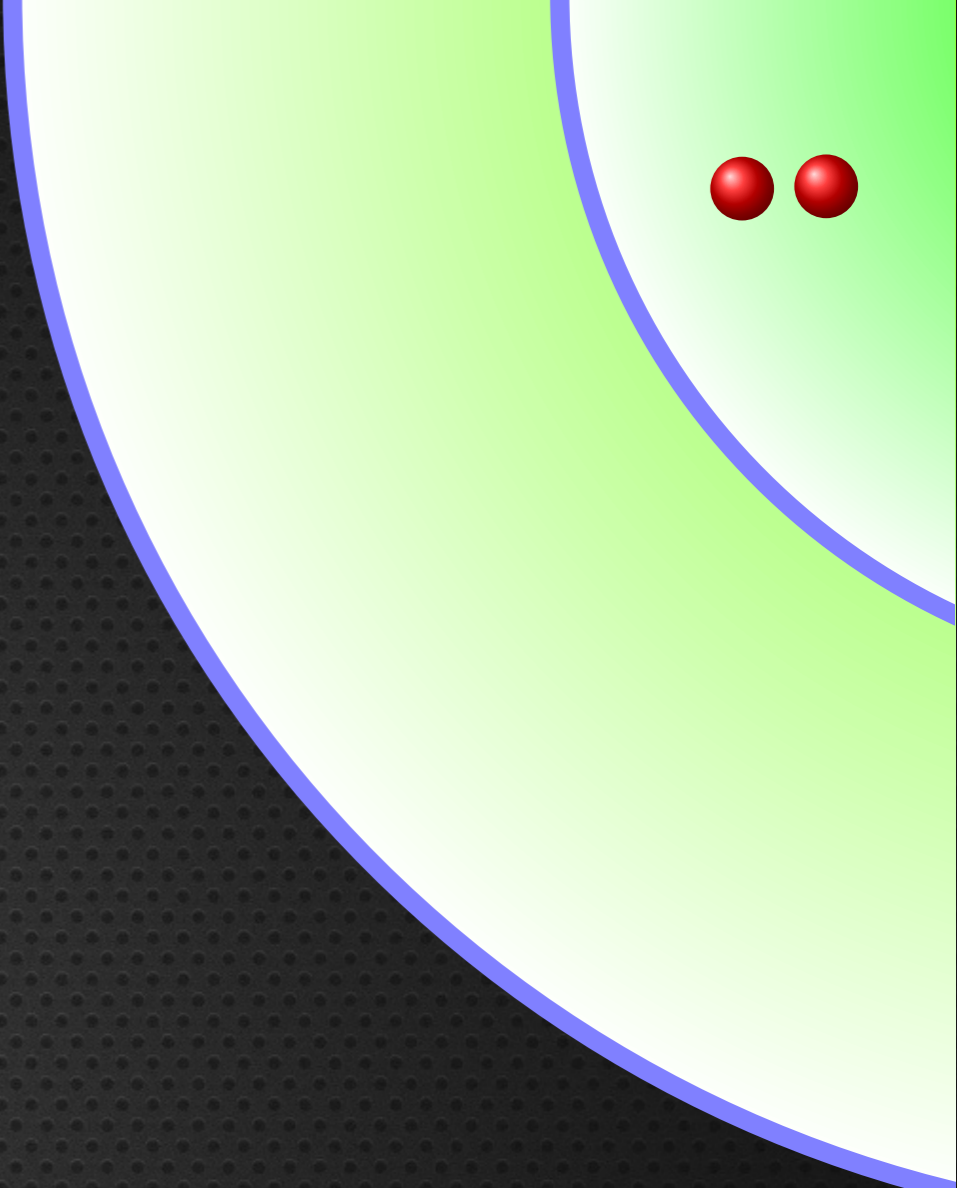
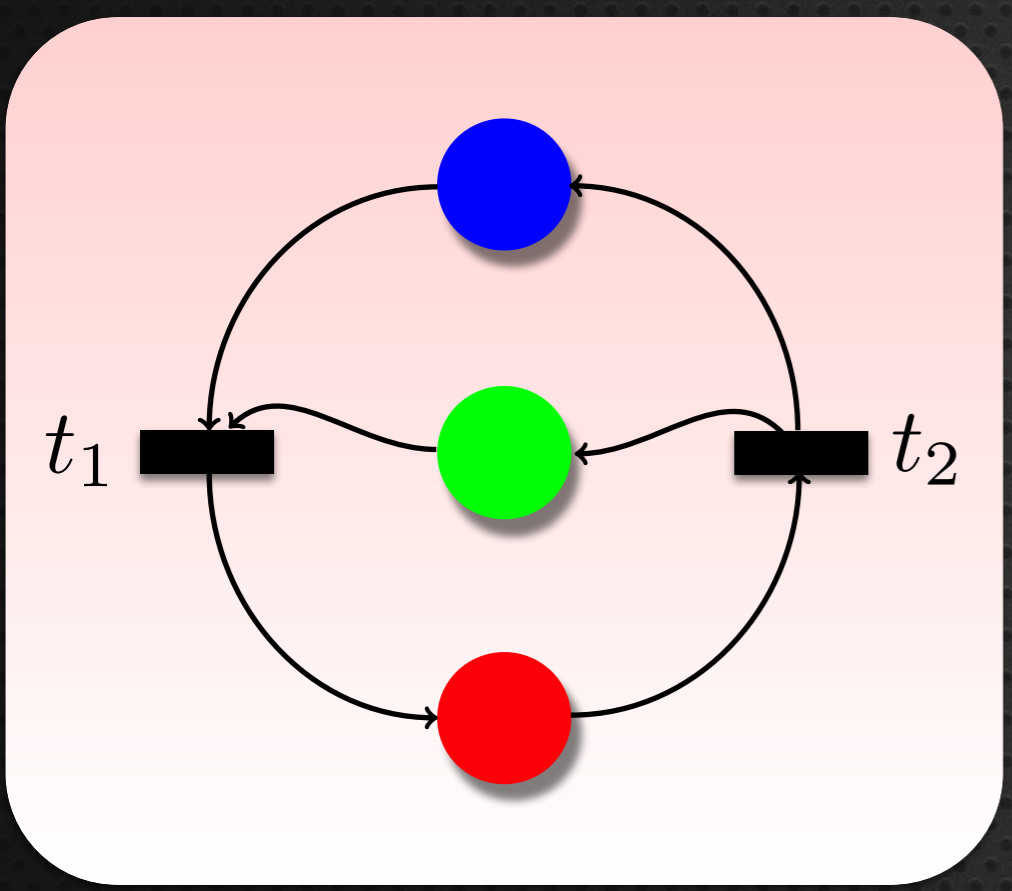
Petri Net

Backward Reachability



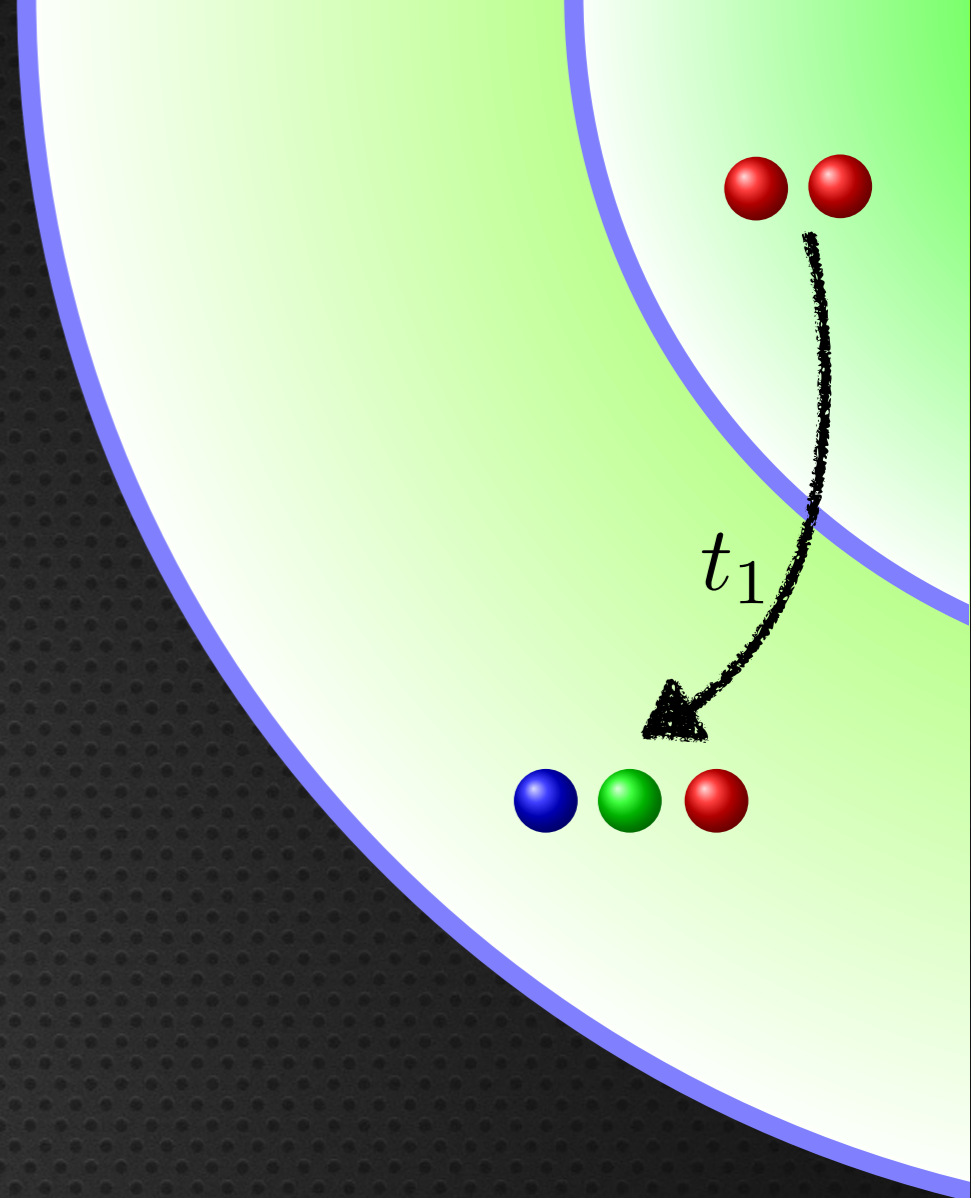
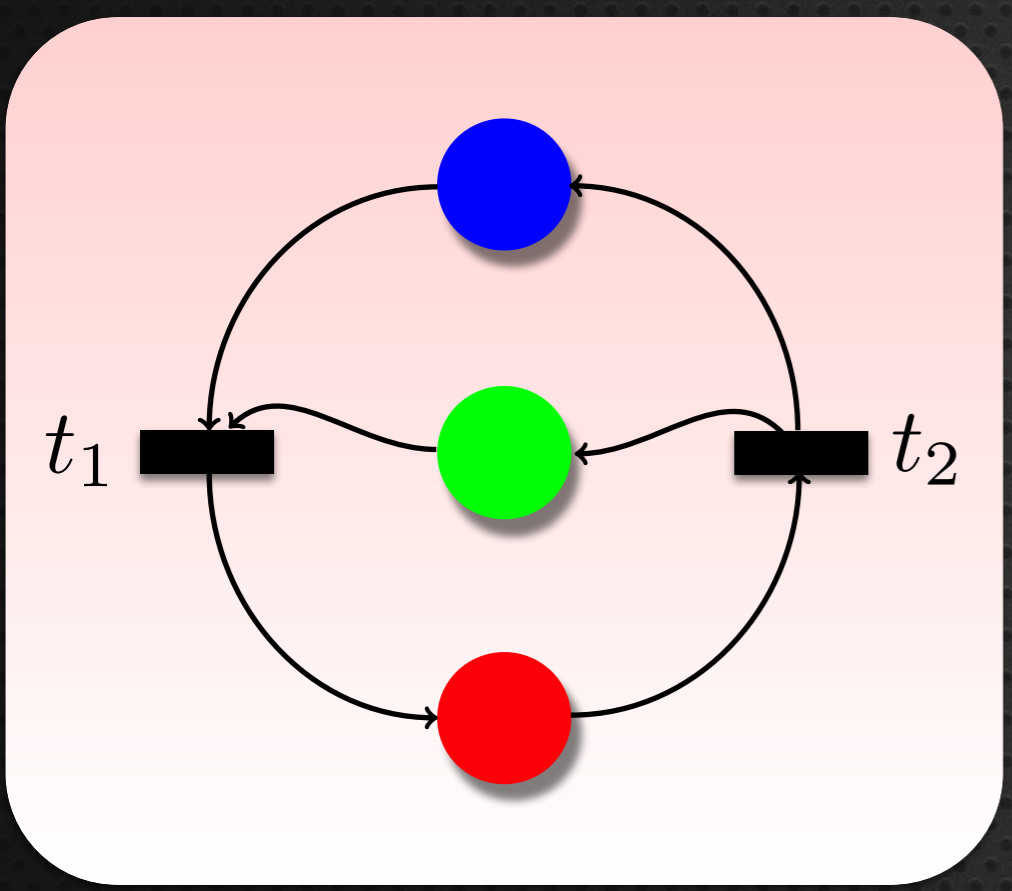
Petri Net

Backward Reachability



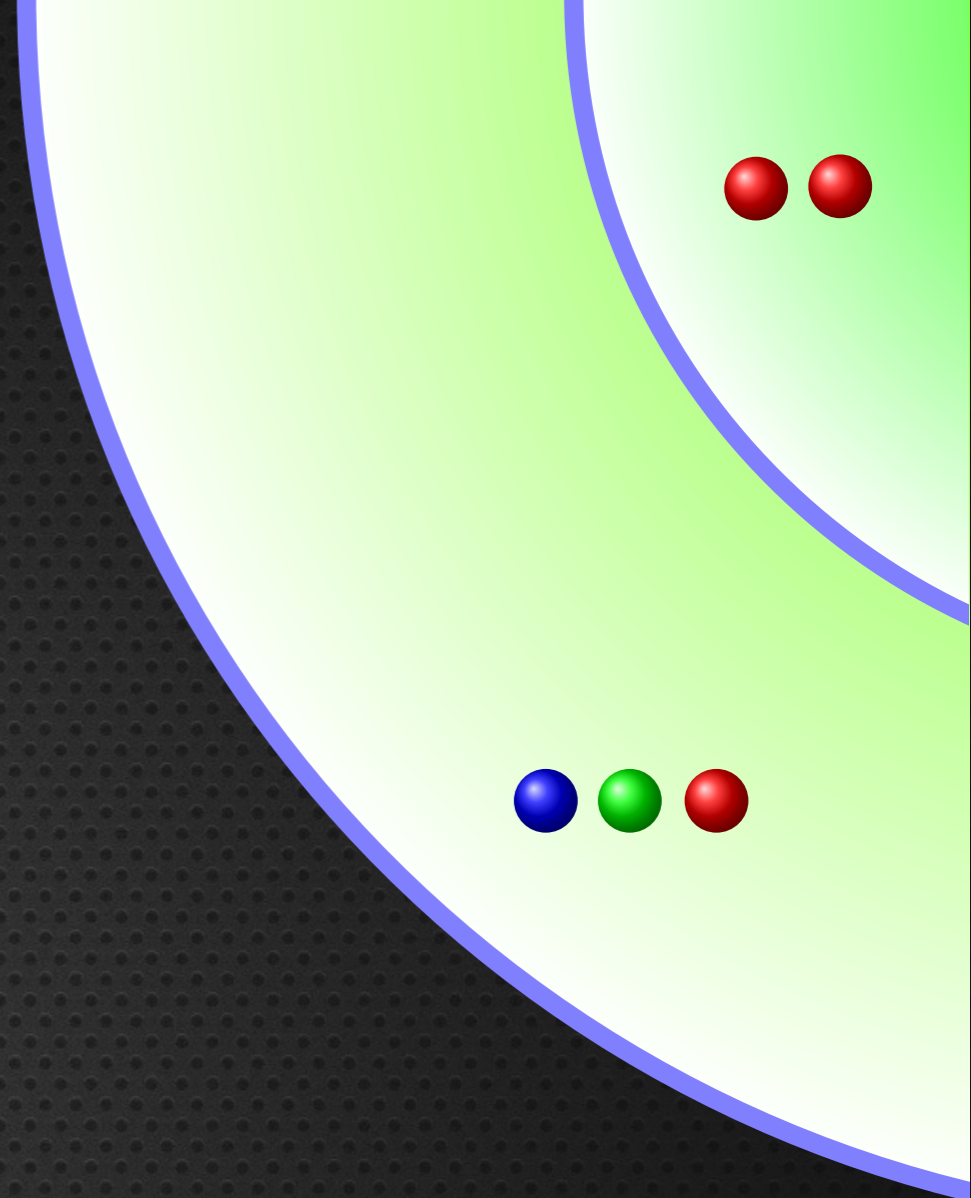
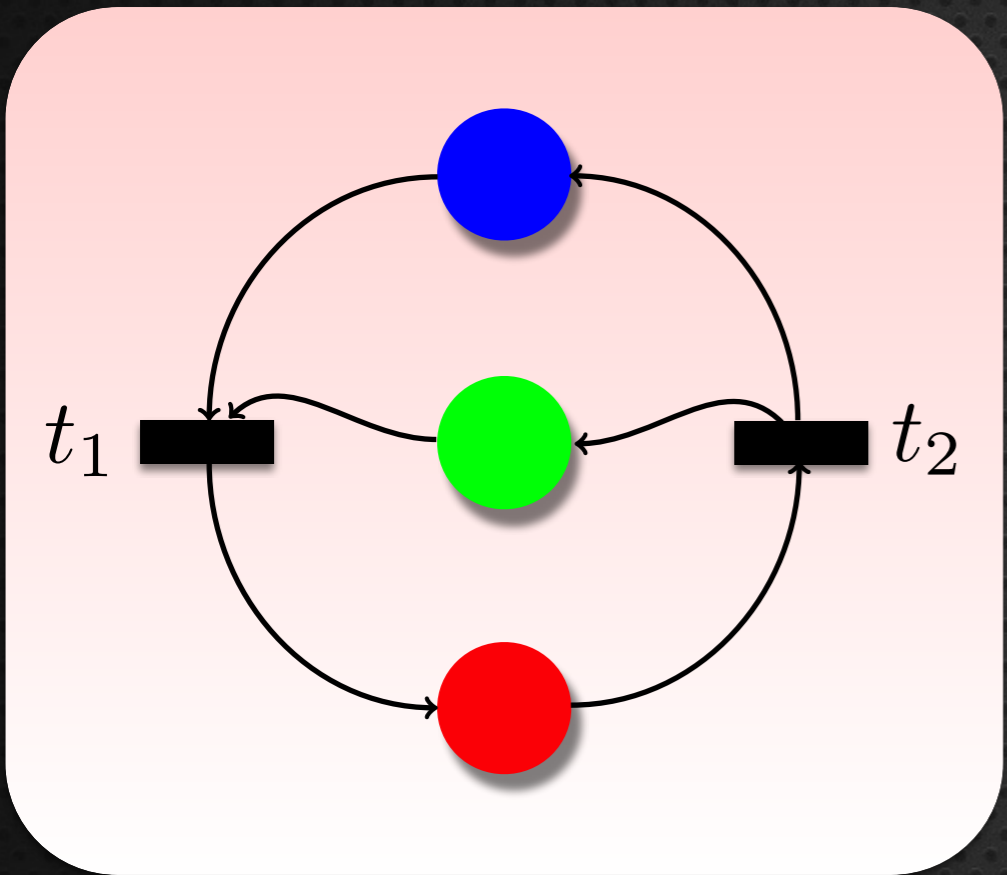
Petri Nets

Backward Reachability



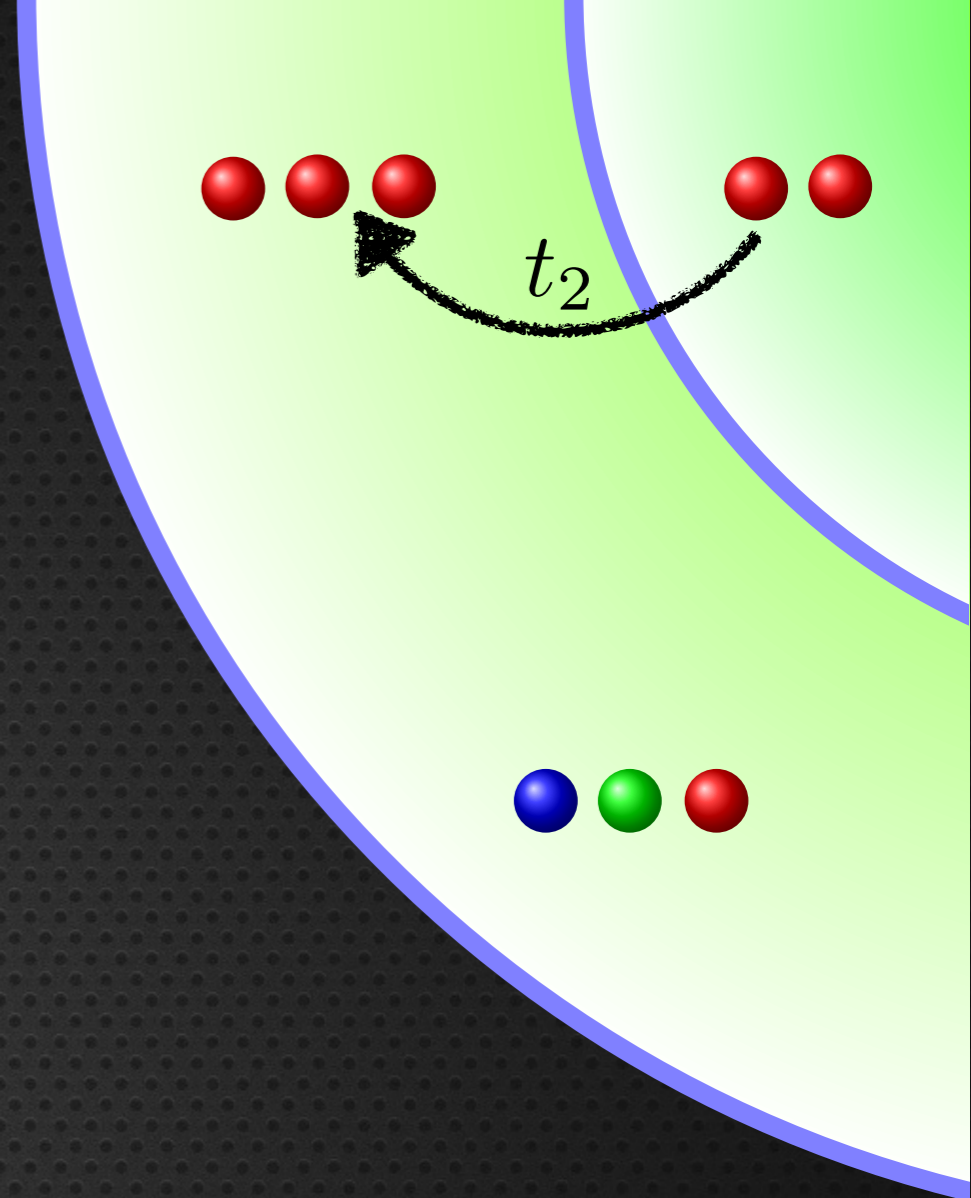
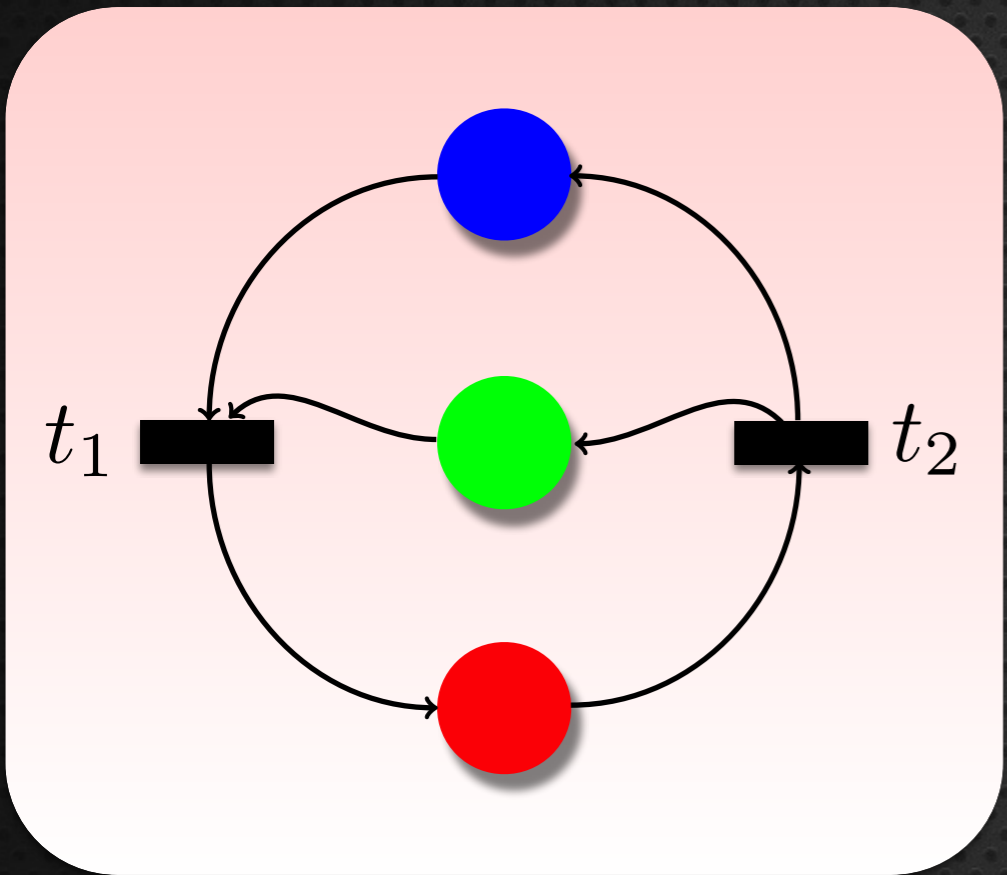
Petri Net

Backward Reachability



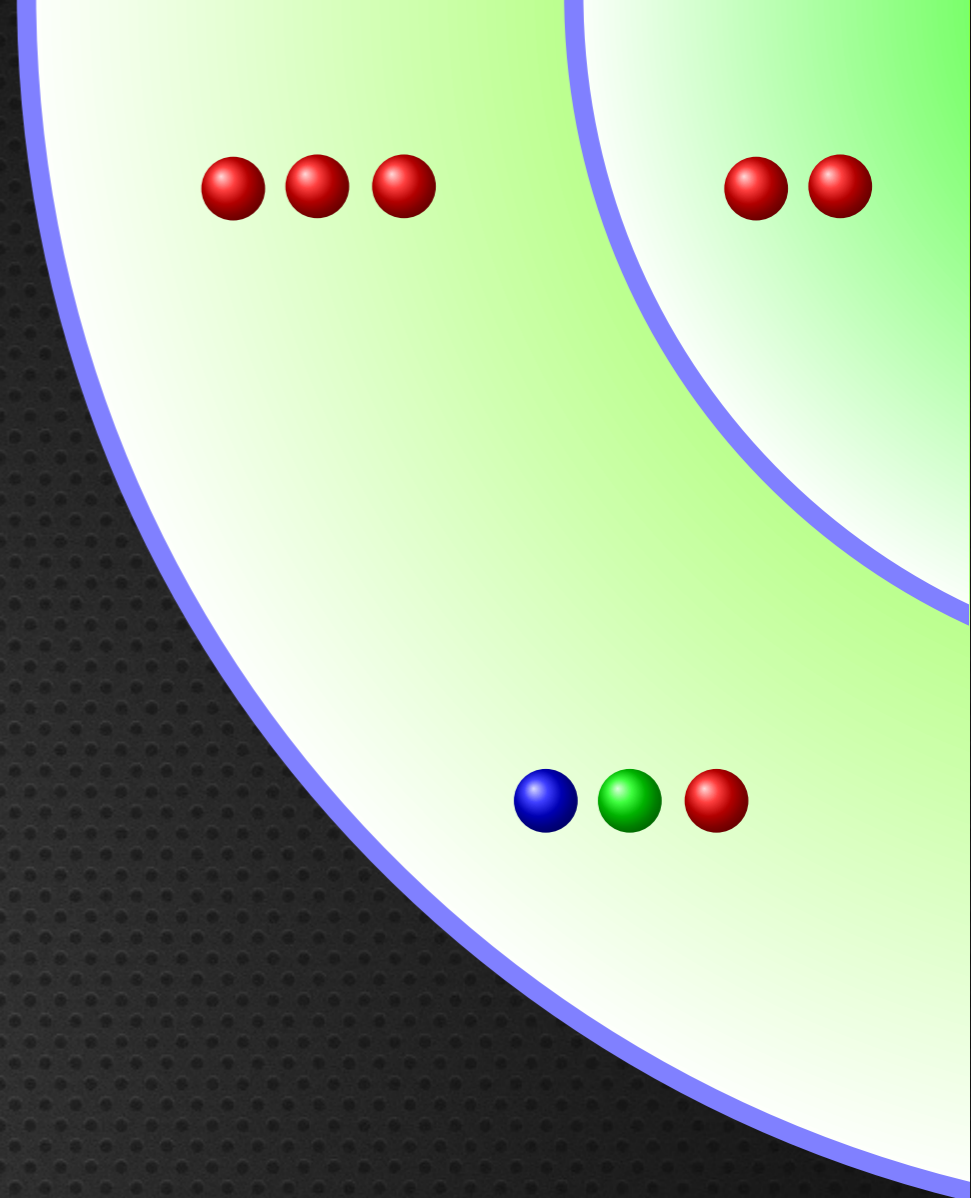
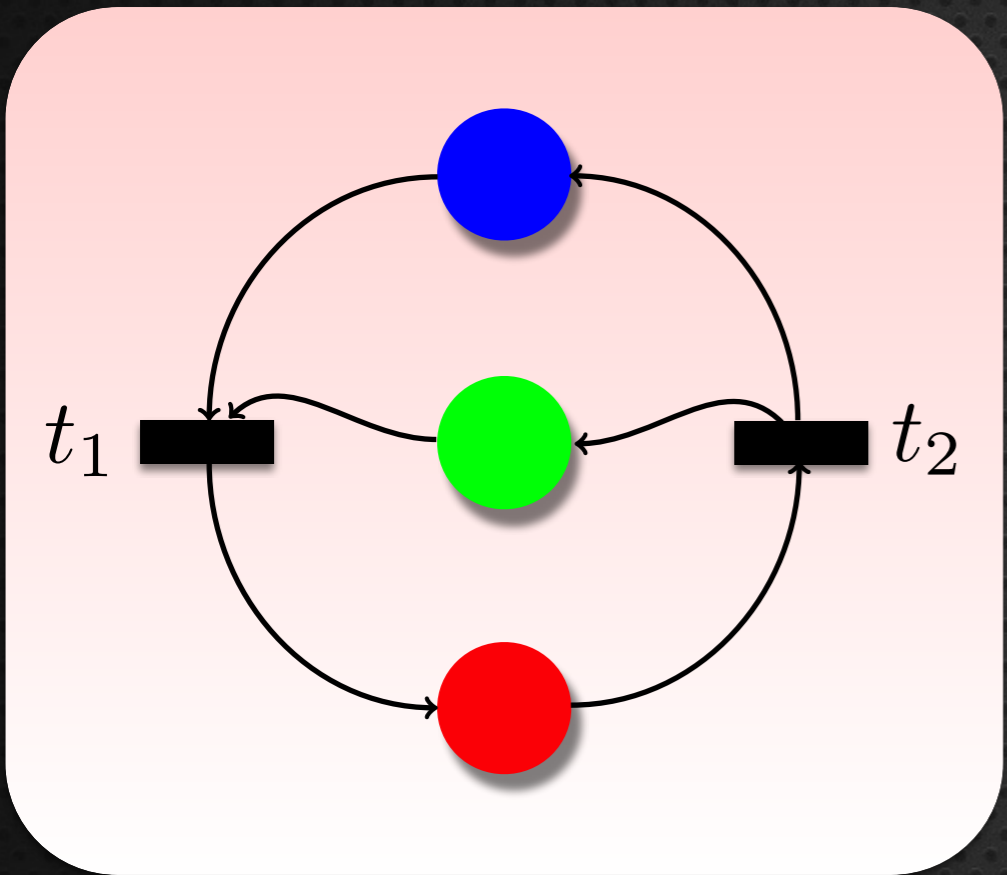
Petri Nets

Backward Reachability



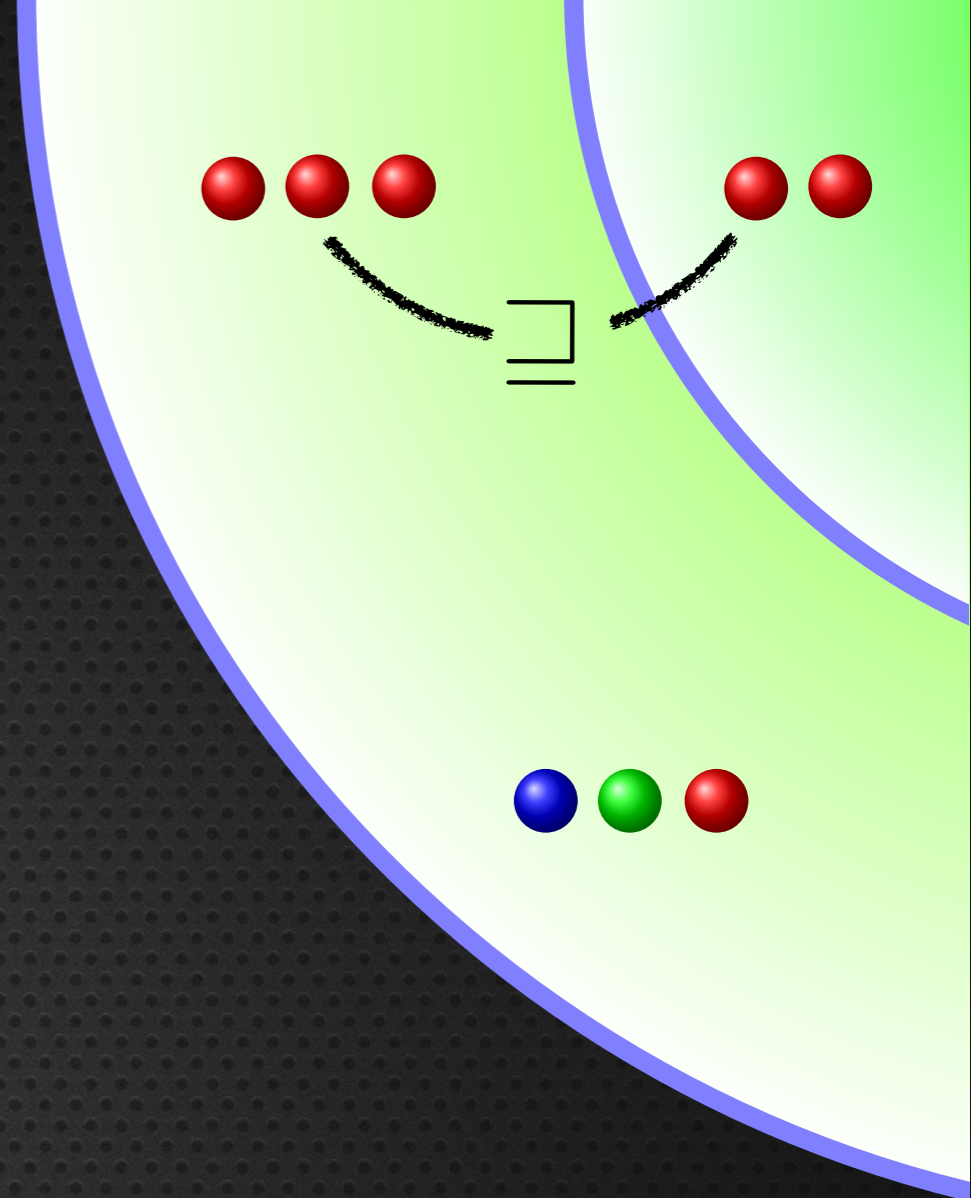
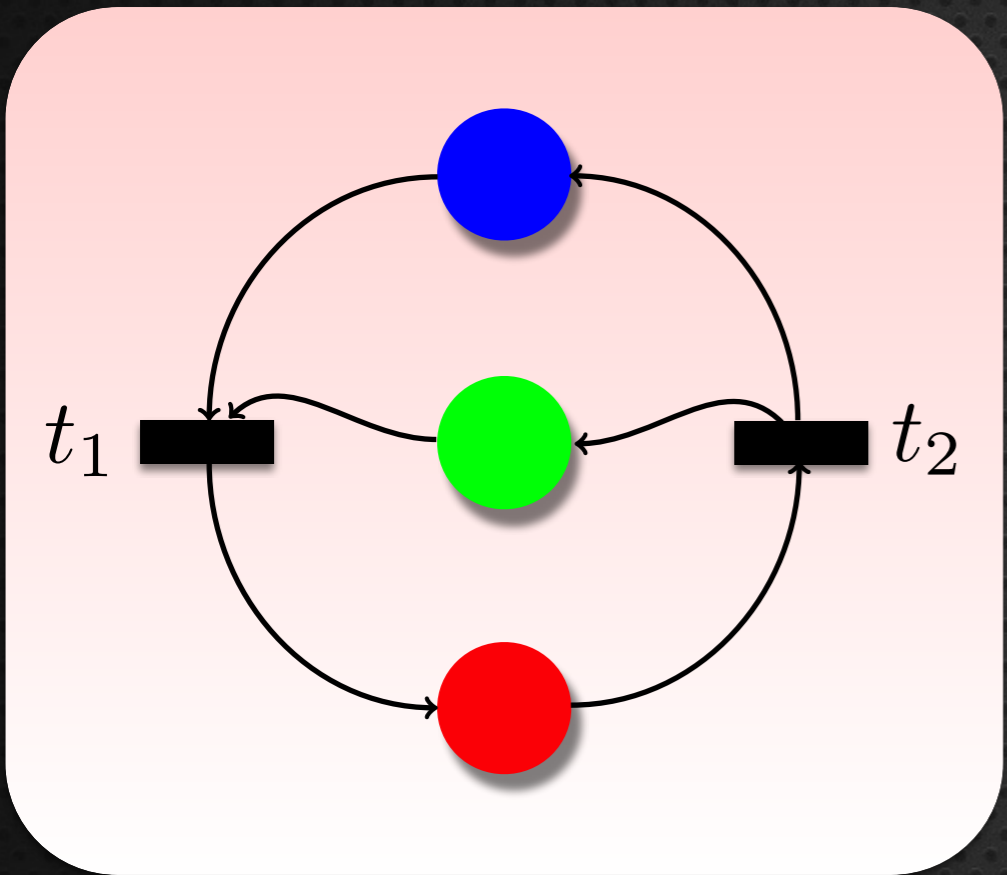
Petri Nets

Backward Reachability



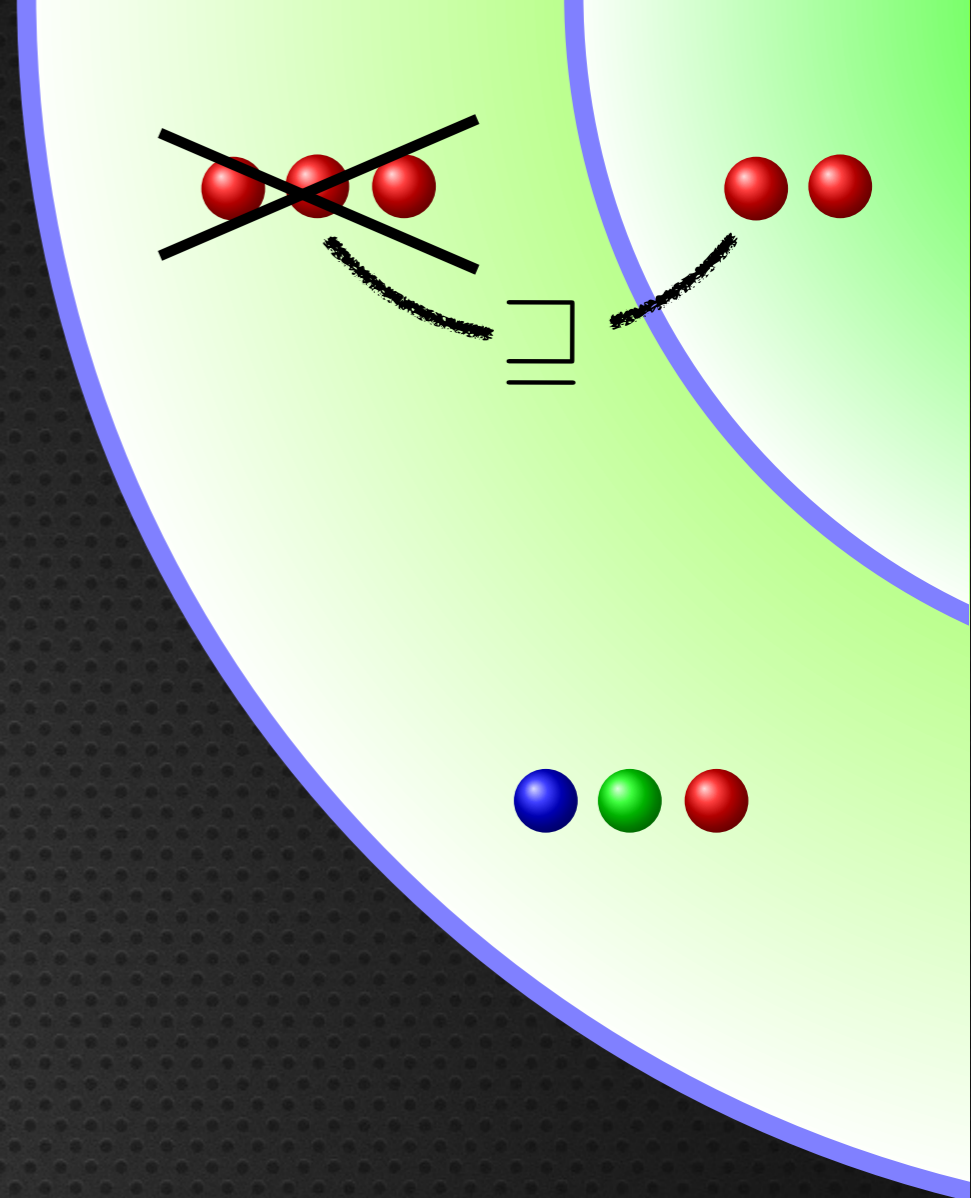
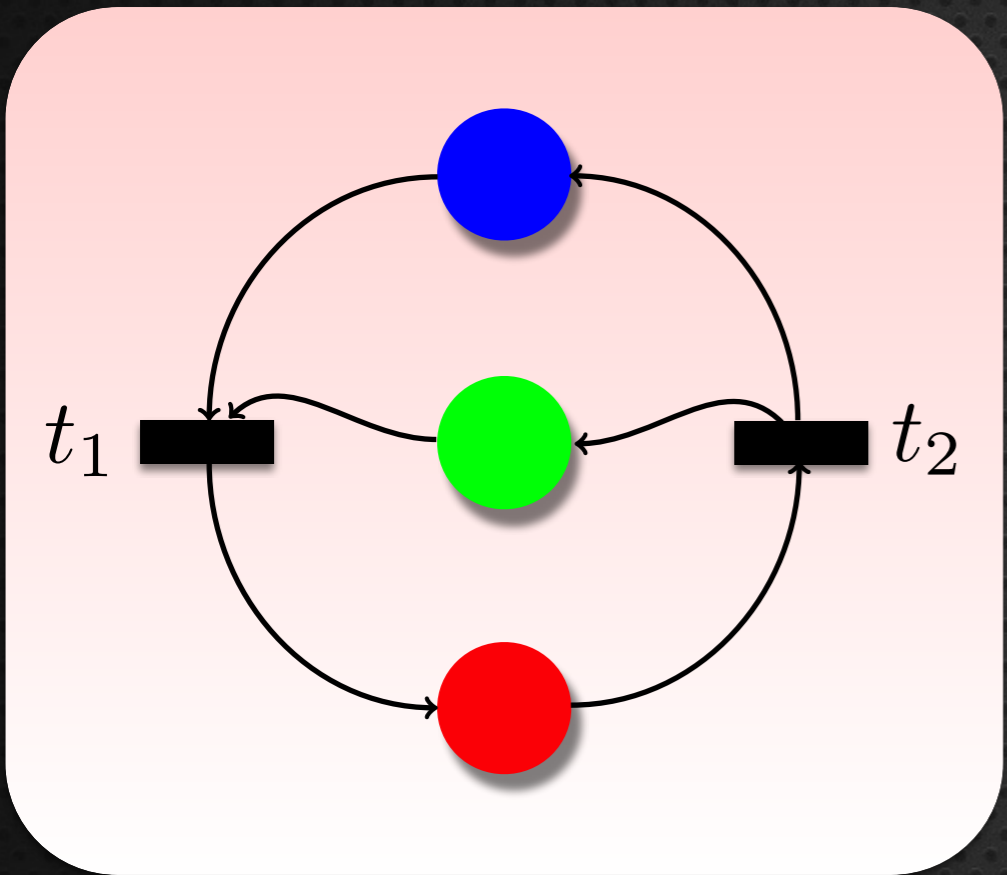
Petri Nets

Backward Reachability



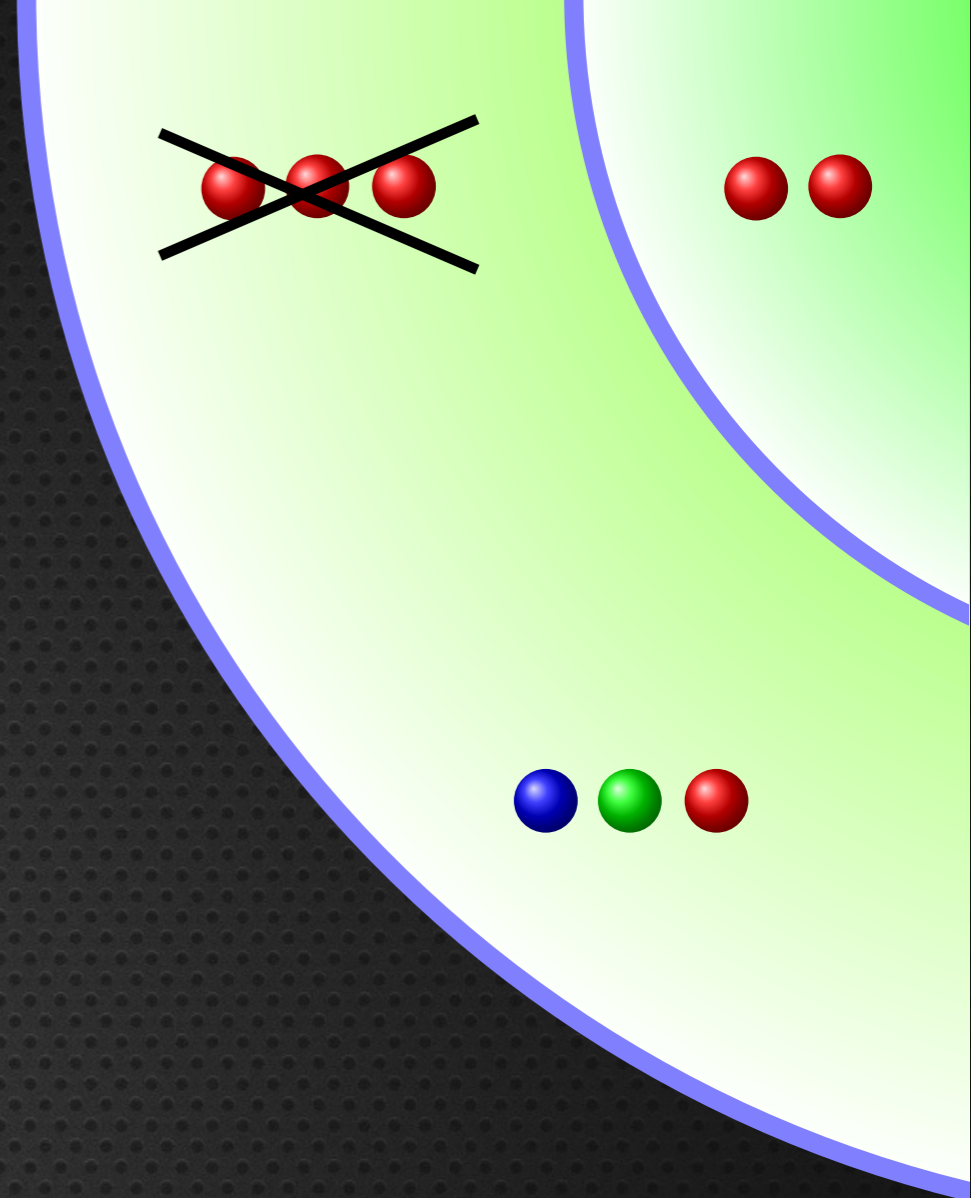
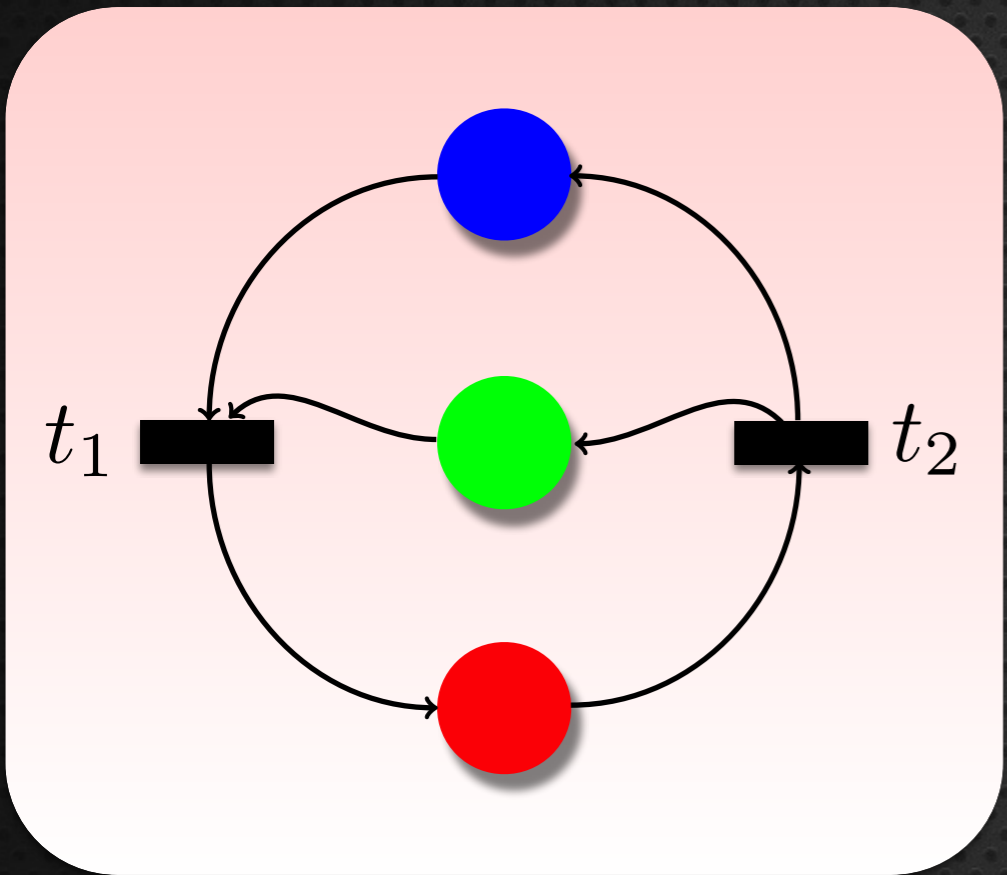
Petri Nets

Backward Reachability



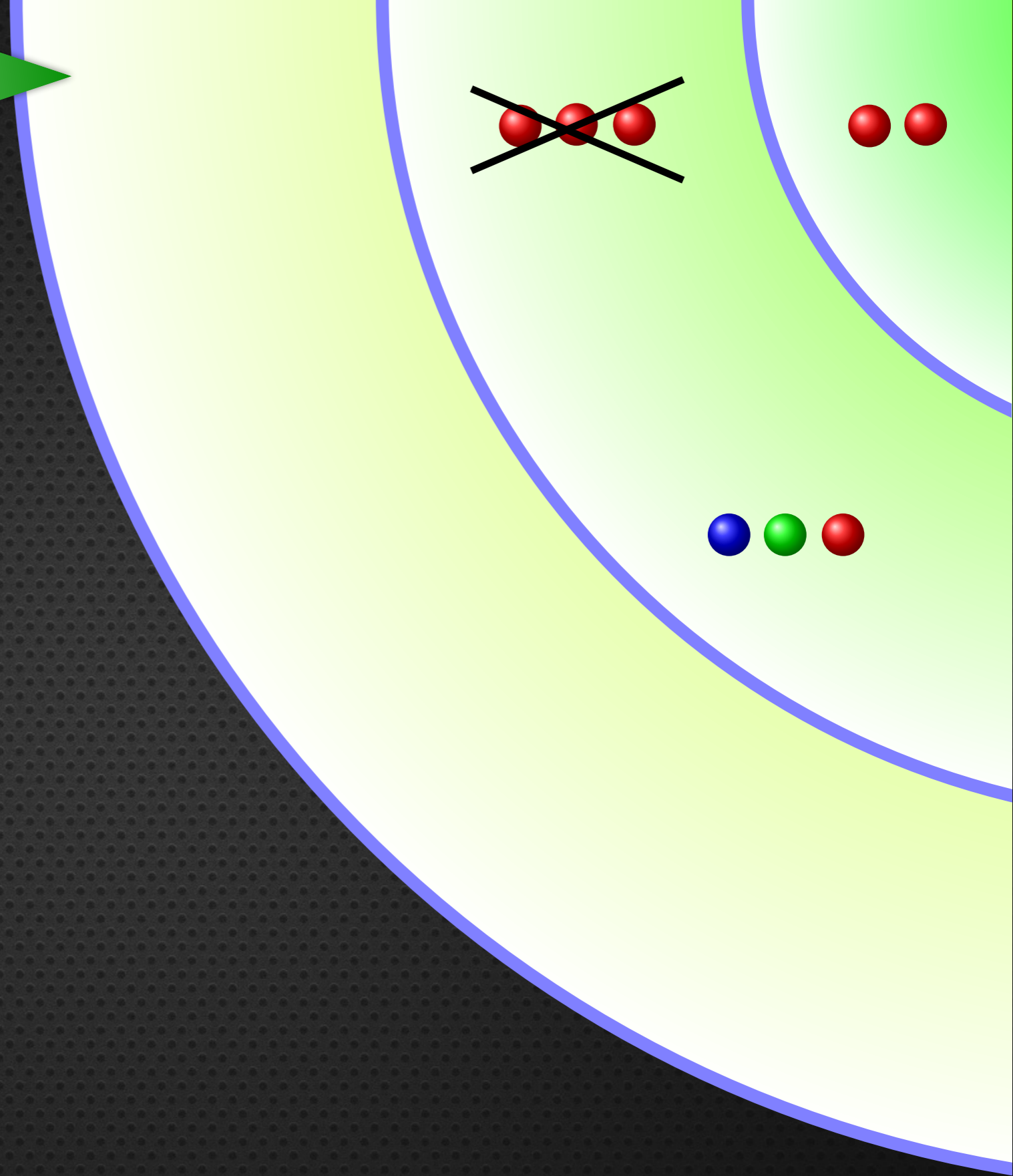
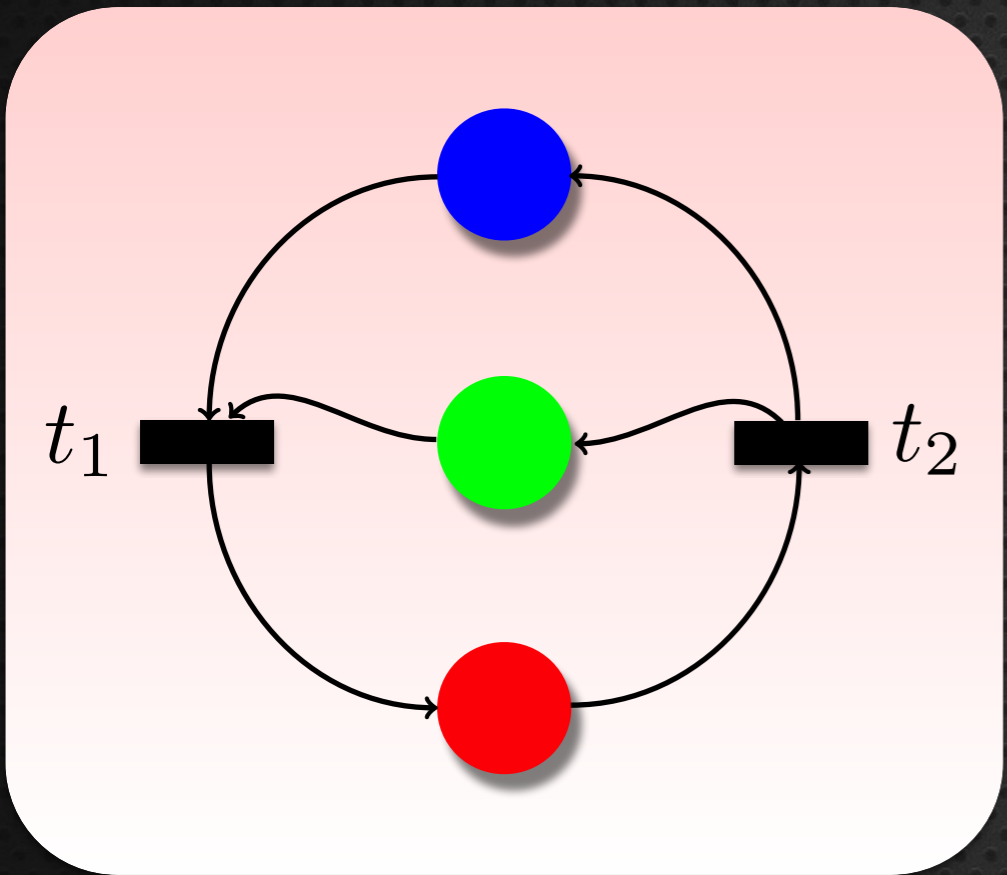
Petri Nets

Backward Reachability

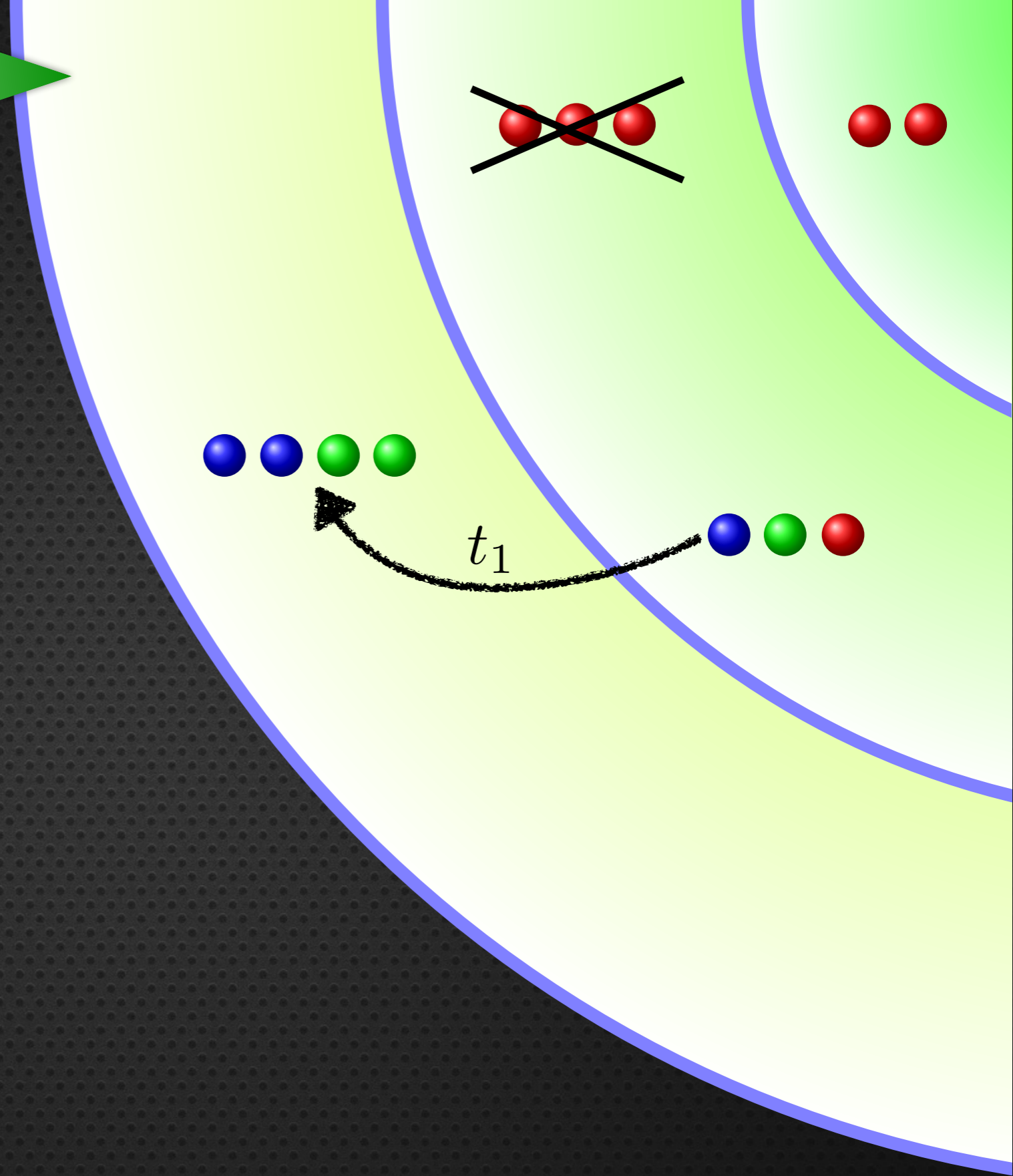
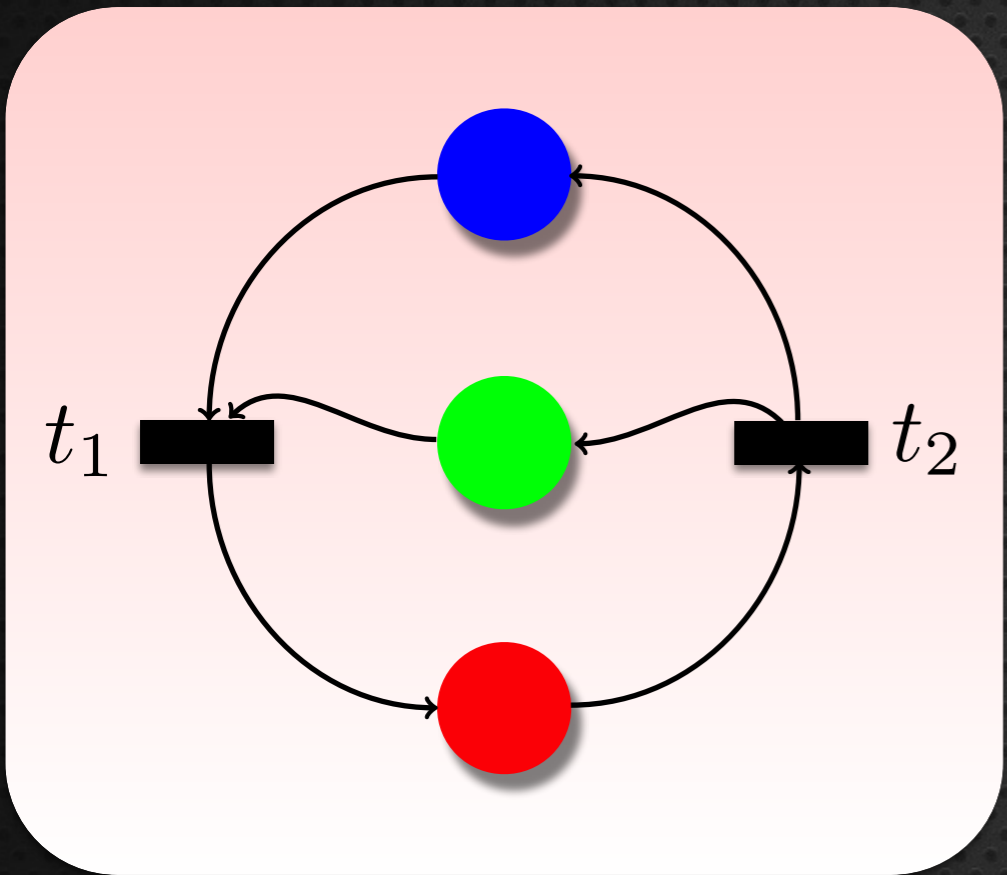


Petri Nets

Backward Reachability

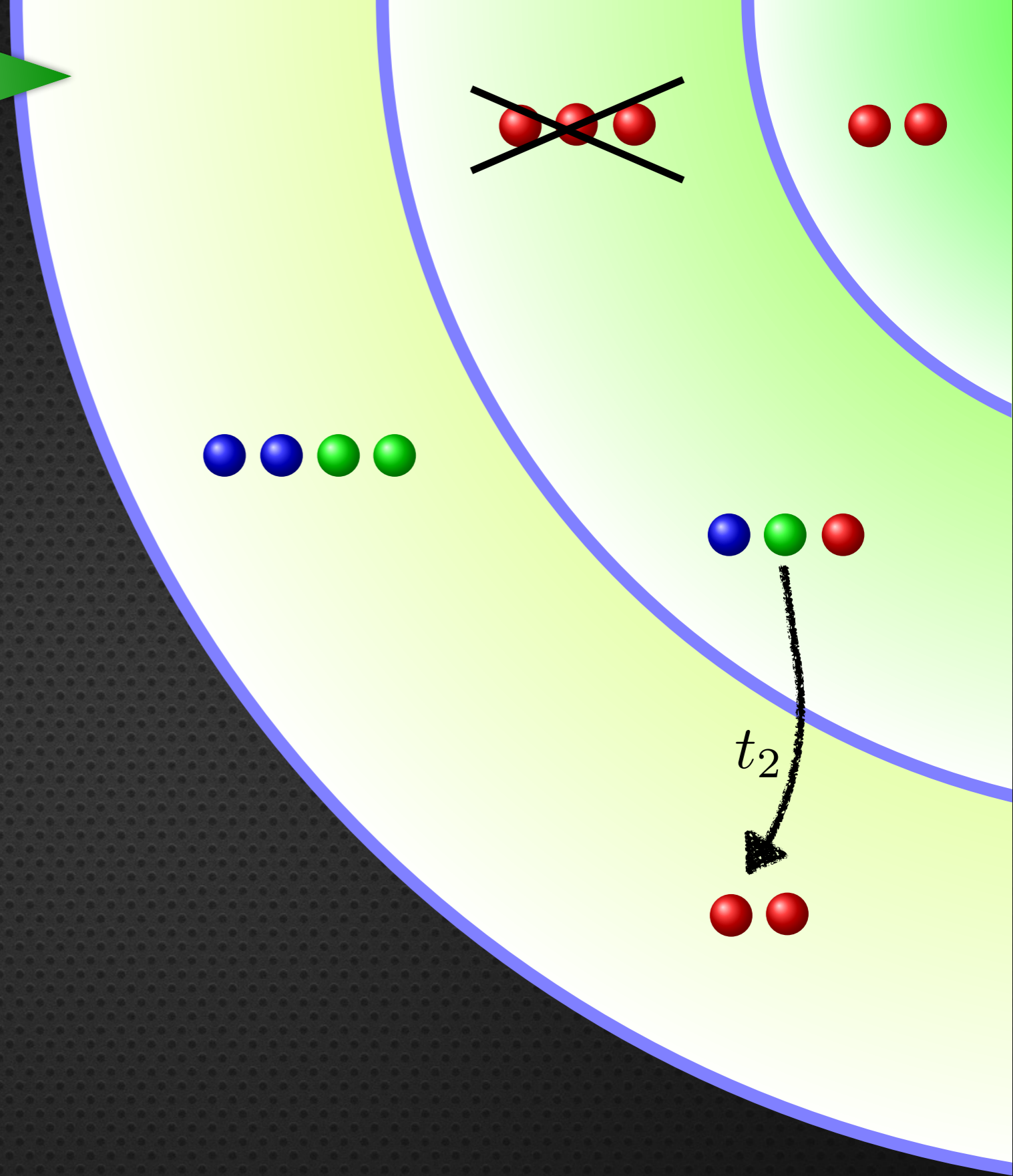
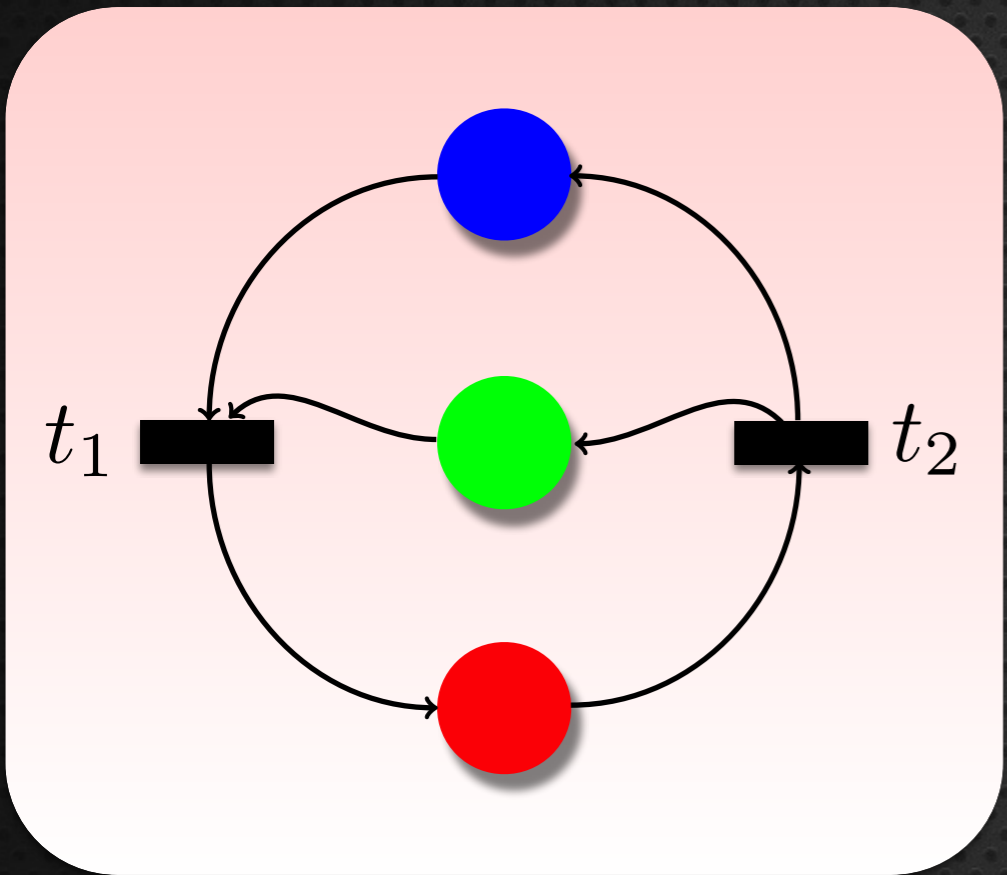


Petri Net Backward Reachability



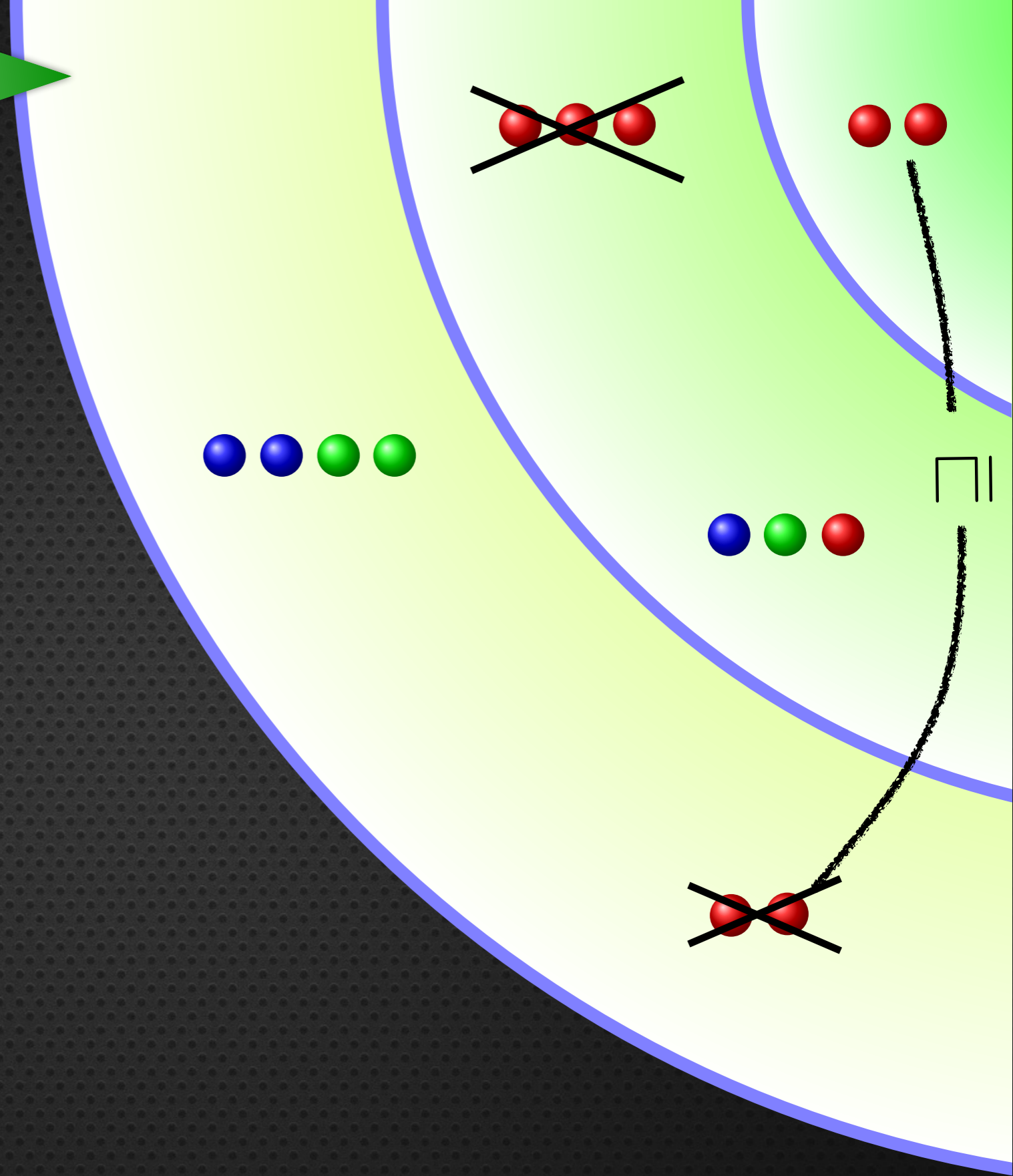
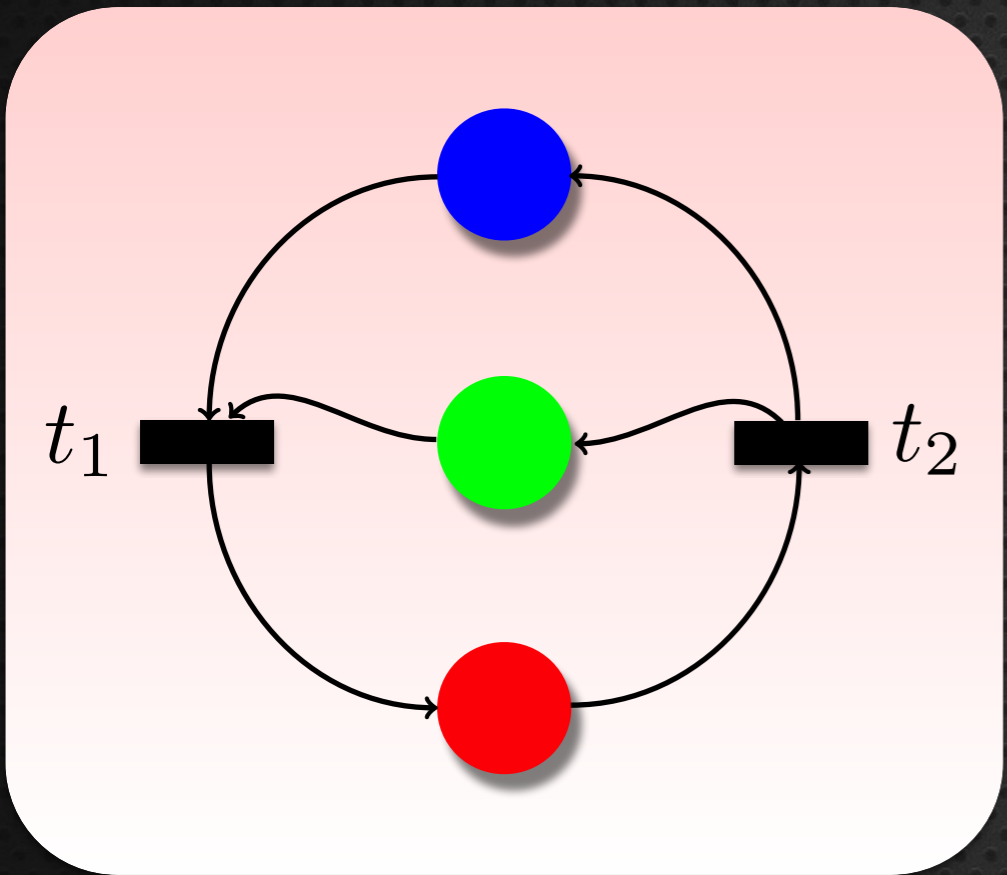
Petri Nets

Backward Reachability

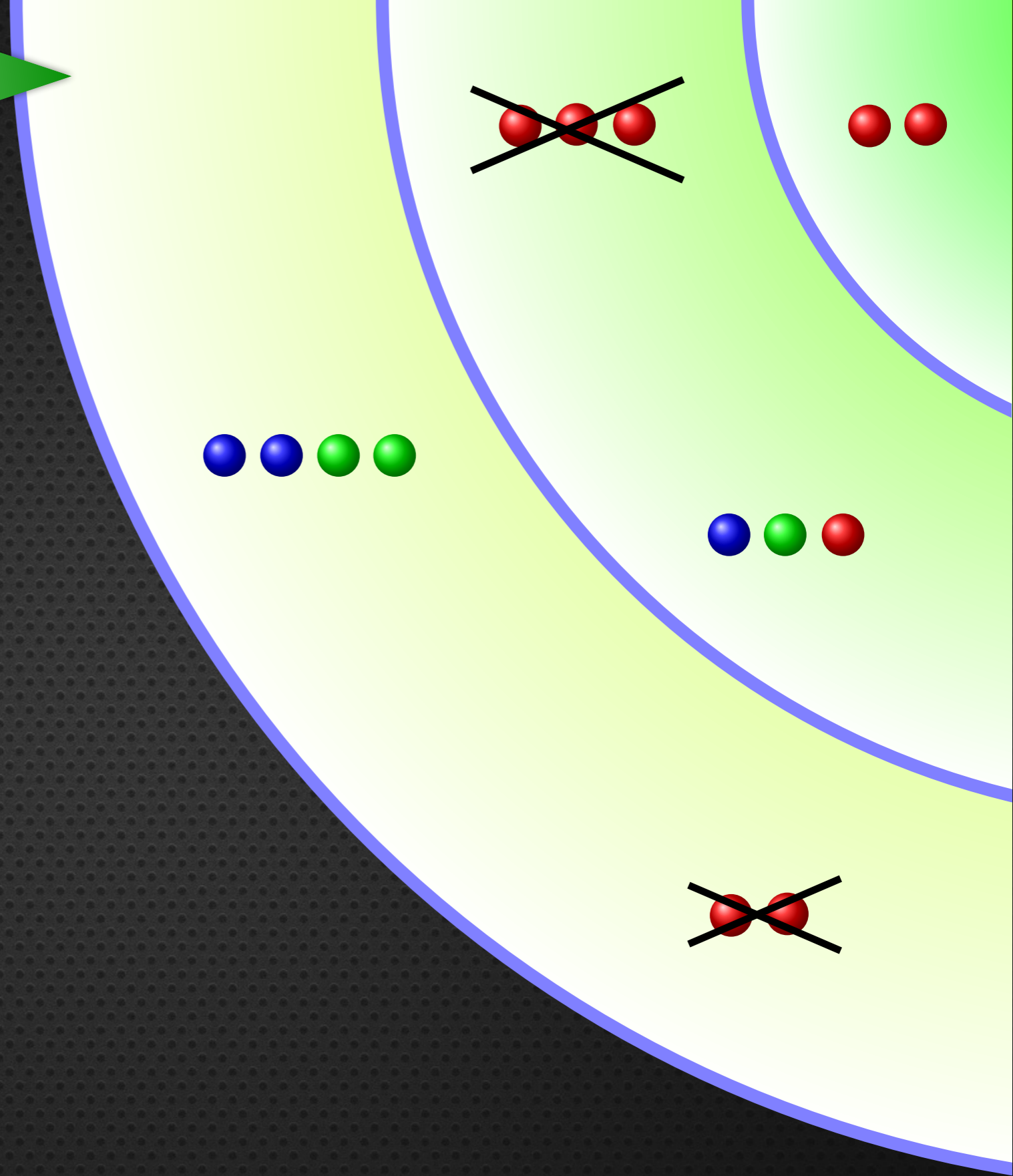
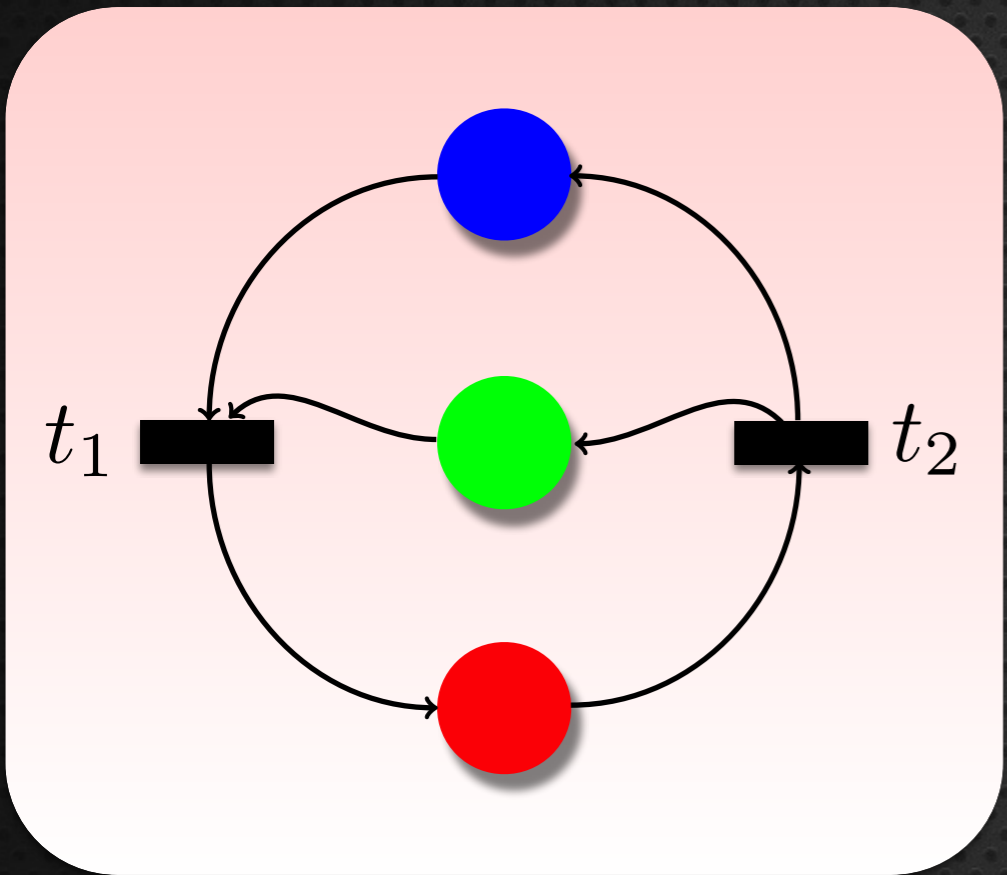


Petri Nets

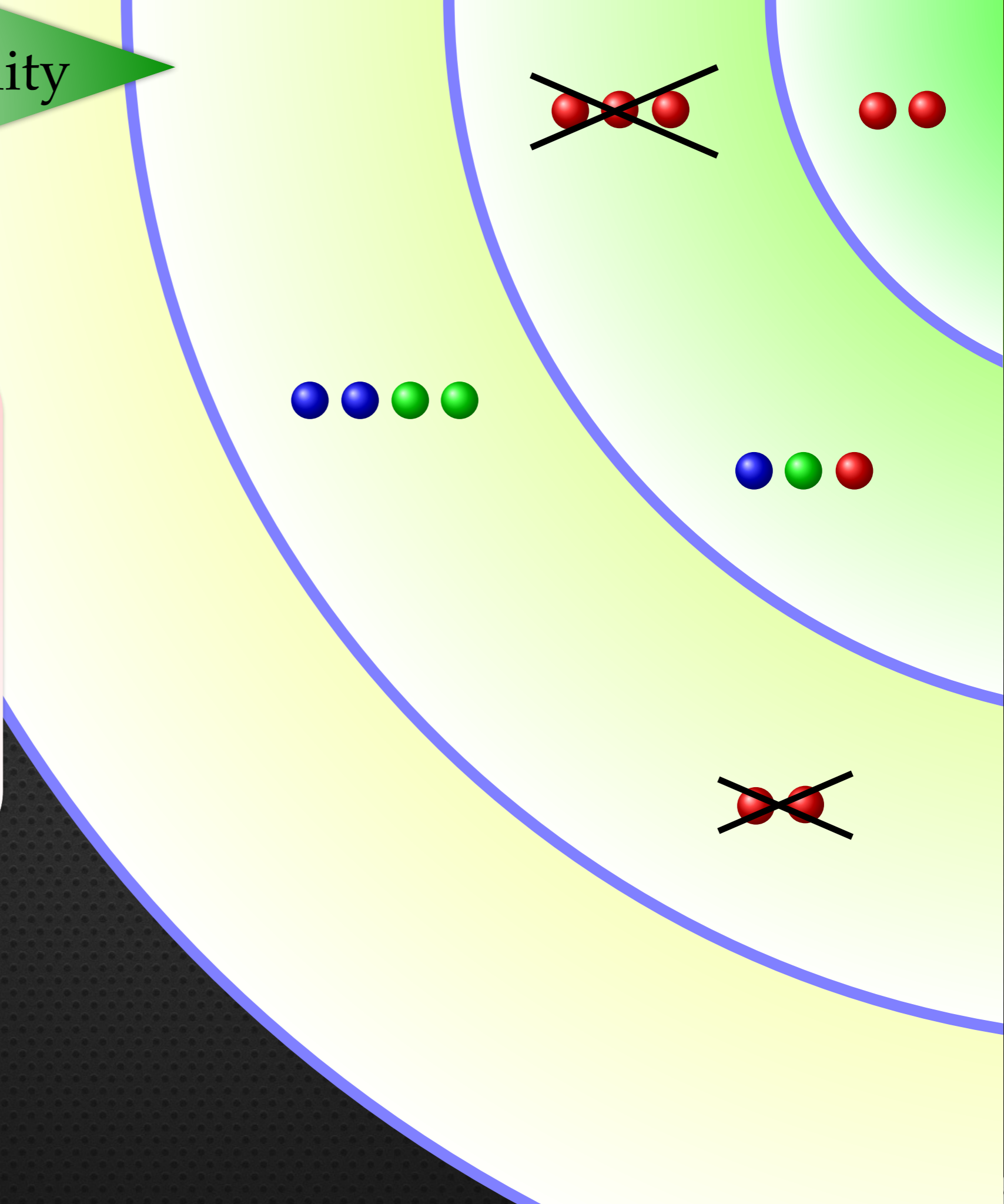
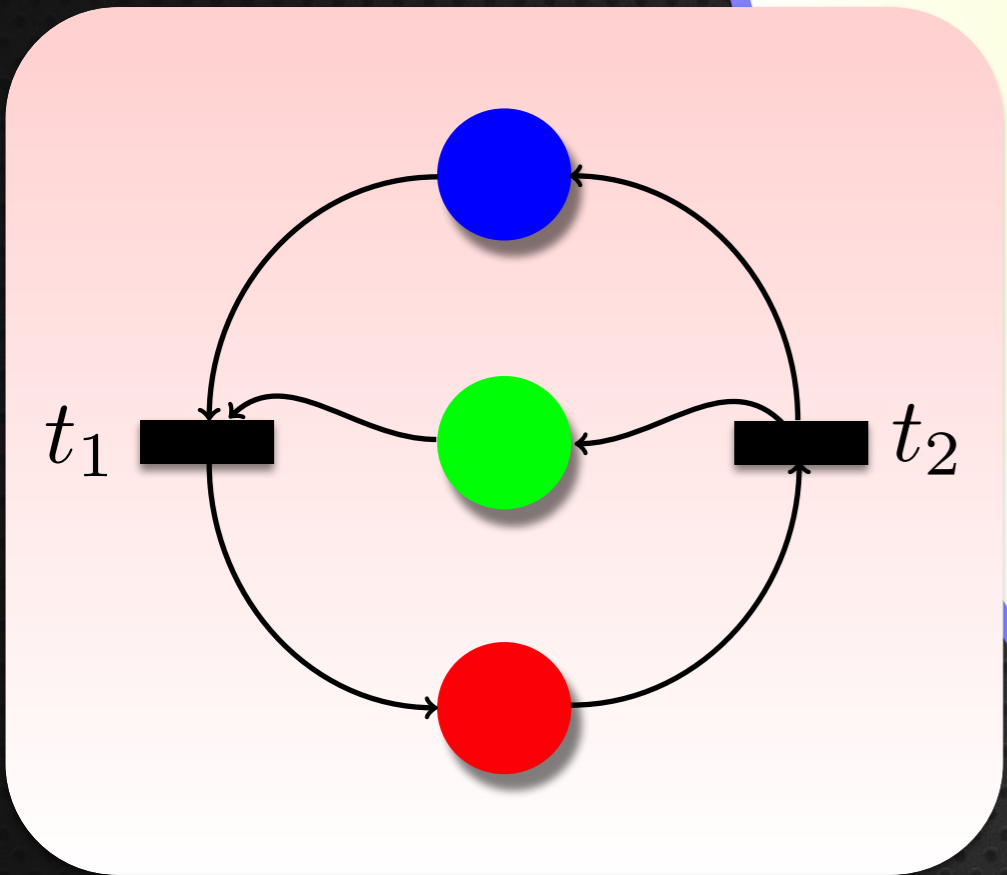
Backward Reachability



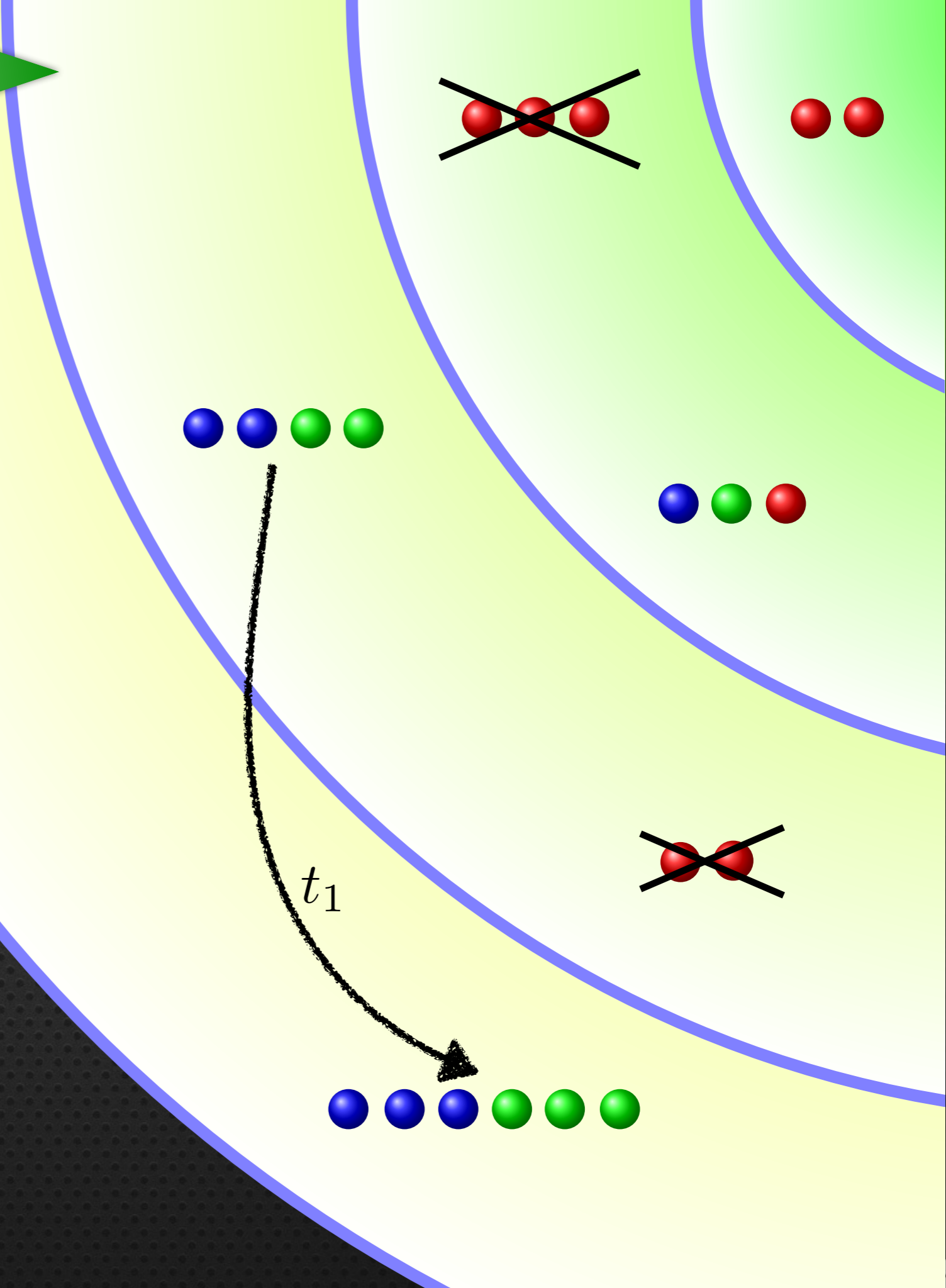
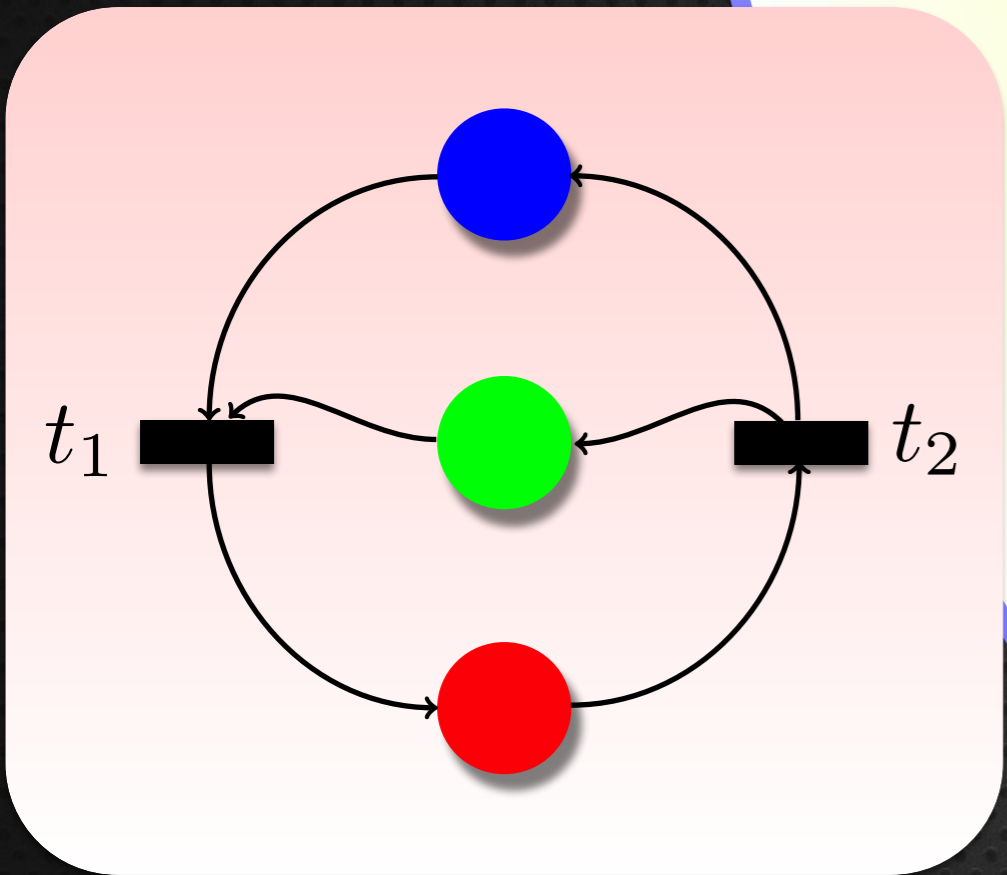
Petri Net Backward Reachability



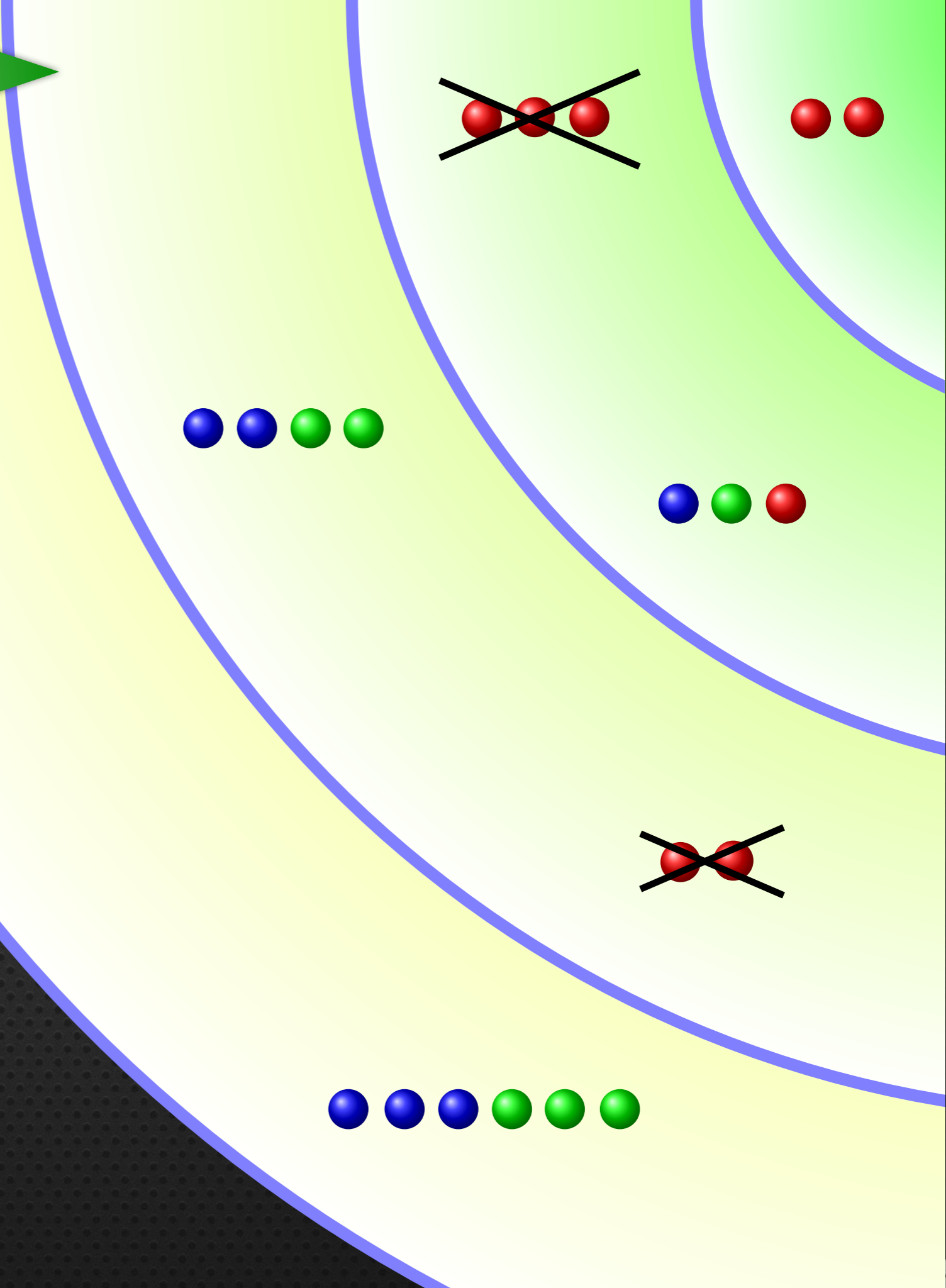
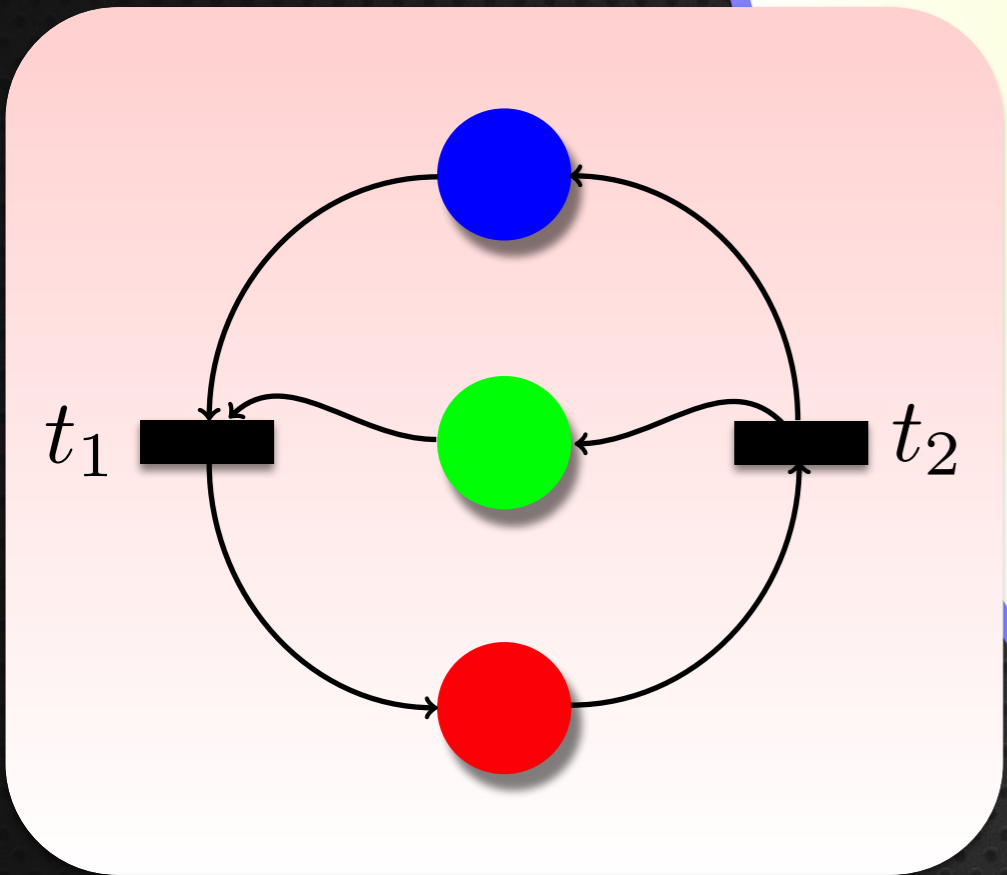
Petri Net Backward Reachability



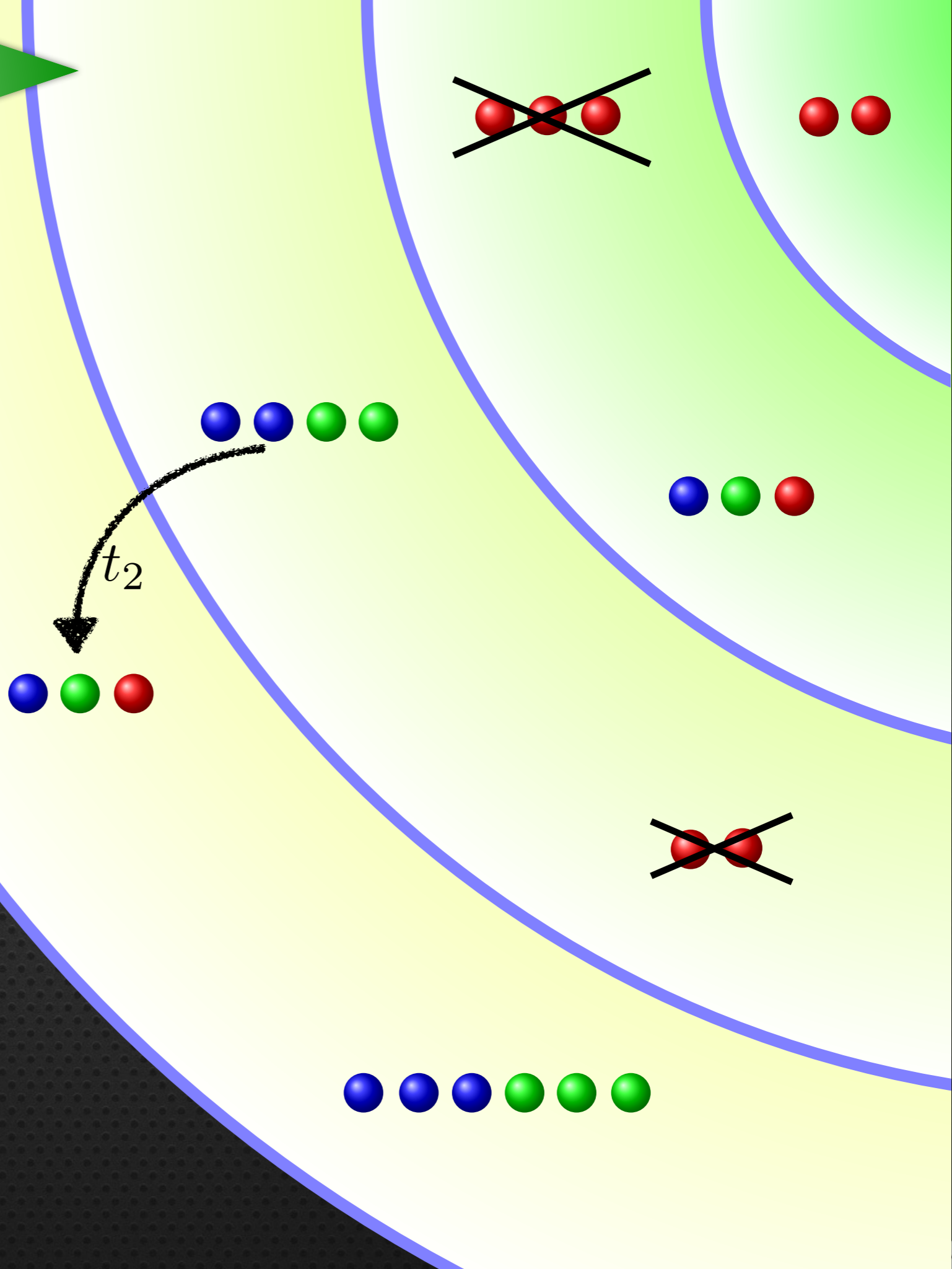
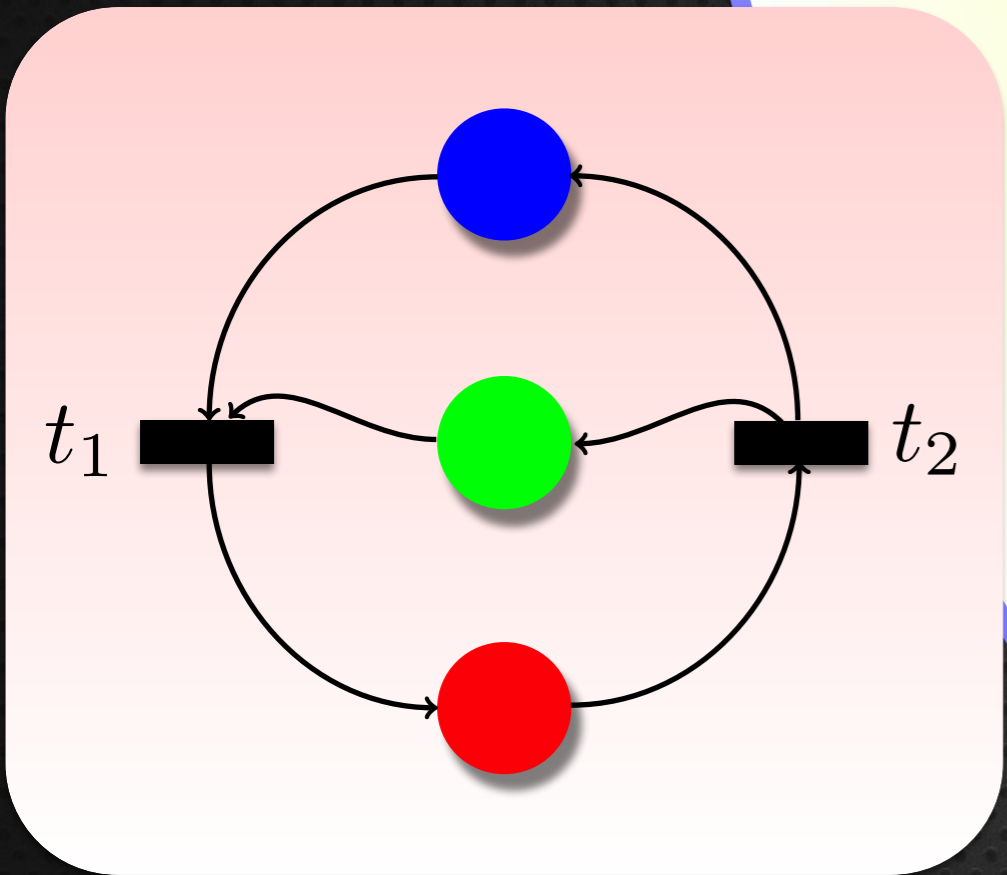
Petri Net Backward Reachability



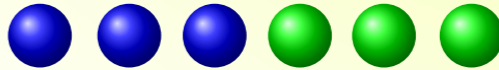
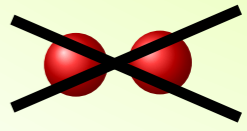
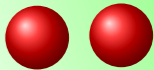
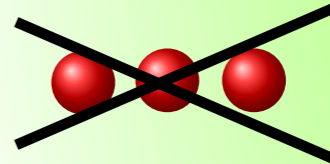
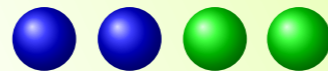
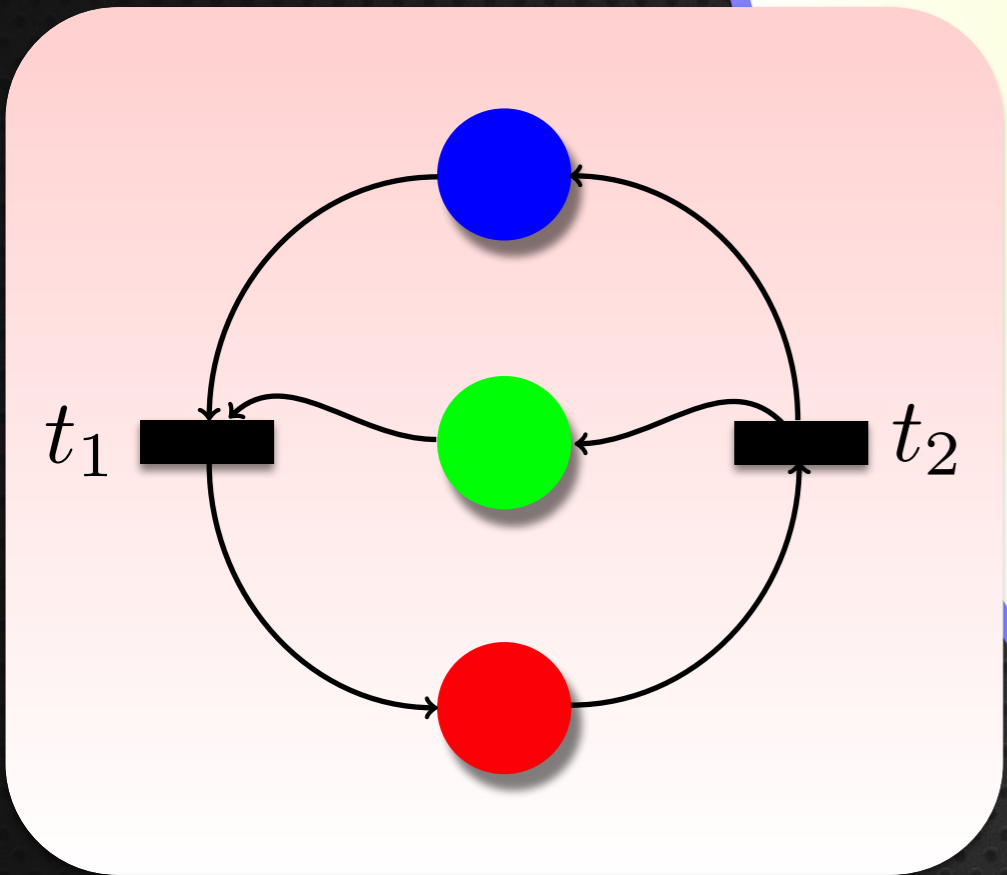
Petri Net Backward Reachability



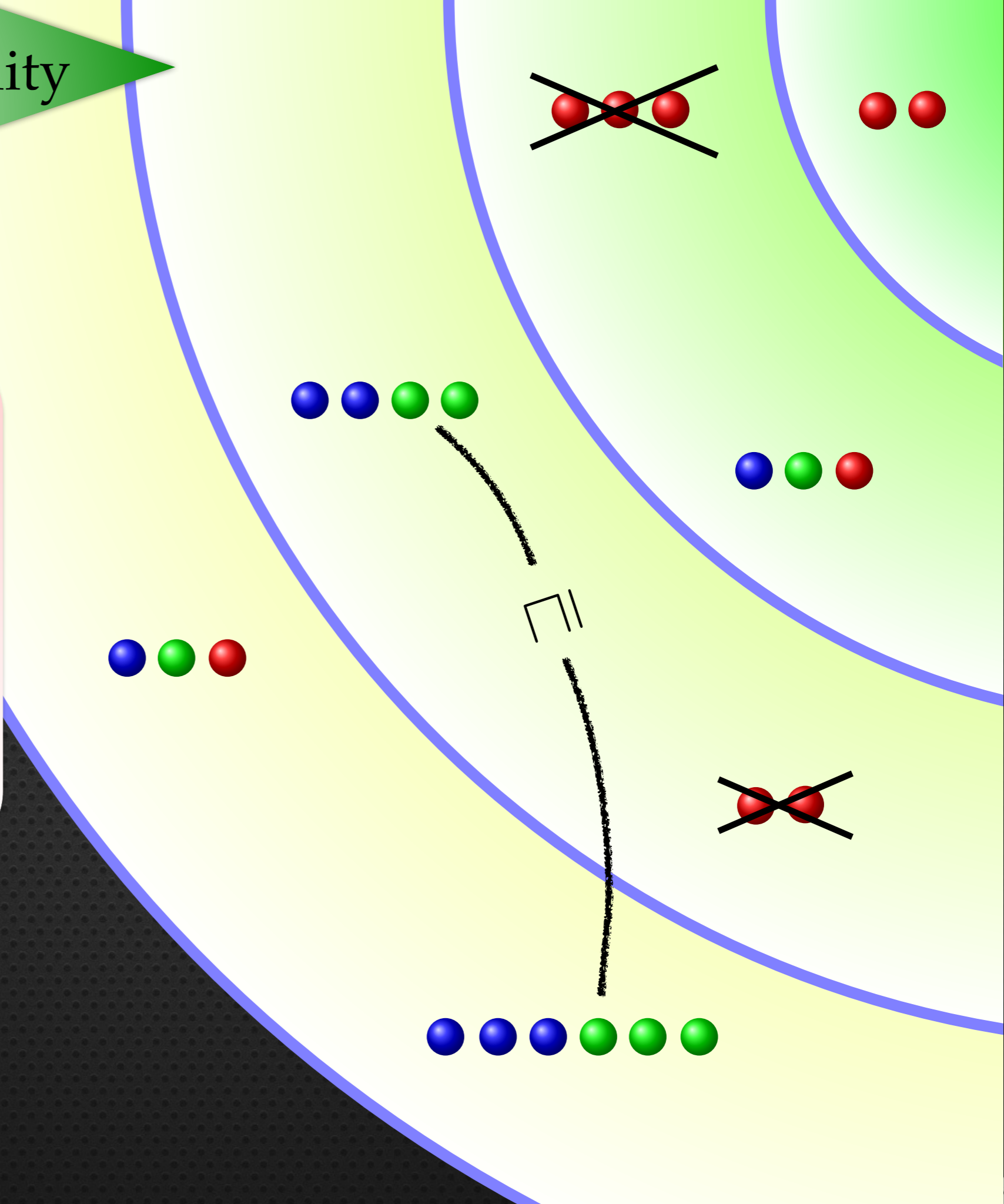
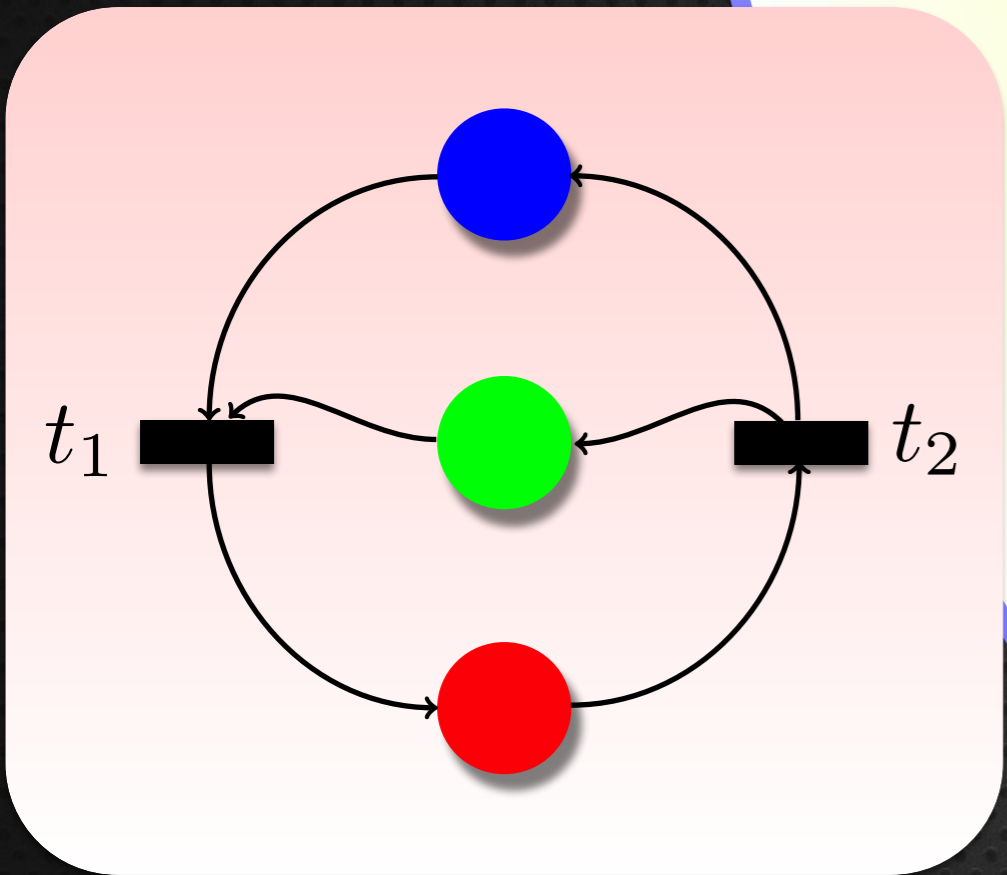
Petri Net Backward Reachability



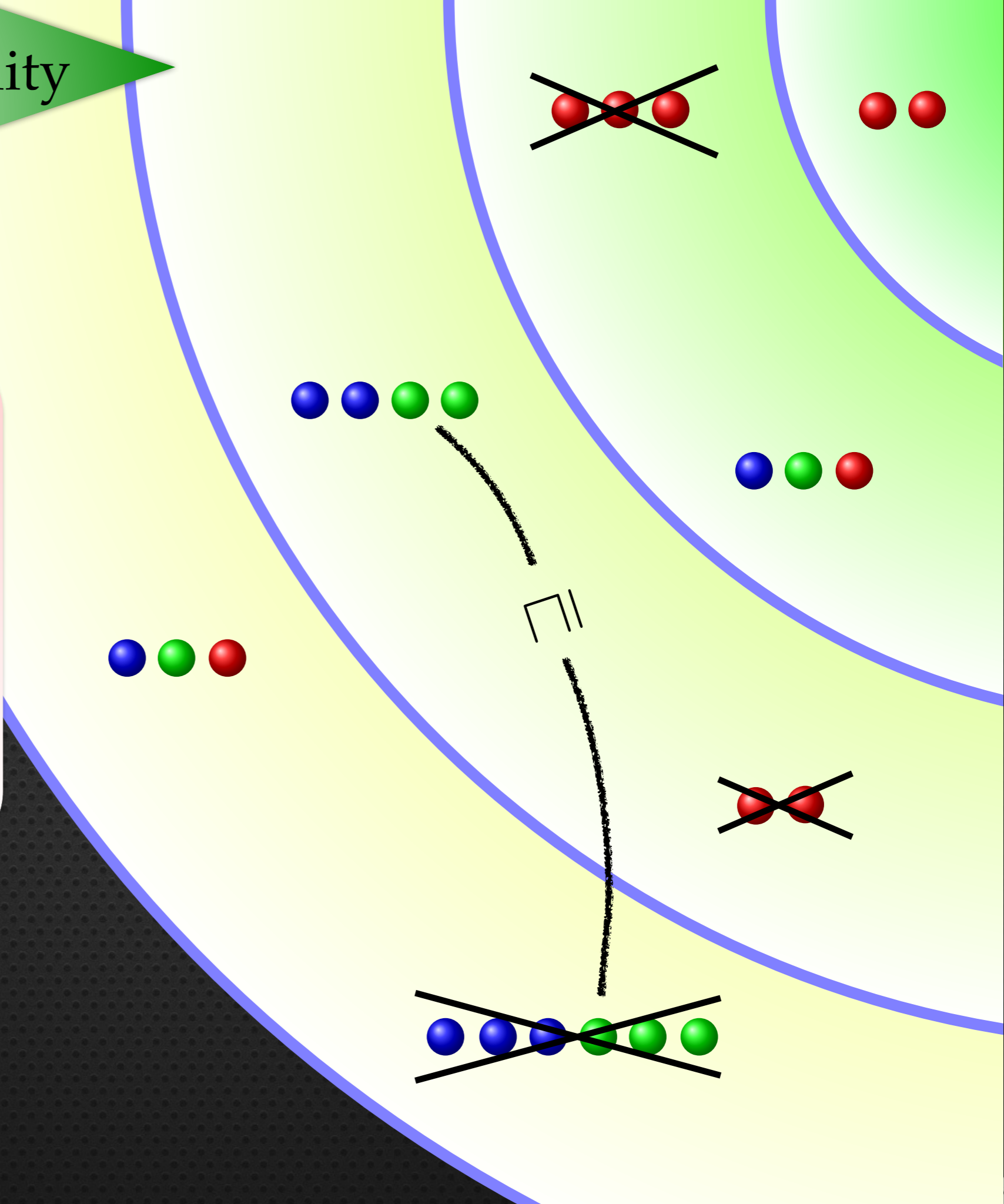
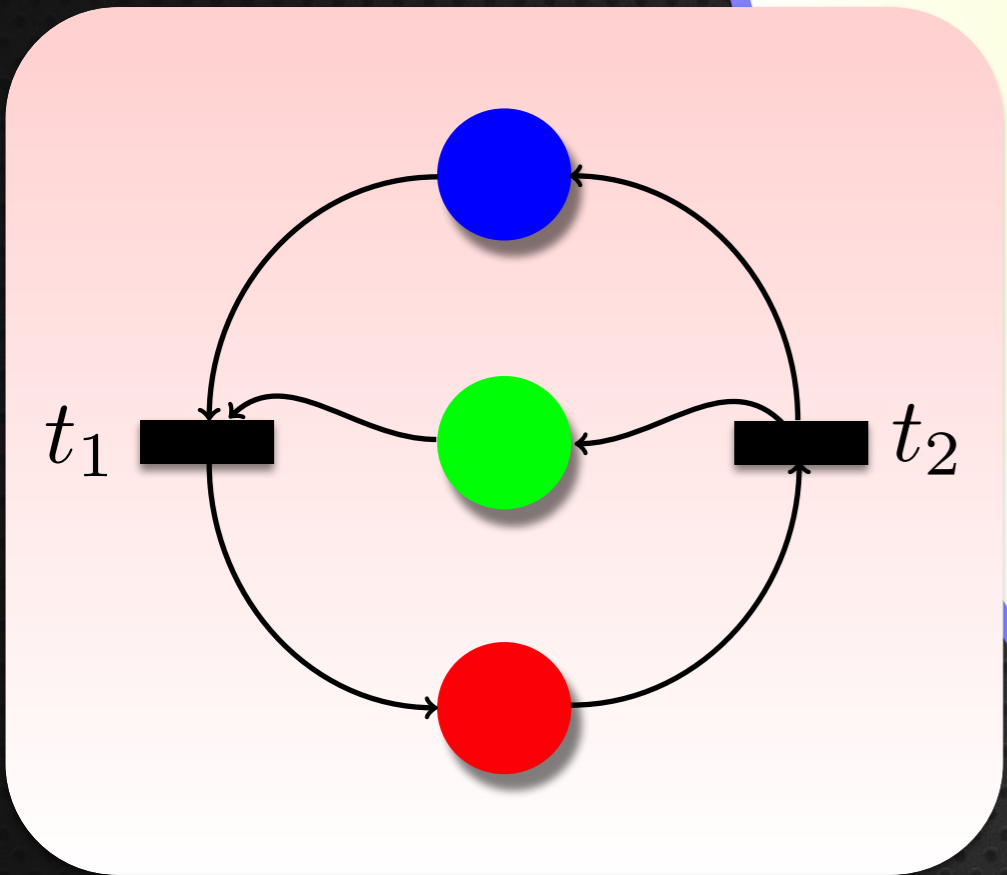
Petri Net Backward Reachability



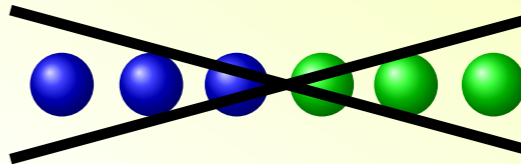
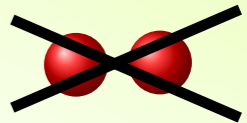
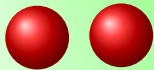
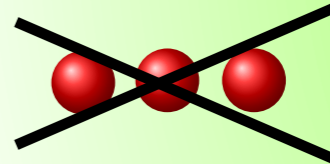
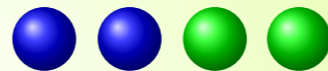
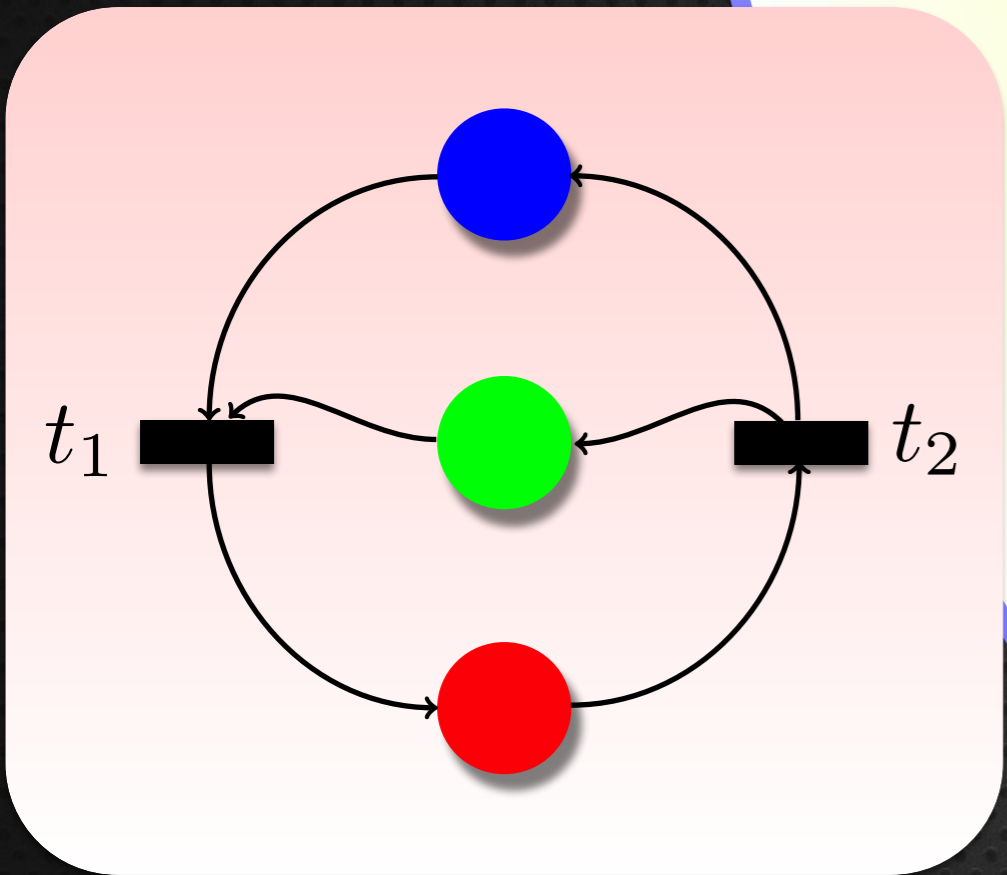
Petri Net Backward Reachability



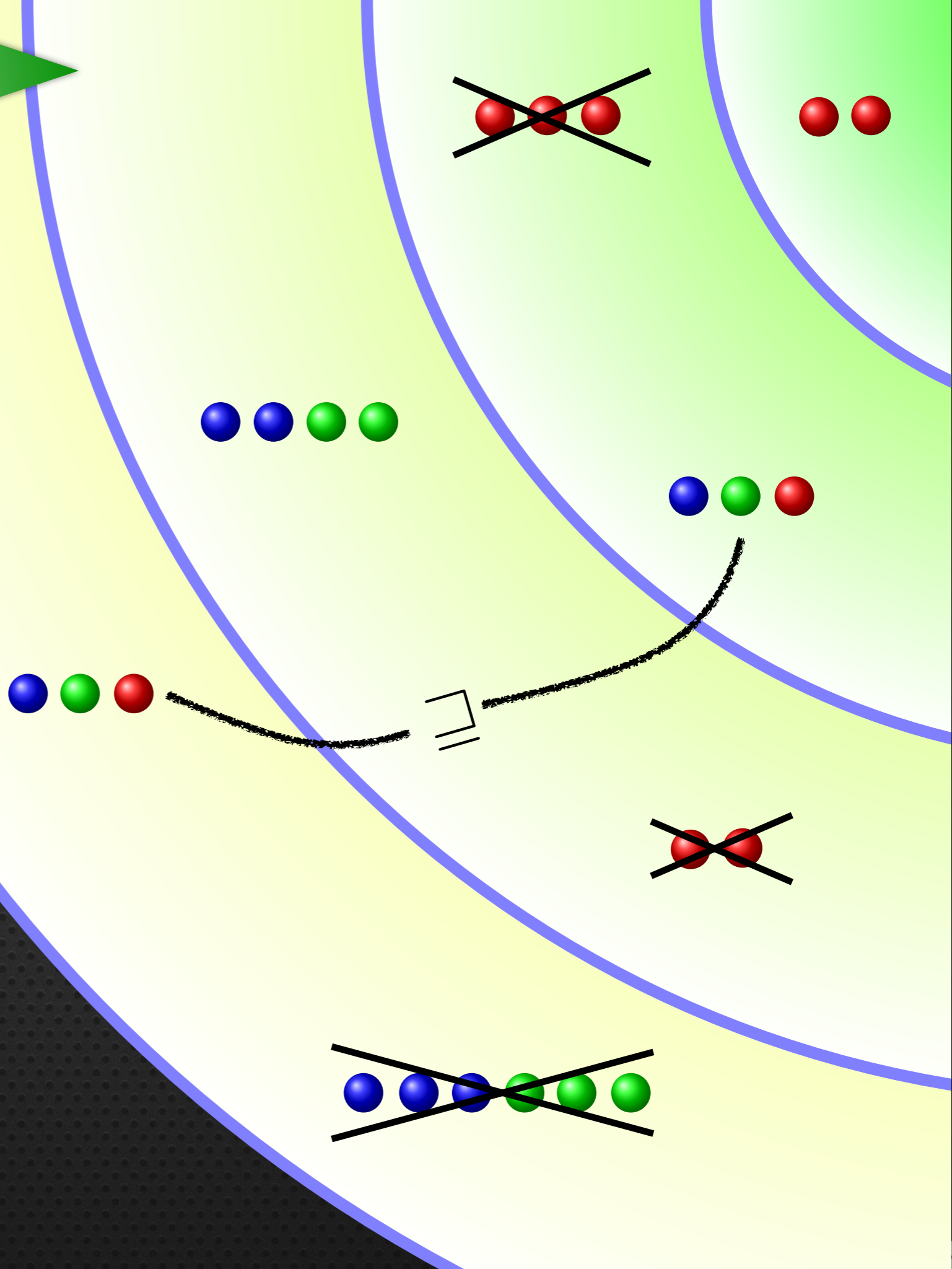
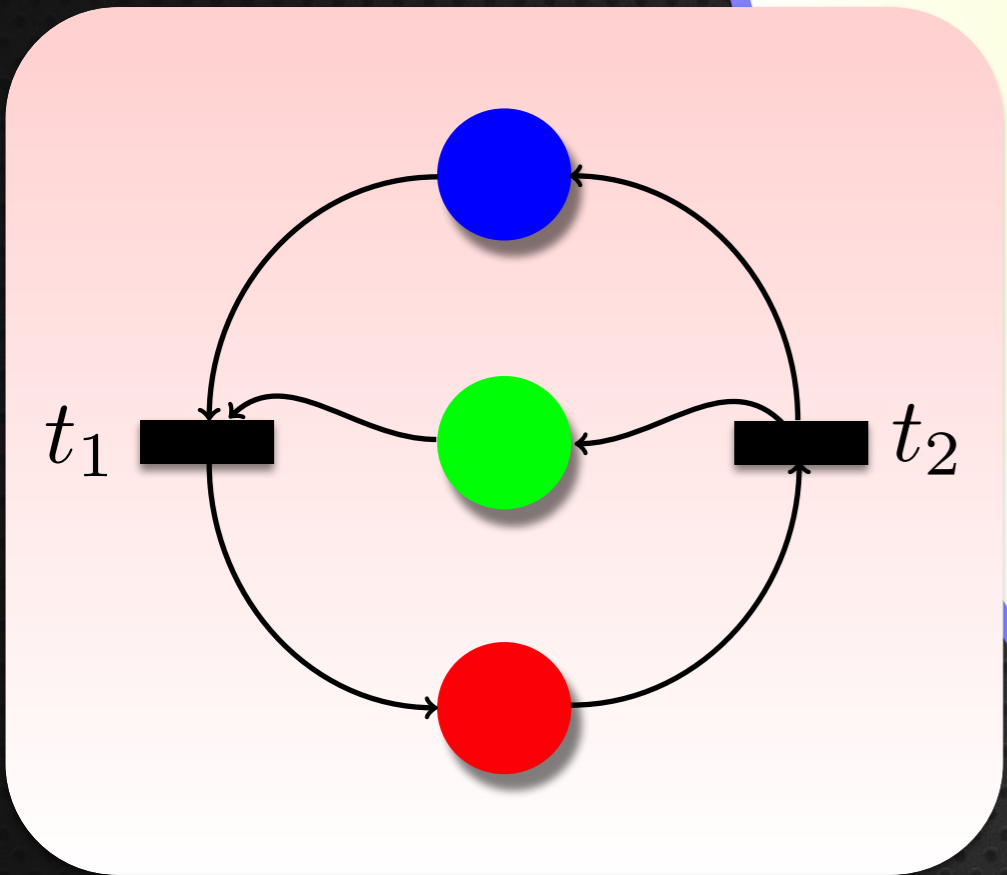
Petri Net Backward Reachability



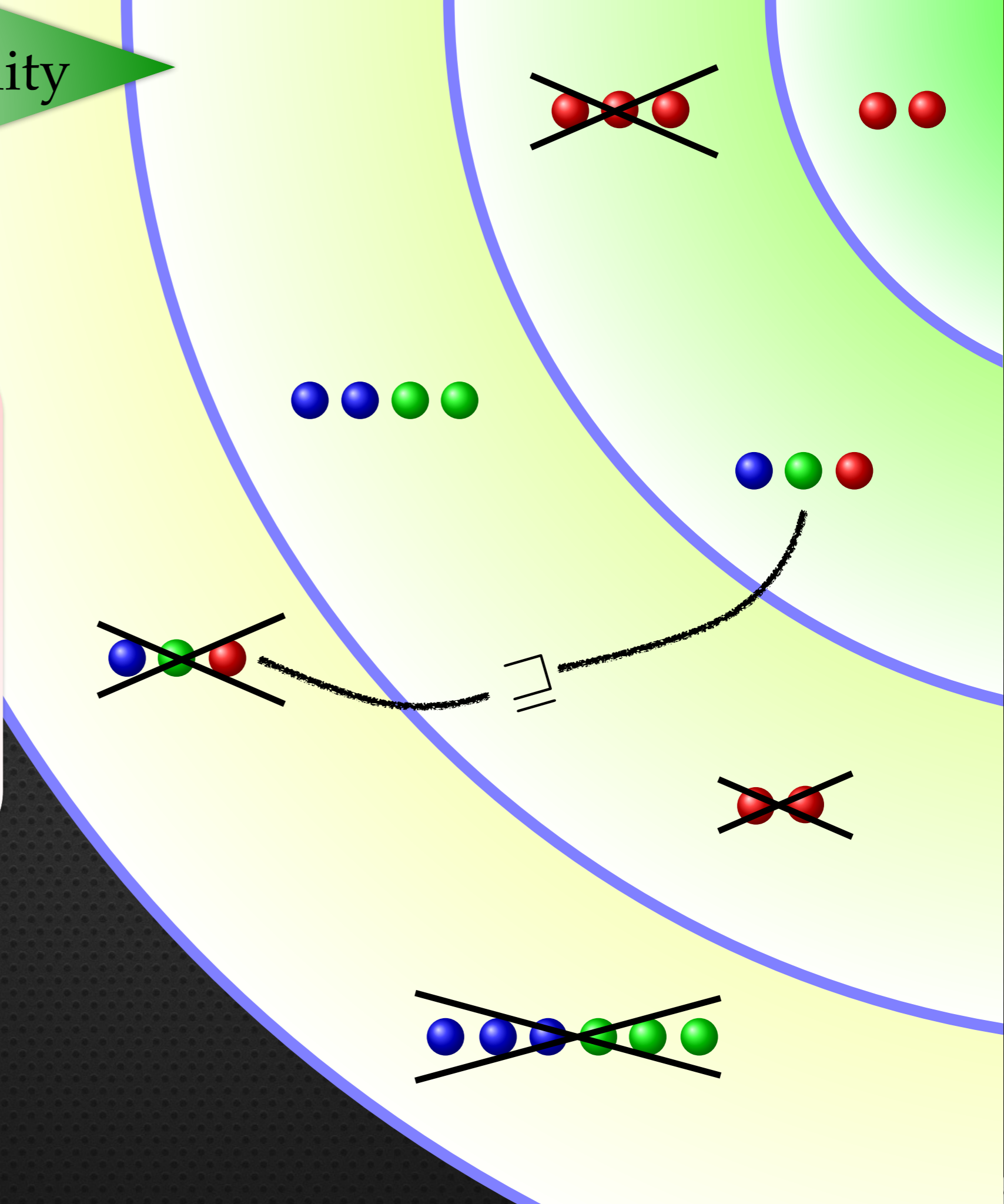
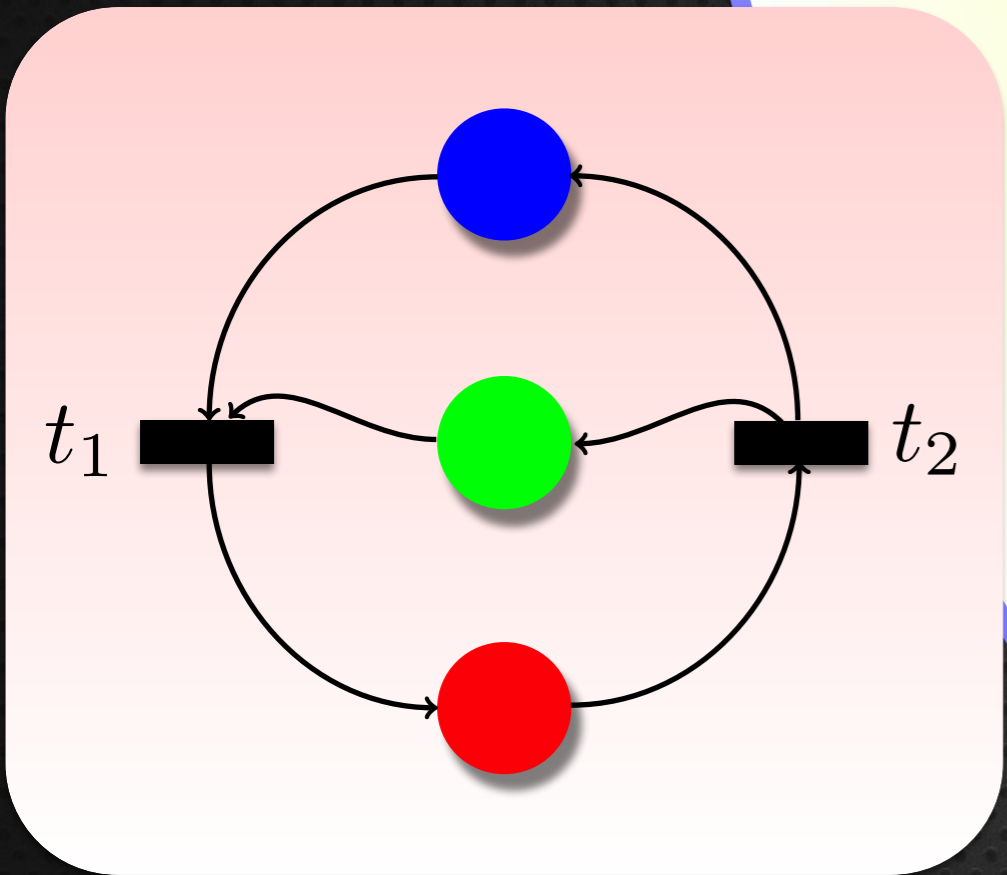
Petri Net Backward Reachability



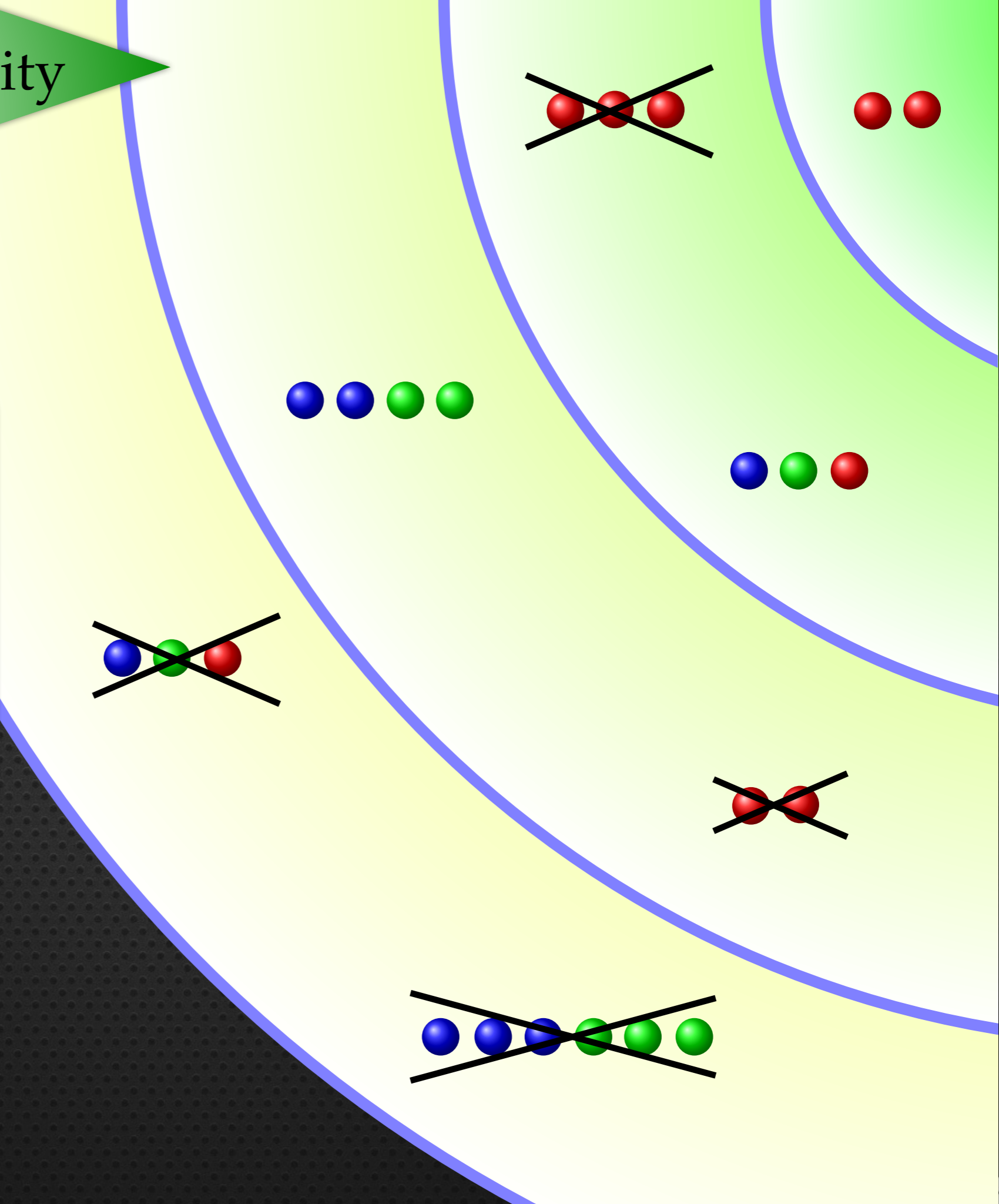
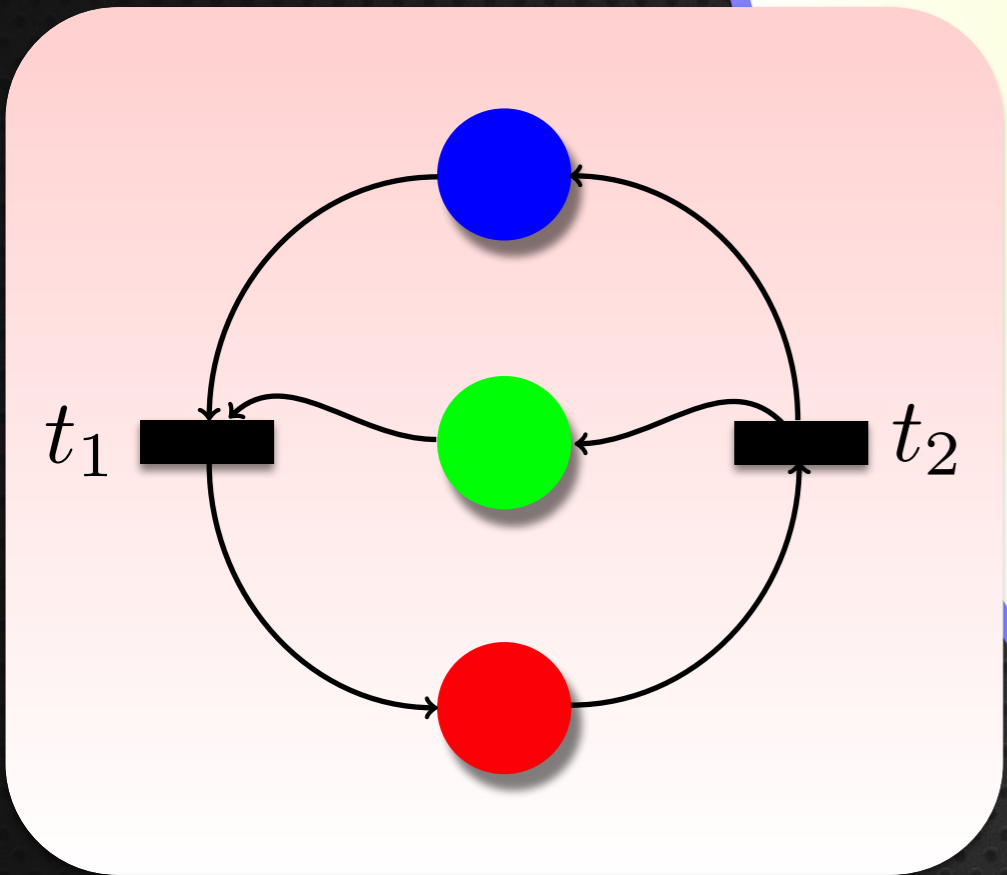
Petri Net Backward Reachability



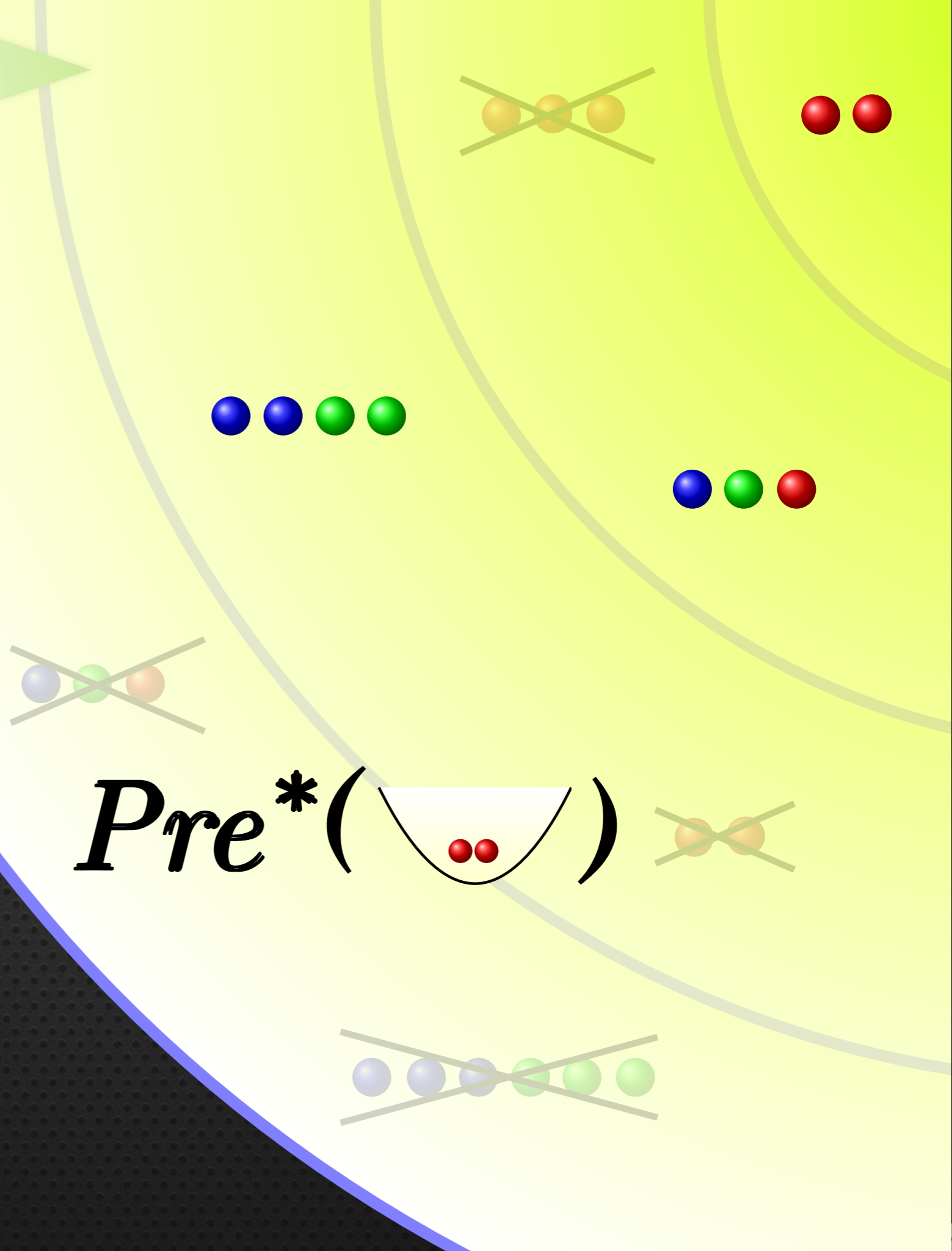
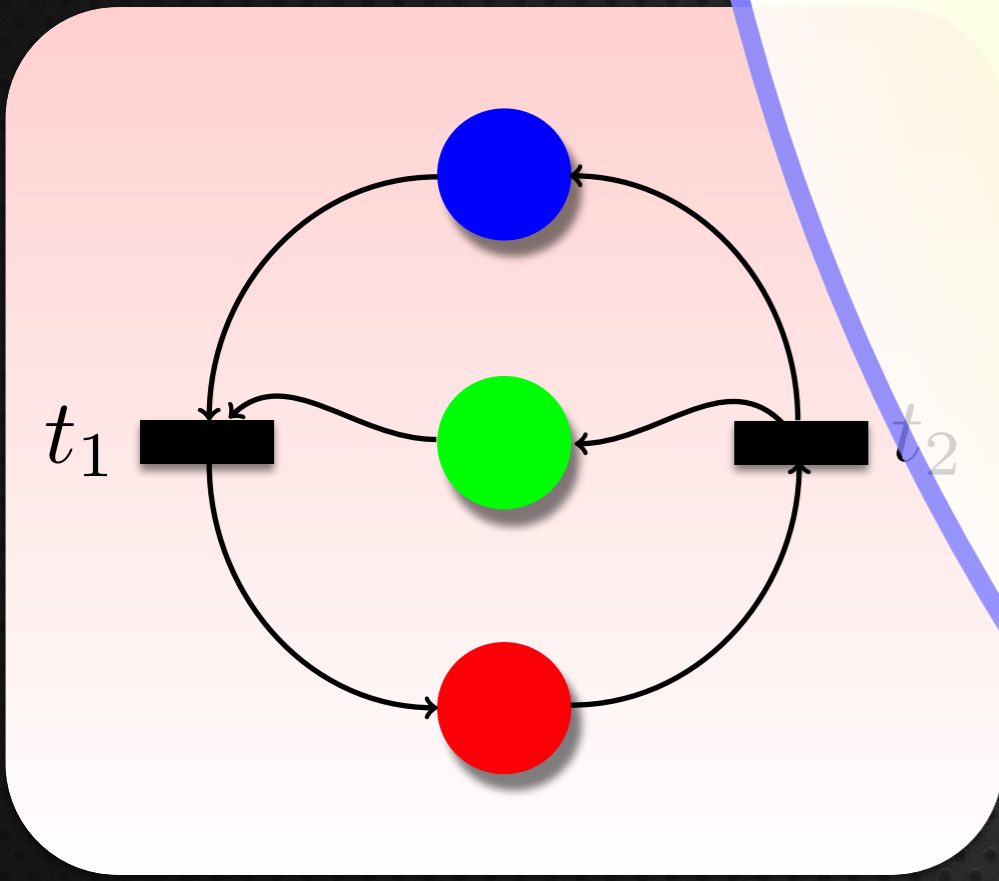
Petri Net Backward Reachability



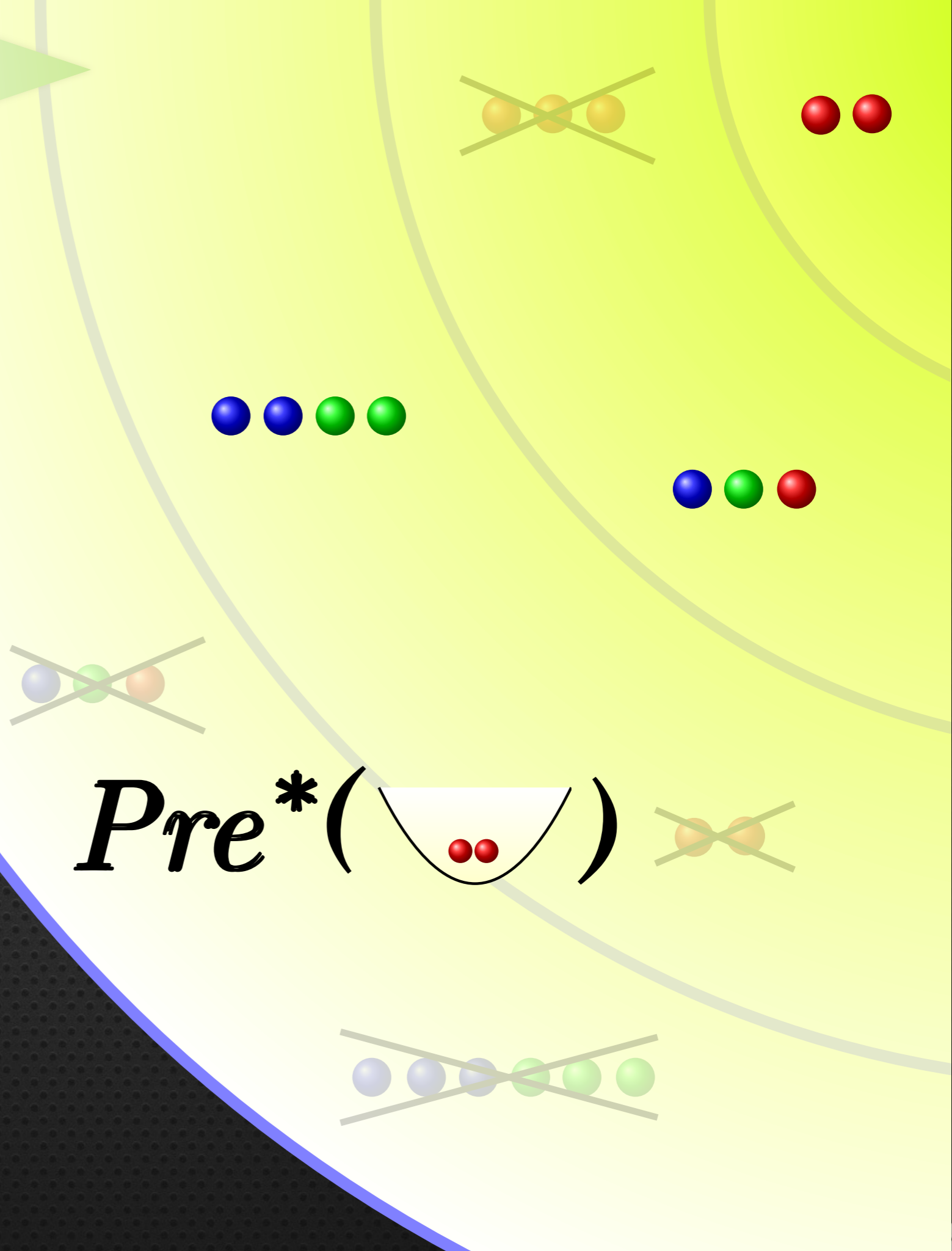
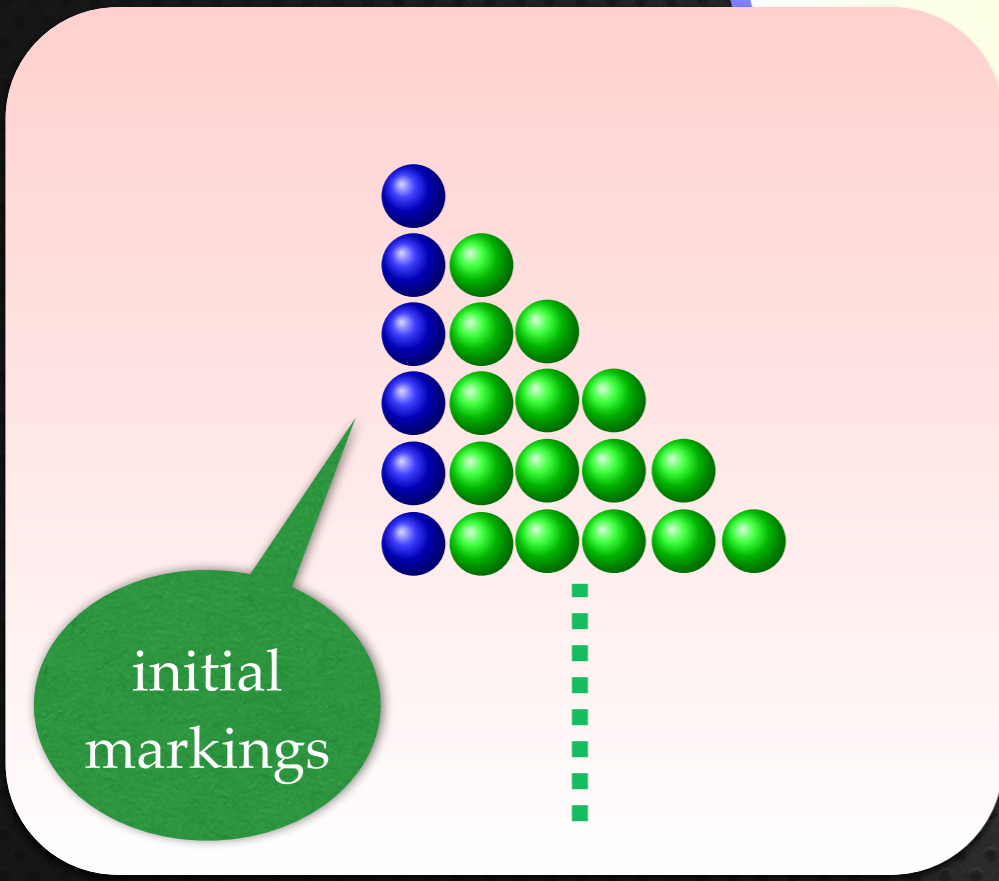
Petri Net Backward Reachability



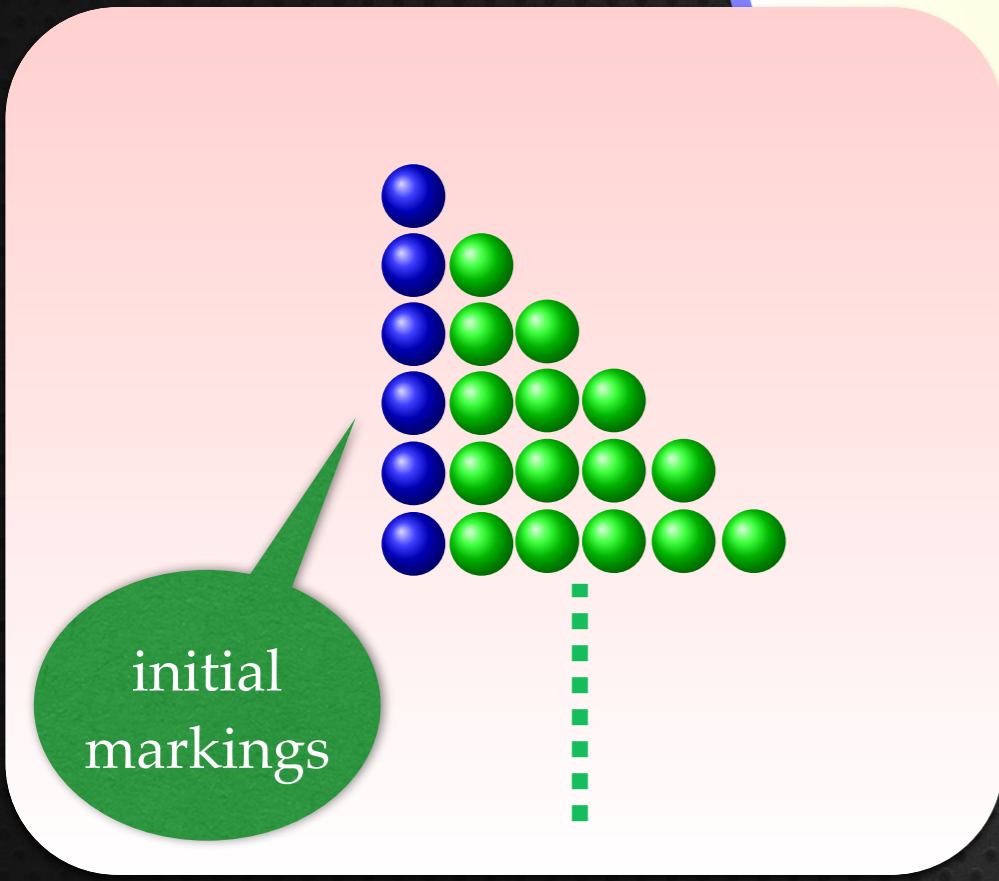
Petri Net Backward Reachability

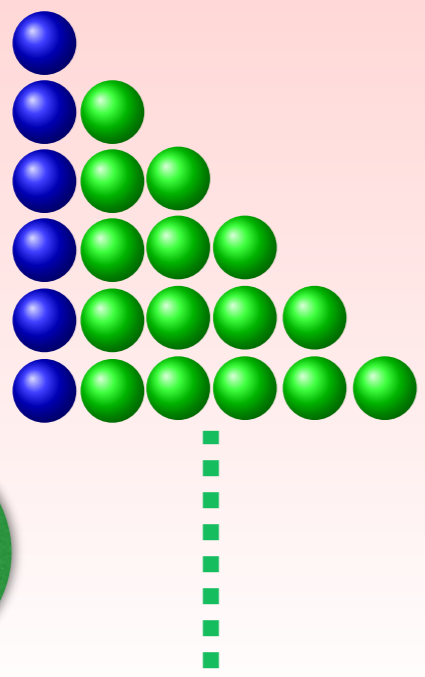


Petri Net Backward Reachability



Petri Net Backward Reachability



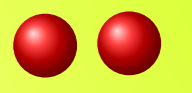
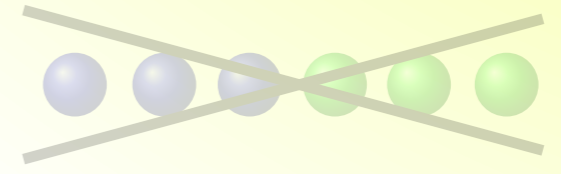


initial markings

System Safe !

symbolic representation = finite multisets

$$Pre^* \left(\text{Petri Net Diagram} \right)$$

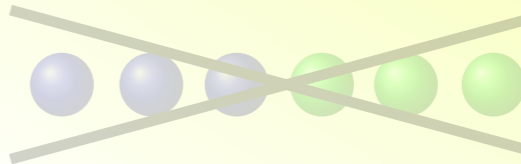
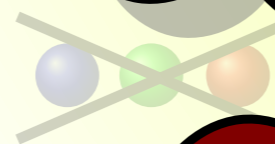
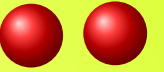
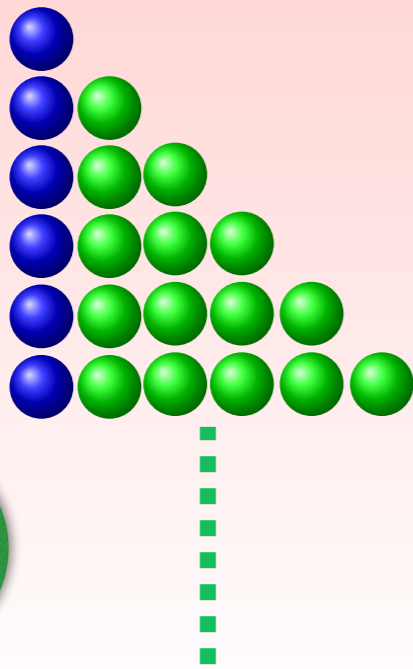


System Safe !

symbolic representation = finite multisets

Termination: multisets well quasi-ordered

initial
markings



Petri

Well Quasi-Ordering

Well Quasi-Ordering

infinite sequence of markings

$m_0, m_1, m_2, \dots, m_i, \dots, m_j, \dots$

Well Quasi-Ordering

infinite sequence of markings

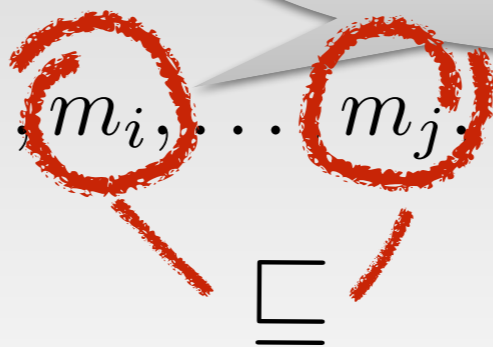
$m_0, m_1, m_2, \dots, m_i, \dots, m_j, \dots$

$$\exists i < j : m_i \sqsubseteq m_j$$

Well Quasi-Ordering

infinite sequence of markings

$m_0, m_1, m_2, \dots, m_i, \dots, m_j, \dots$



$$\exists i < j : m_i \sqsubseteq m_j$$

Petri

Backward

Termination

**Assume:
non-termination**

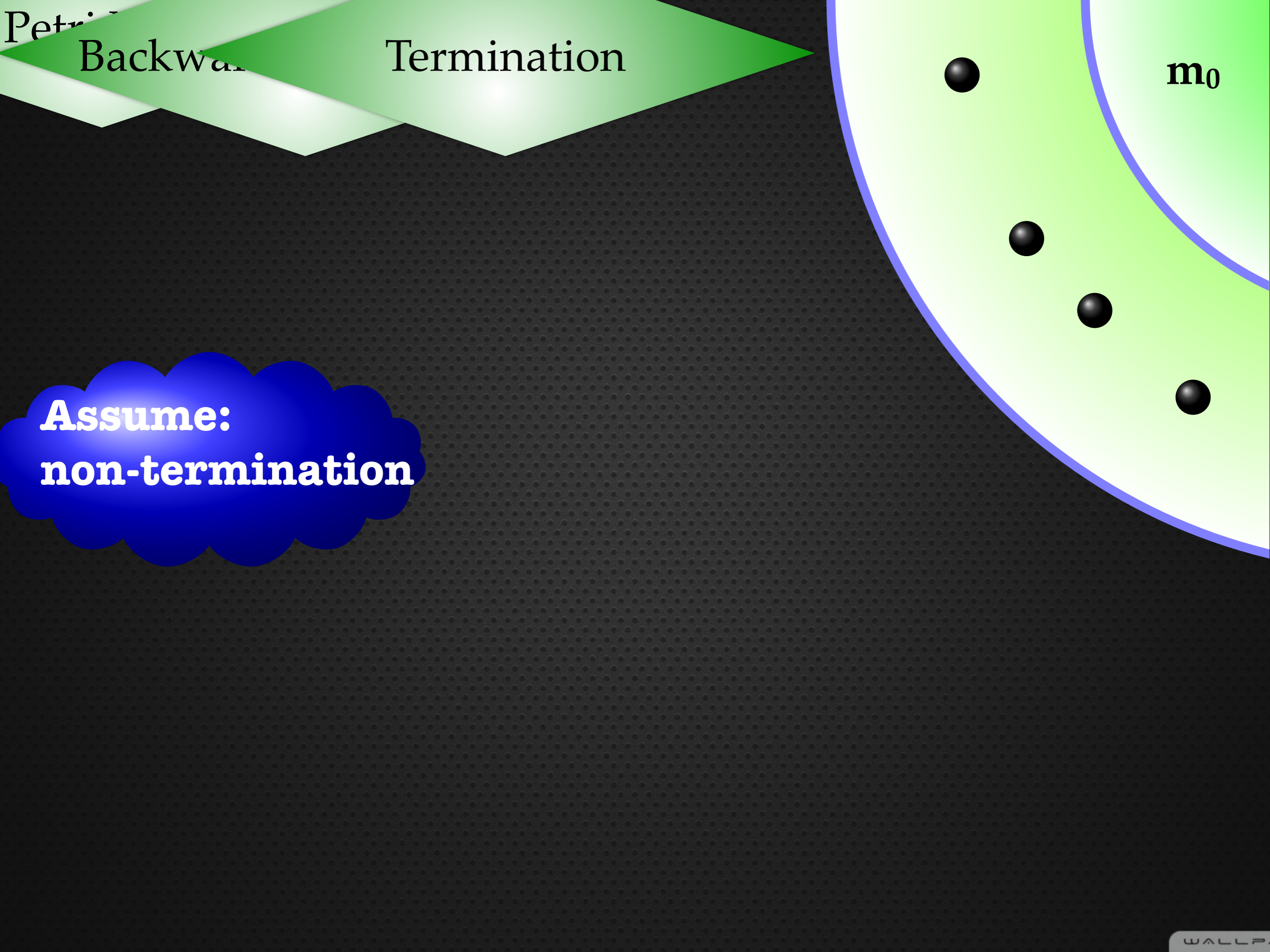
Petri

Backward

Termination

m_0

**Assume:
non-termination**

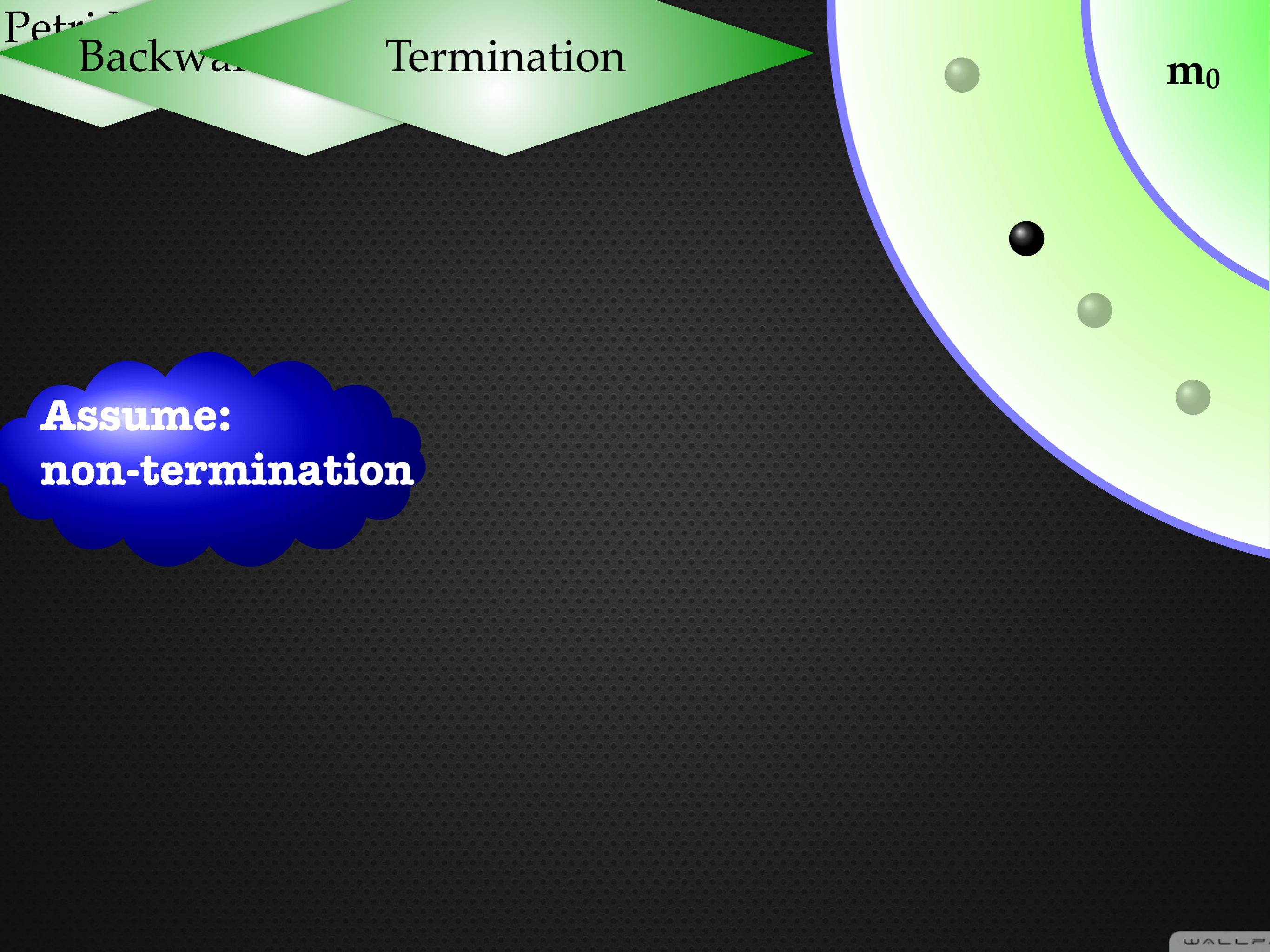


Petri
Backward

Termination

m_0

**Assume:
non-termination**

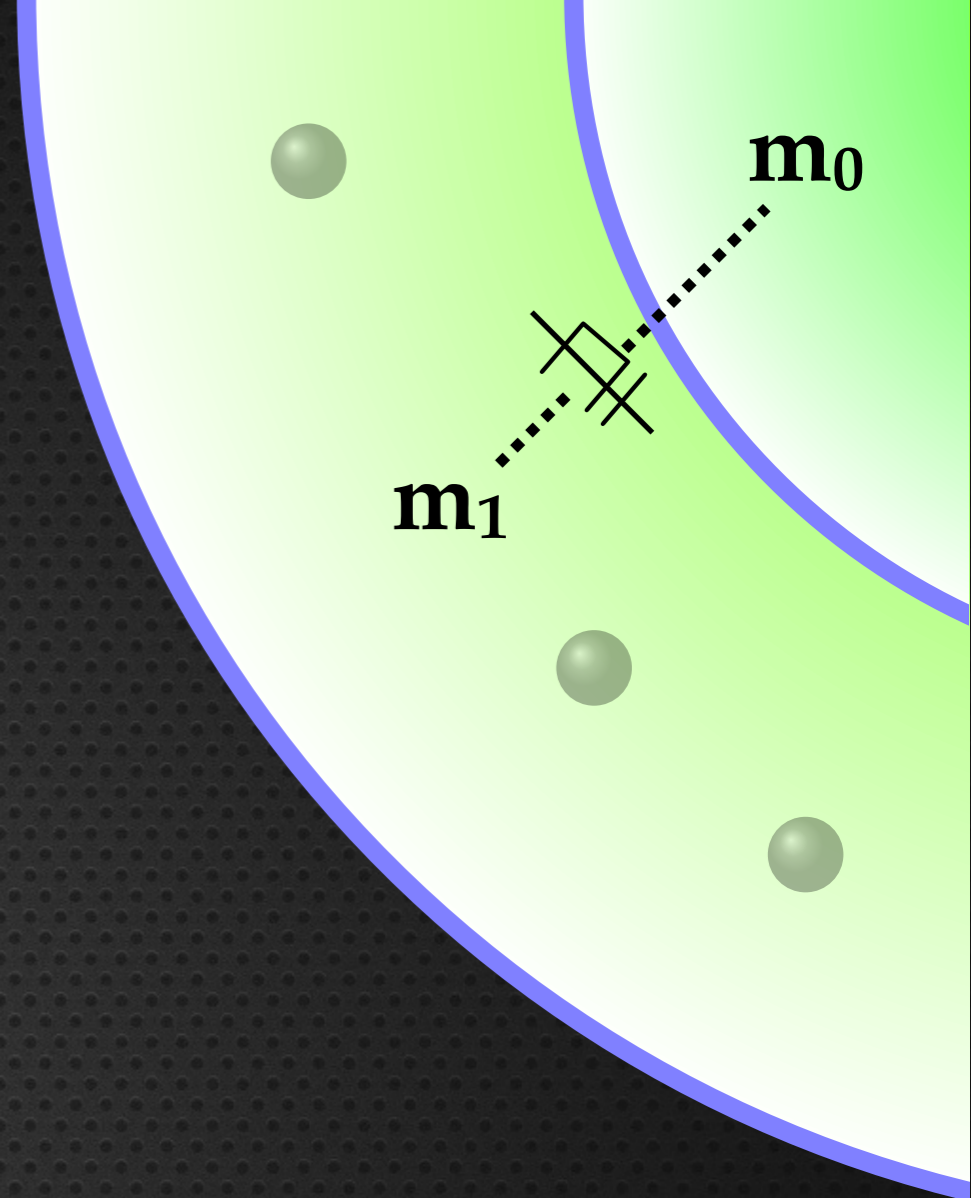


Petri
Backward

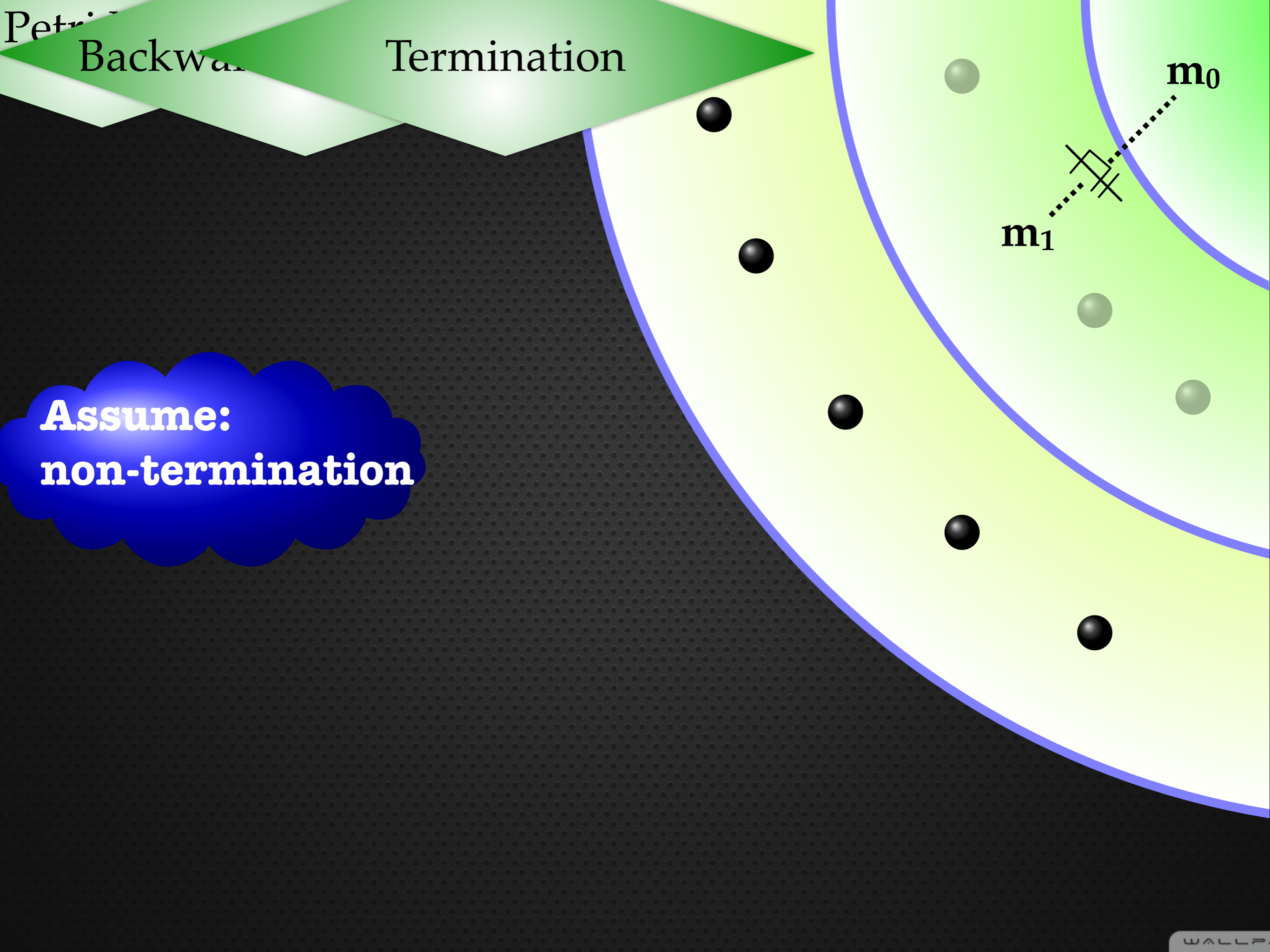
Termination

m_0

**Assume:
non-termination**



**Assume:
non-termination**



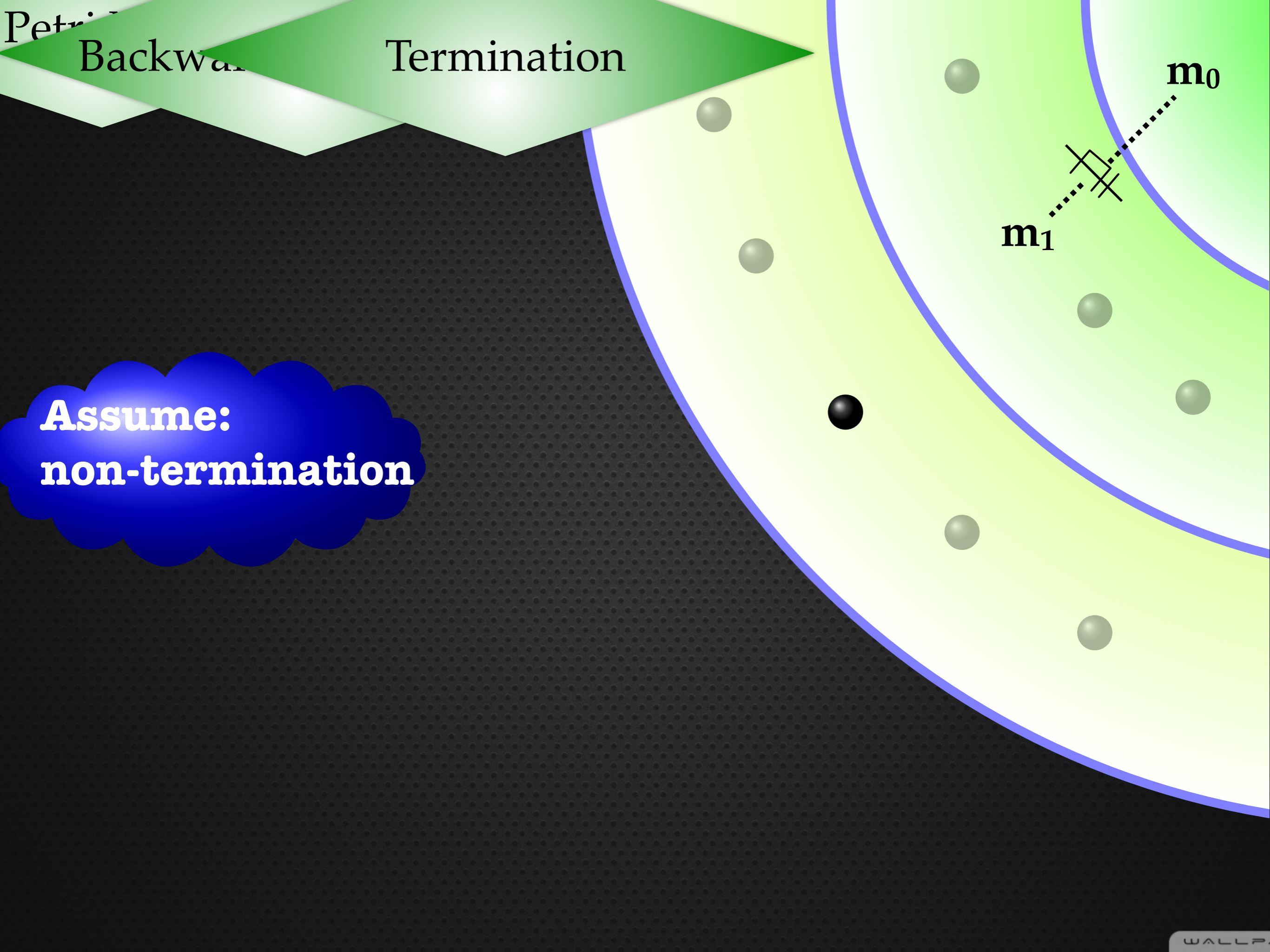
Backward

Termination

m_0

m_1

**Assume:
non-termination**



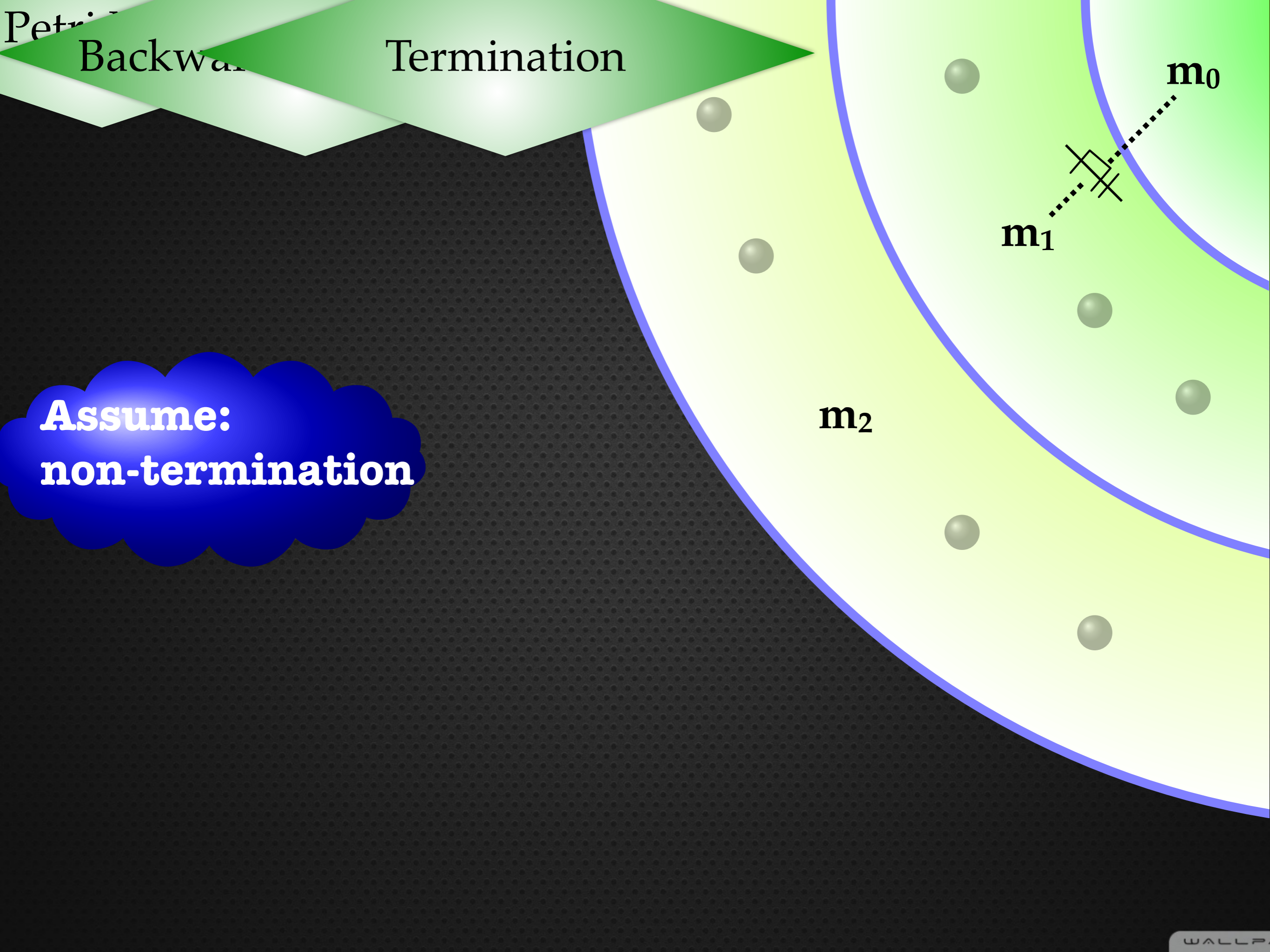
Backward

Termination

m_0

m_1

Assume:
non-termination



Backward

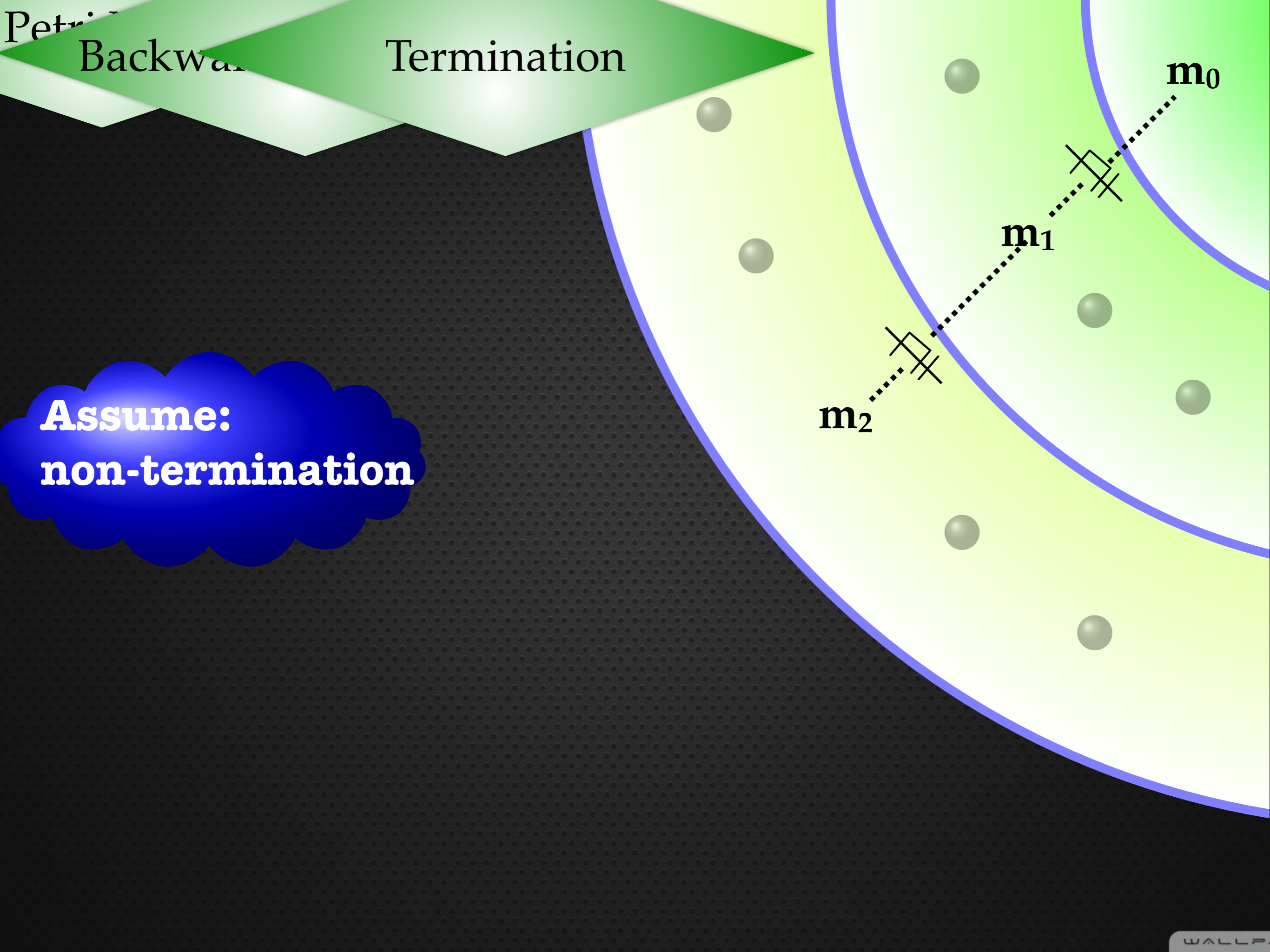
Termination

m_0

m_1

m_2

Assume:
non-termination



Petri

Backward

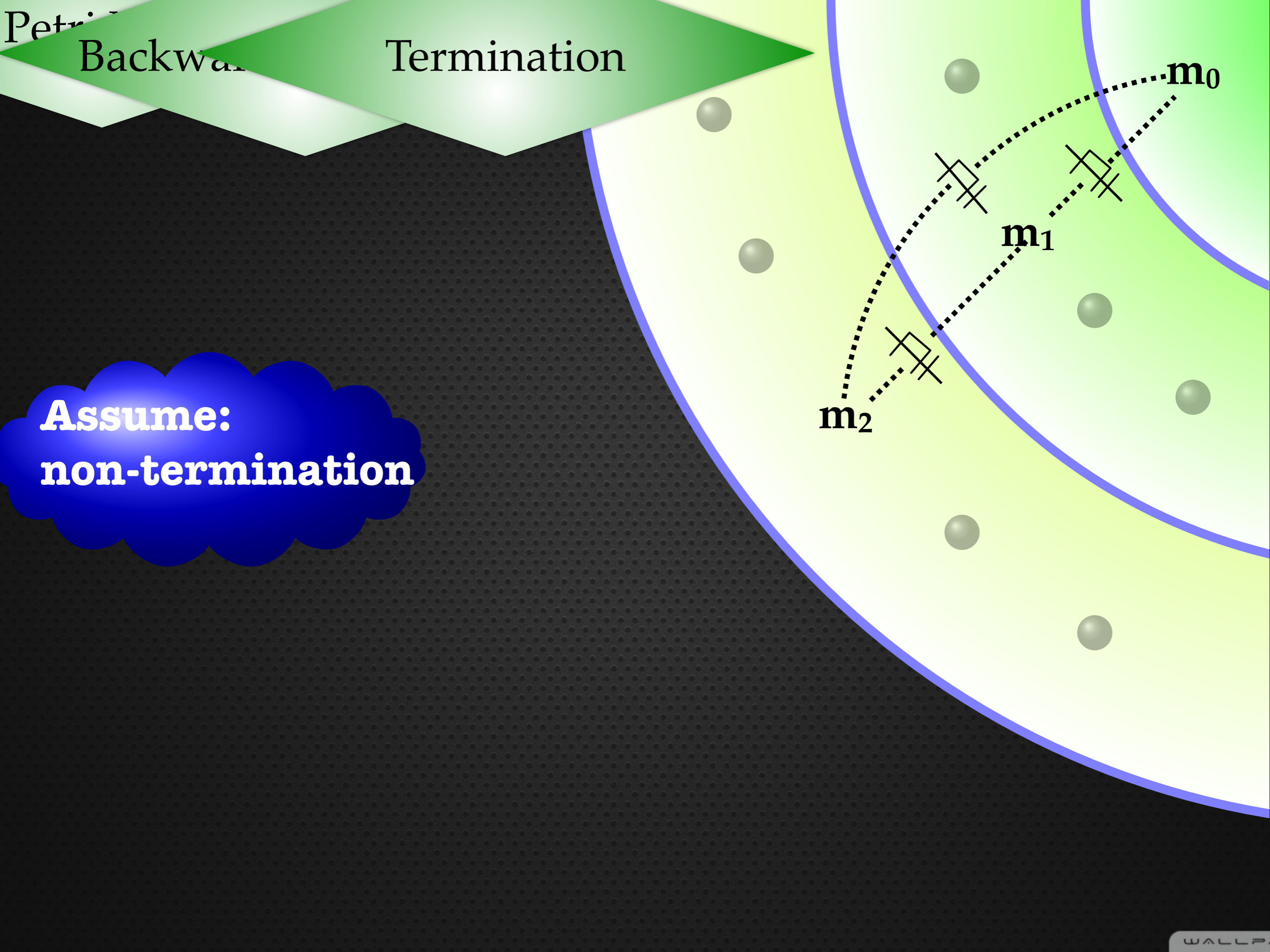
Termination

**Assume:
non-termination**

m_2

m_1

m_0



Backward

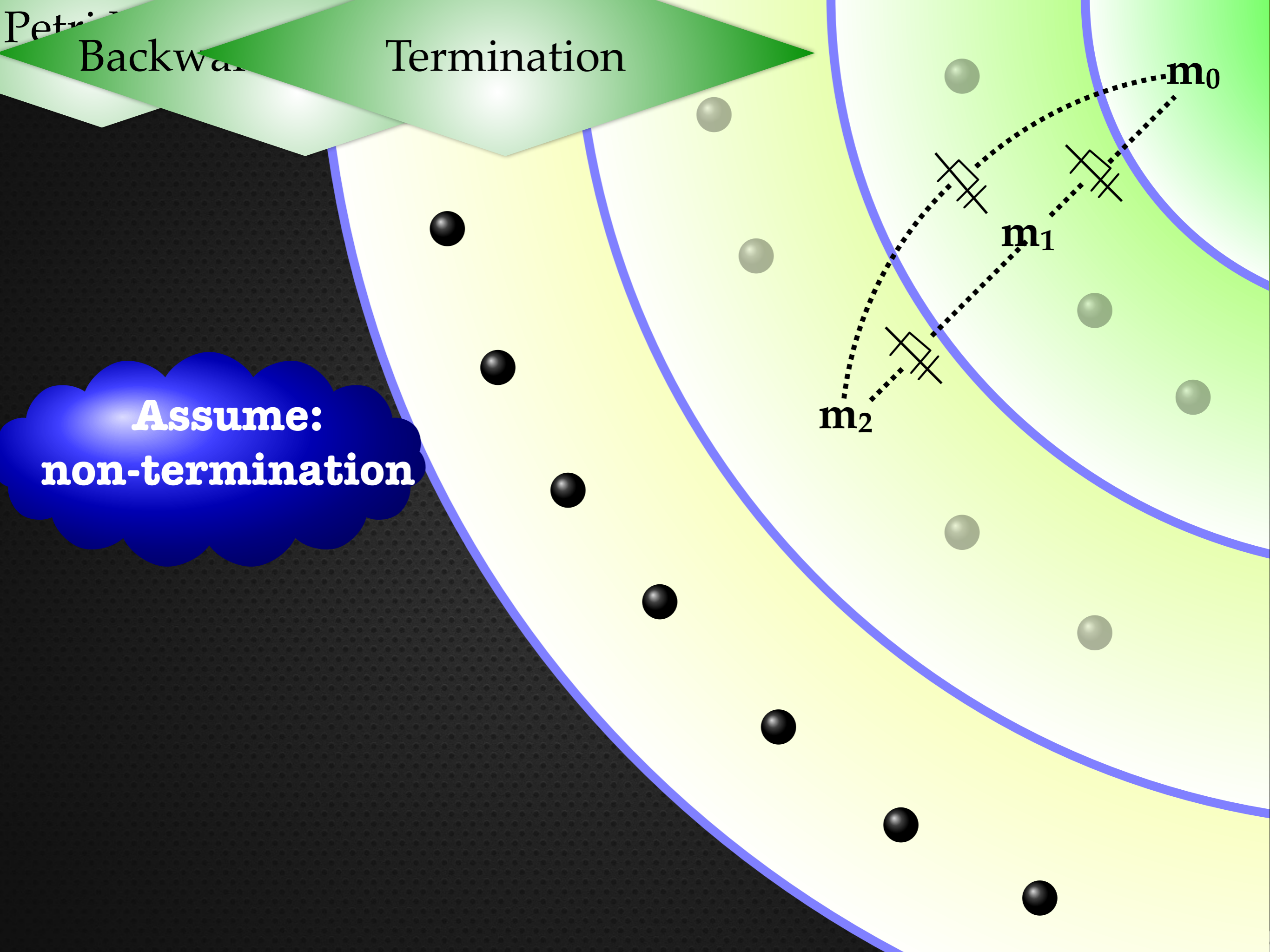
Termination

**Assume:
non-termination**

m_2

m_1

m_0



Petri

Backward

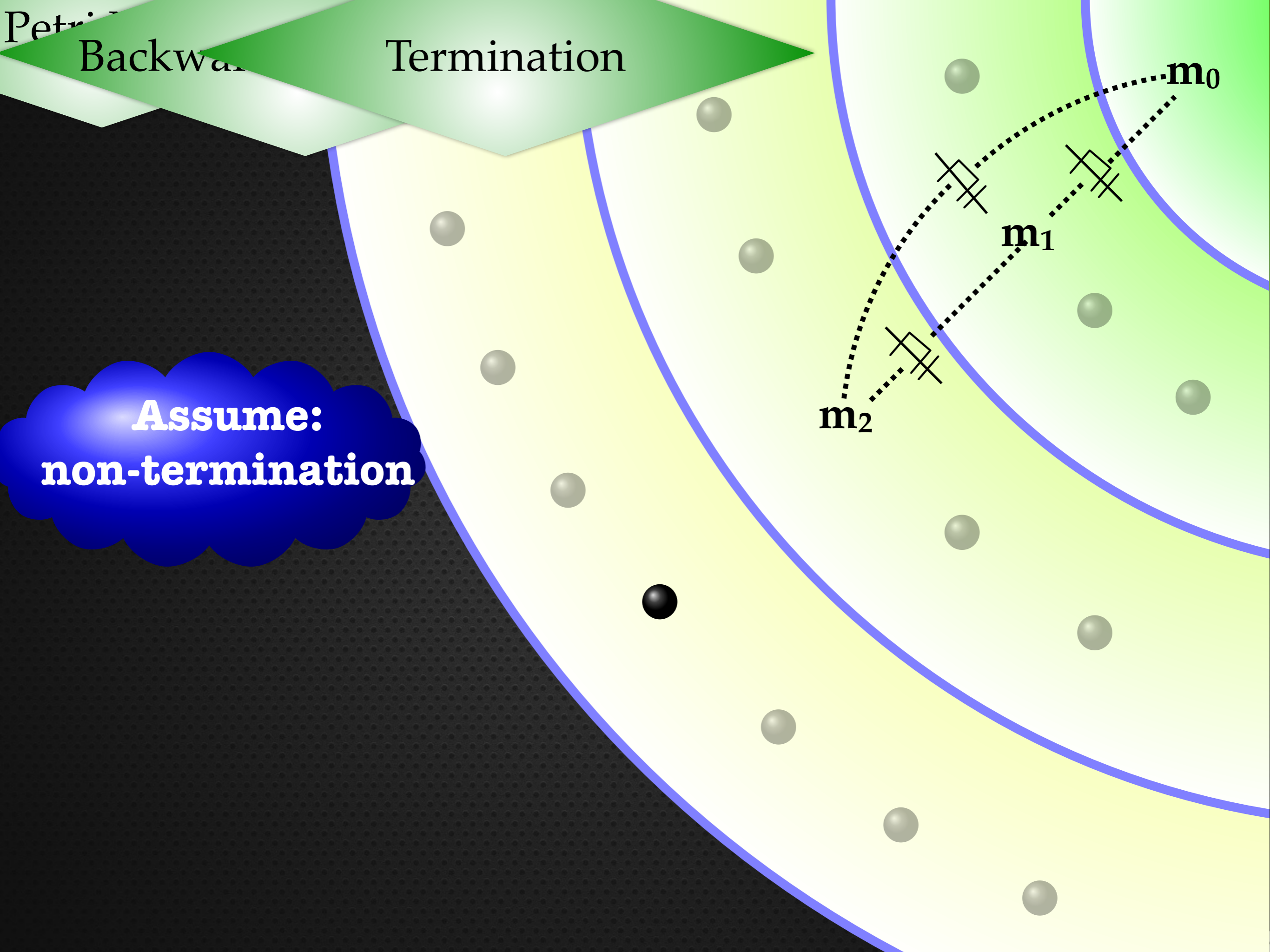
Termination

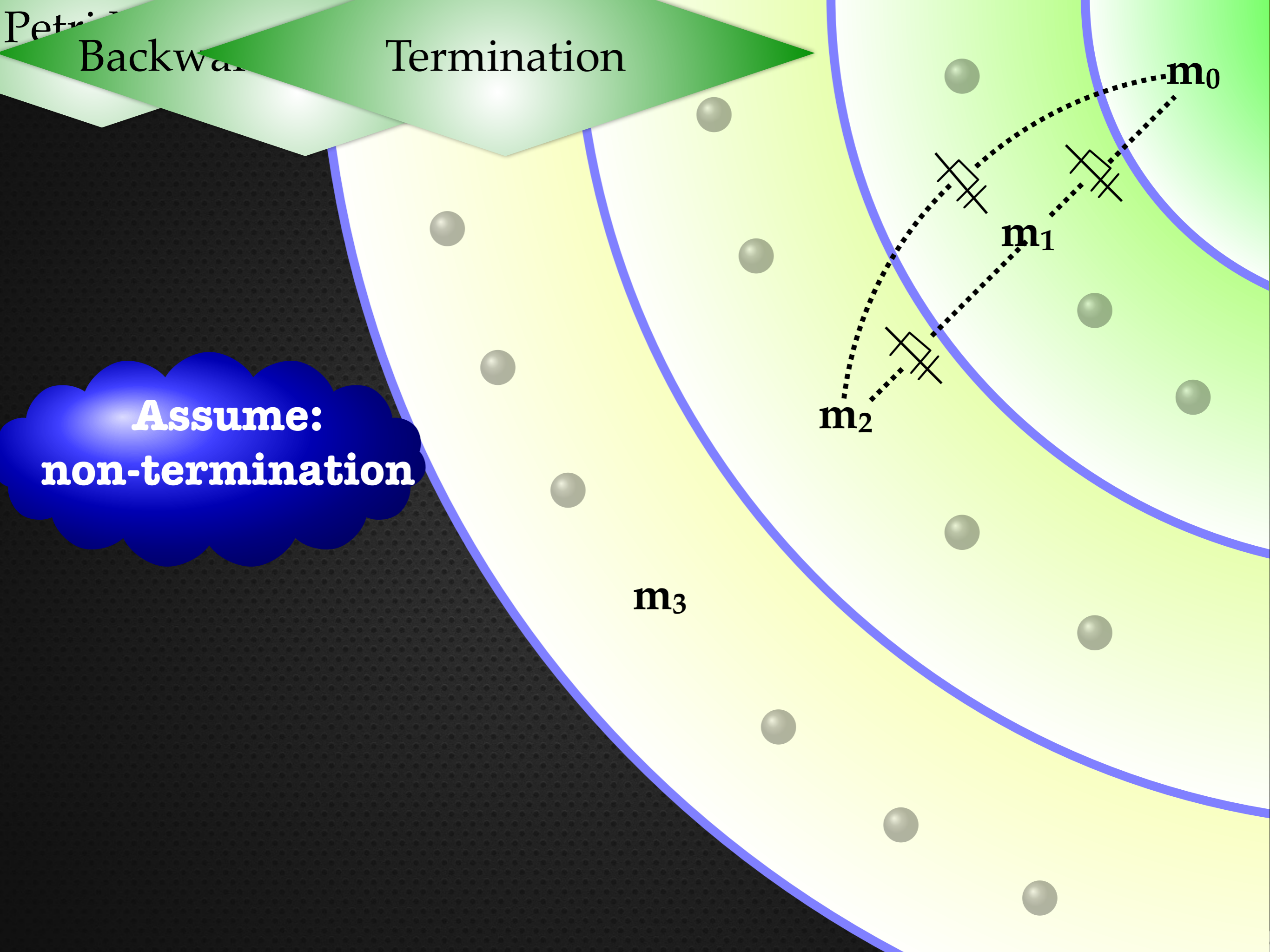
m_0

m_1

m_2

**Assume:
non-termination**





Petri

Backward

Termination

m_0

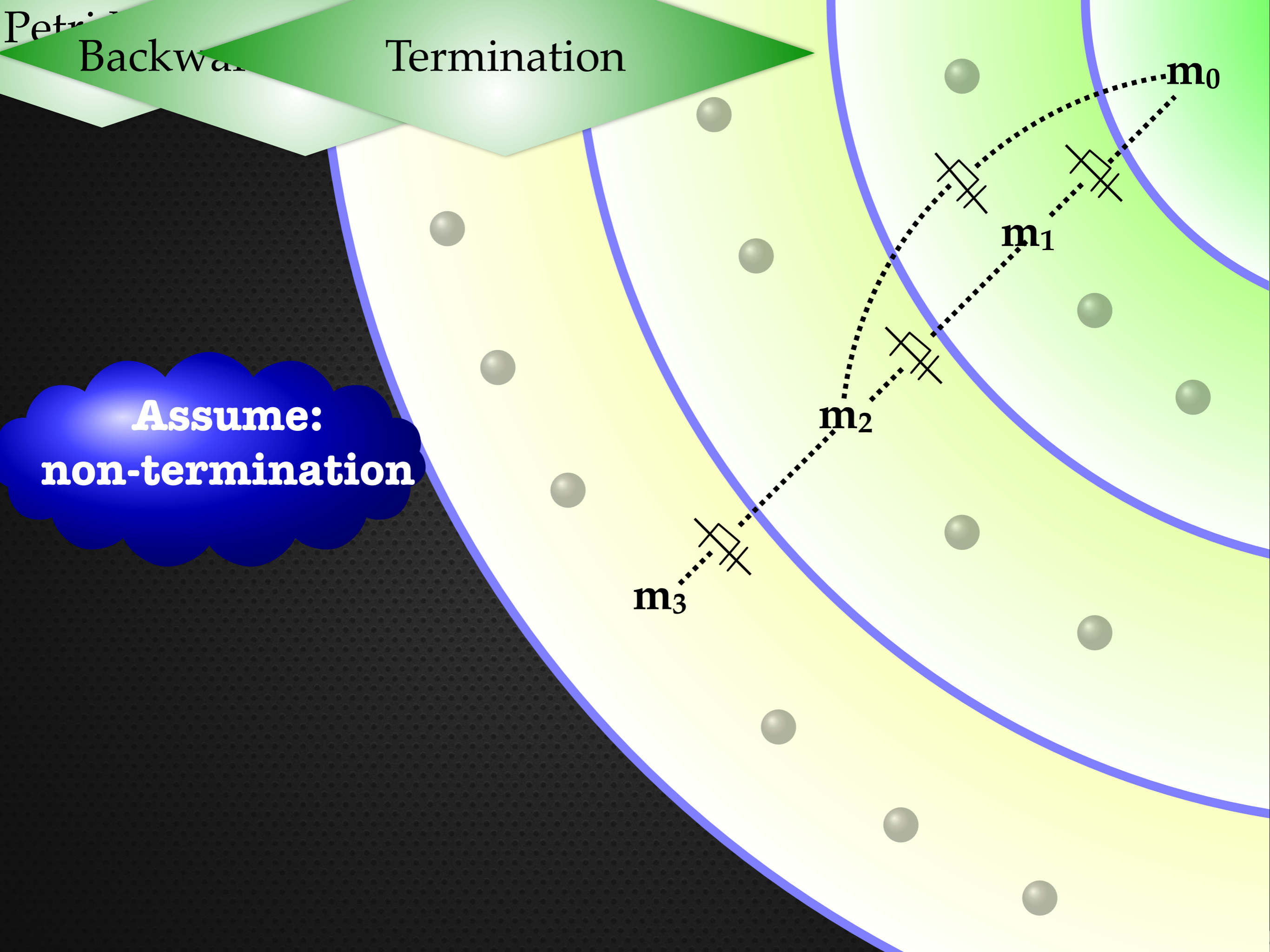
m_1

m_2

m_3

Assume:

non-termination



Petri

Backward

Termination

m_0

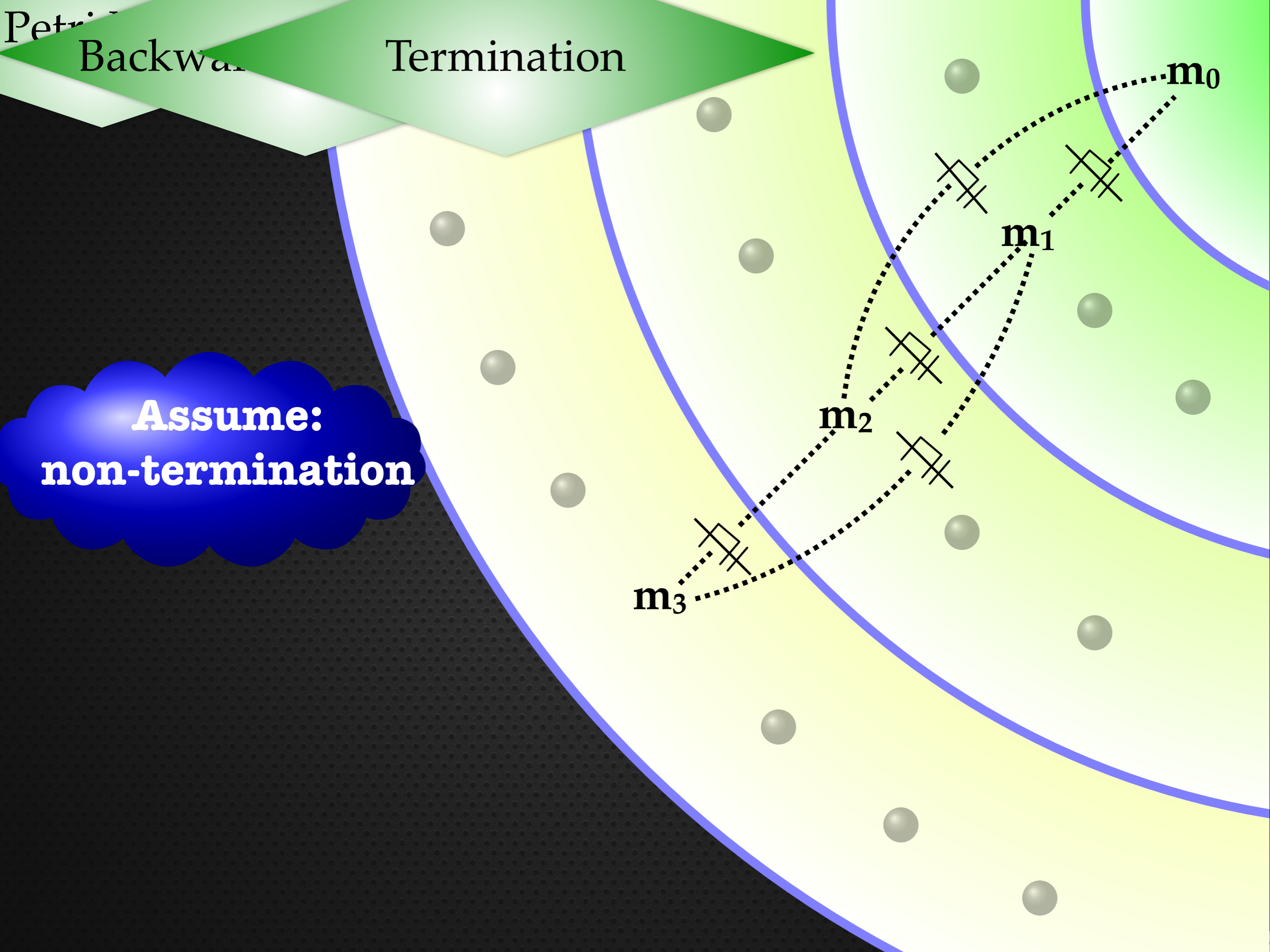
m_1

m_2

m_3

Assume:

non-termination



Petri

Backward

Termination

m_0

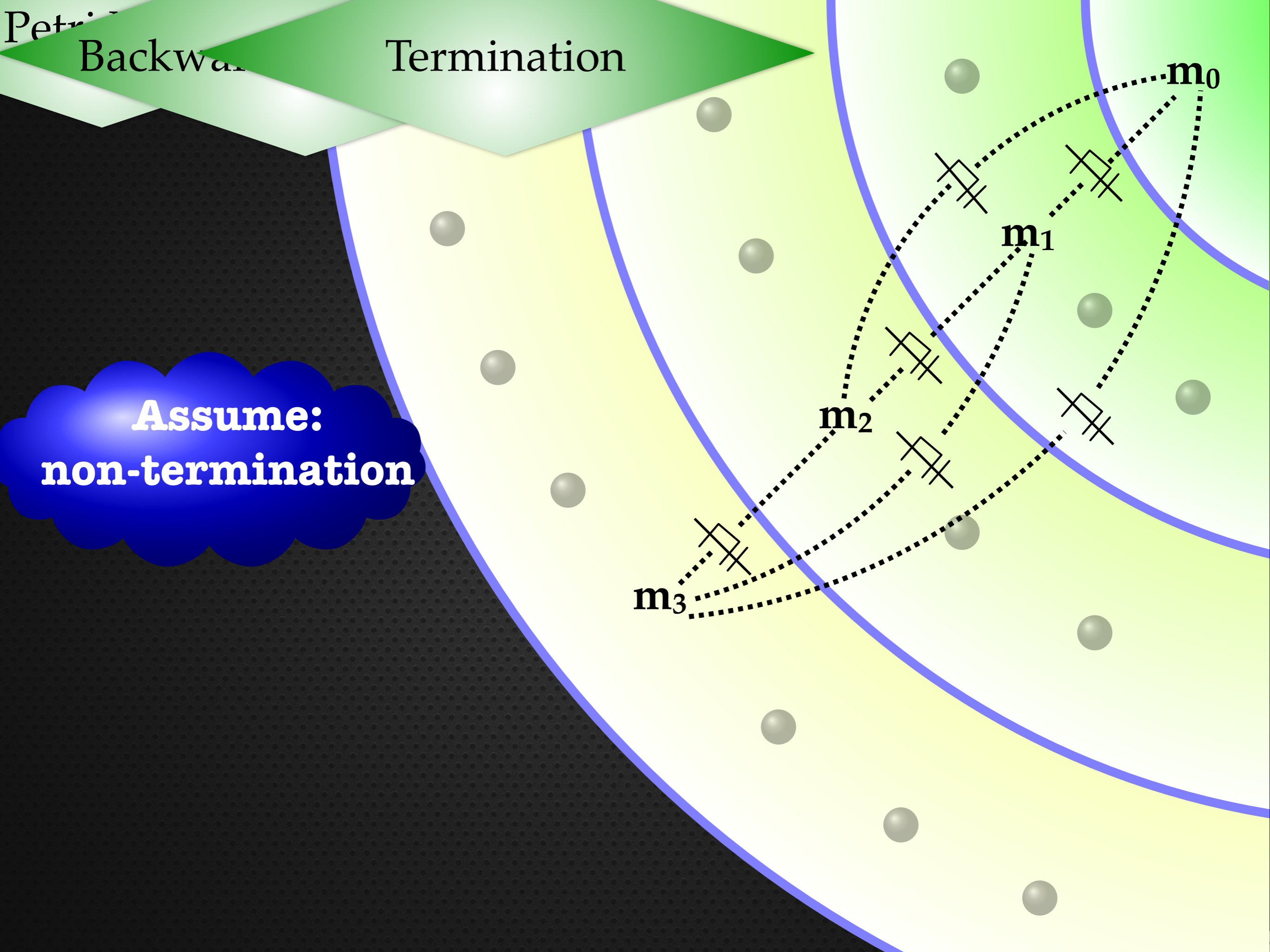
m_1

m_2

m_3

Assume:

non-termination



Petri

Backward

Termination

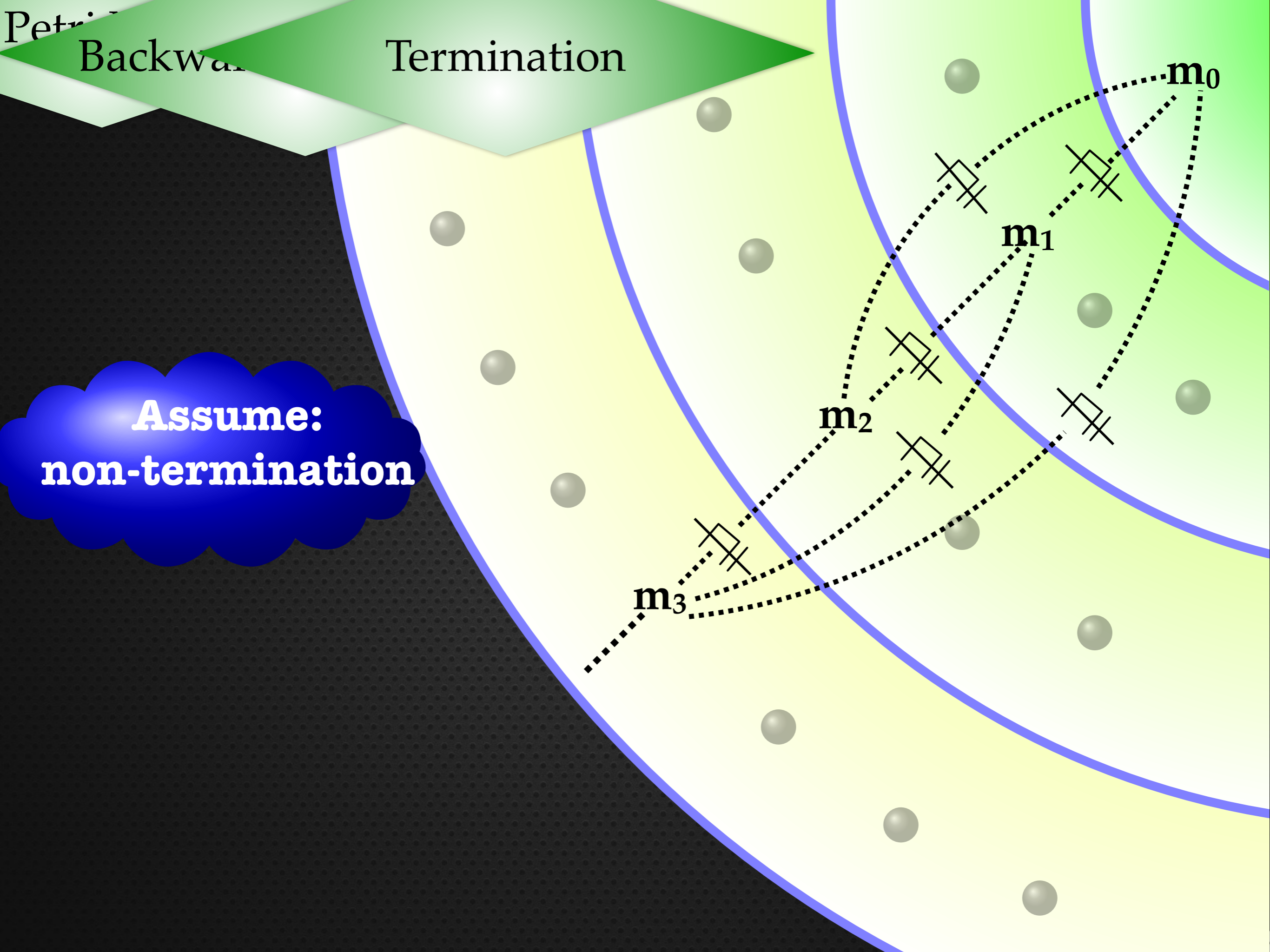
m_0

m_1

m_2

m_3

Assume:
non-termination



Petri

Backward

Termination

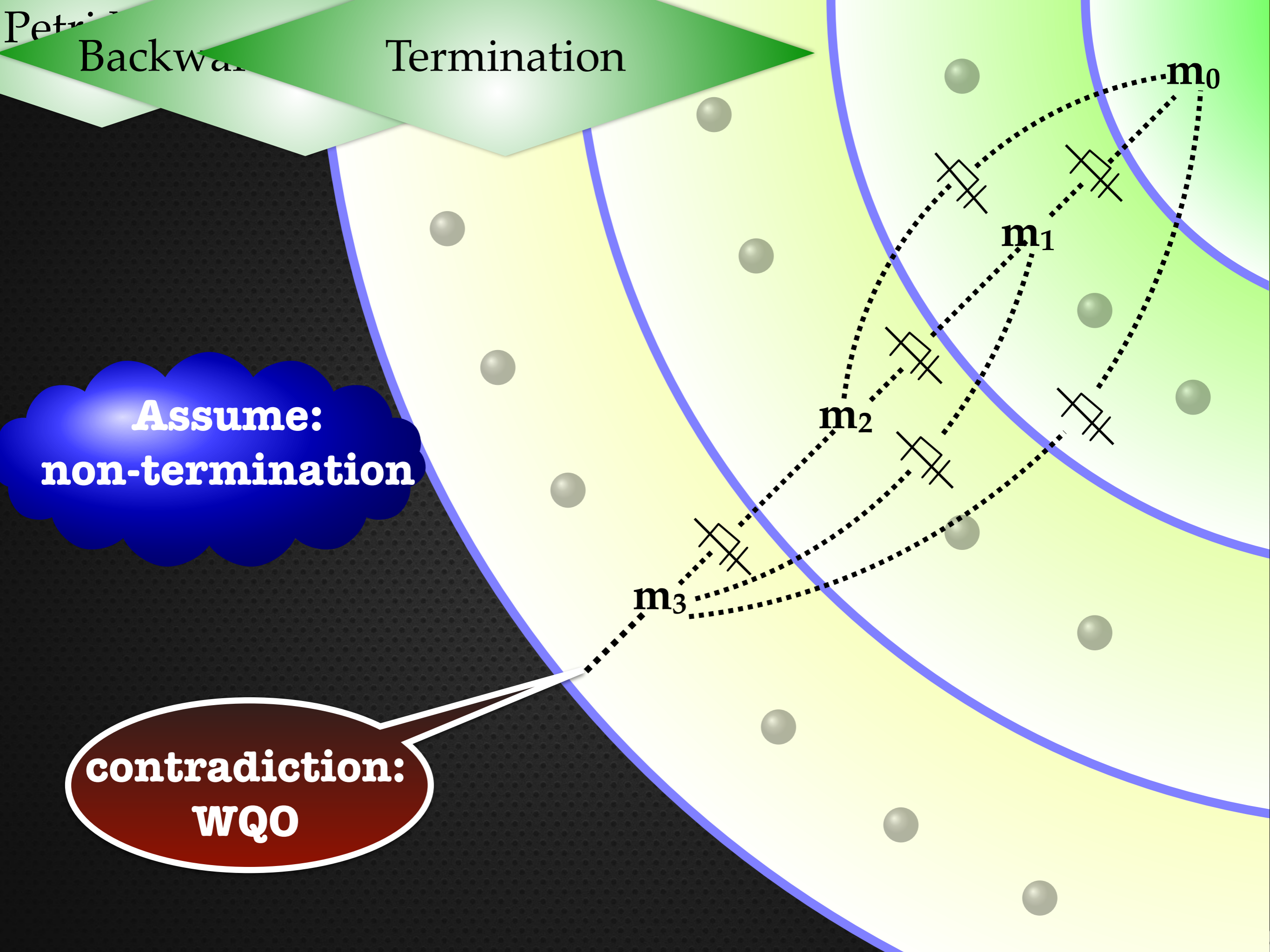
m_0

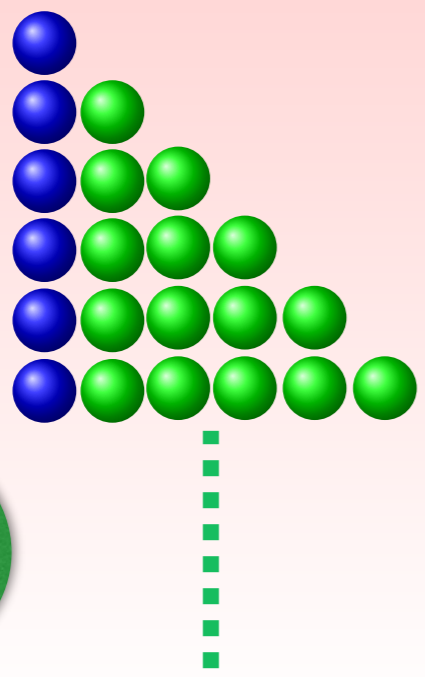
m_1

m_2

m_3

Assume:
non-termination



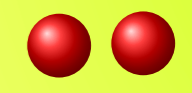
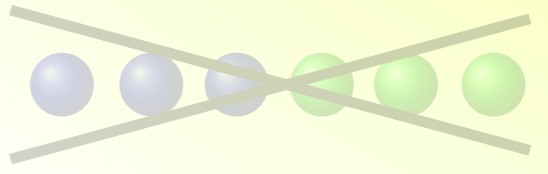


initial markings

System Safe !

symbolic representation = finite multisets

Termination: multisets well quasi-ordered



Ordering:

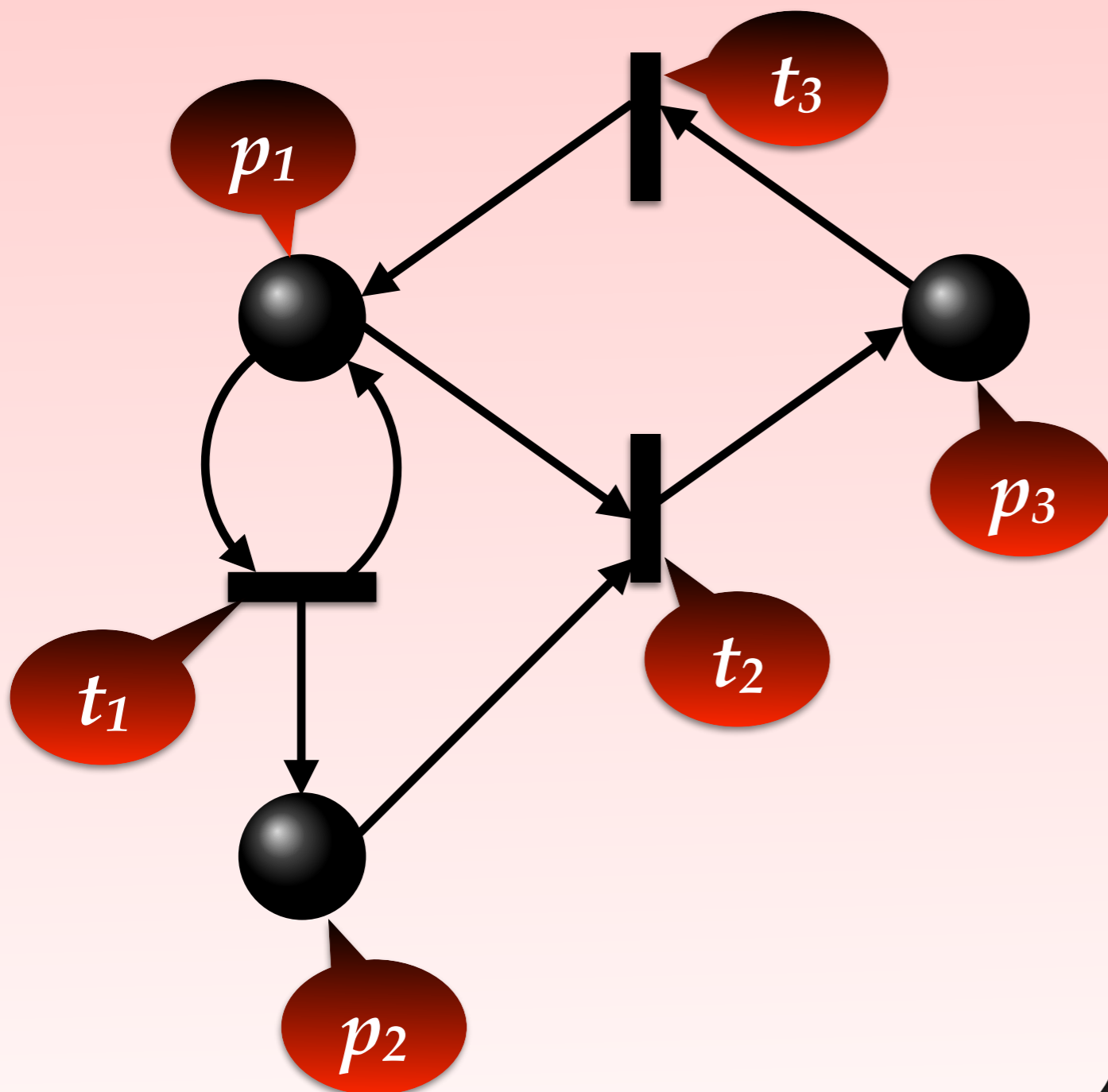
- monotonicity
- computing predecessors
- well quasi-ordering

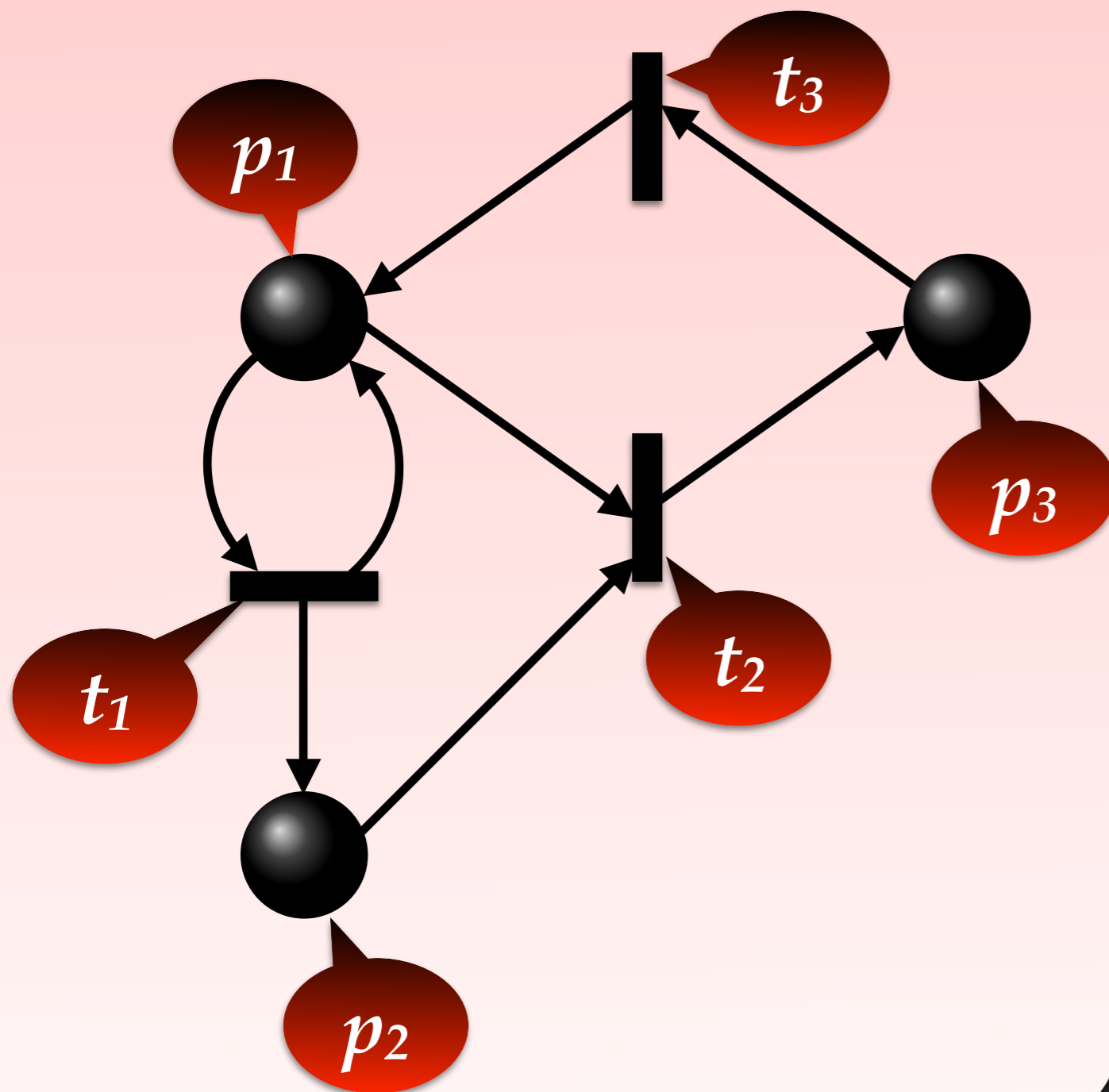
initial
markings

sets

ordered

Petri Nets





- Perform backward reachability analysis from $[p_3, p_3]$
- Reachable from:
 - $[p_1, p_1]$?
 - $[p_1, p_2]$?