ETH zürich

dynamo.ethz.ch

Petri Nets (1)

Jiahui Xu
DYNAMO group

Computer Engineering and
Networks Laboratory

Schedule

We have four exercise sessions:

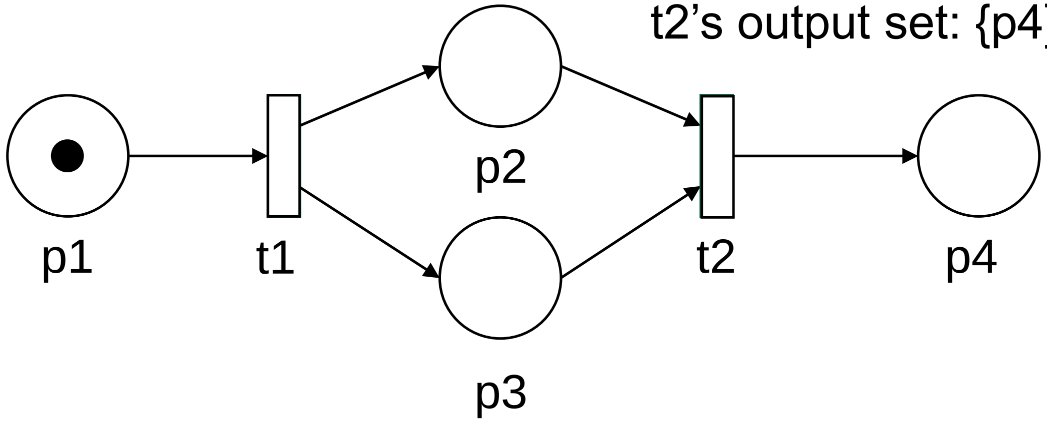
- 30.11.2023: set operations, characteristic functions, BDDs
- 07.12.2023: reachability analysis and temporal logic
- 14.12.2023: Petri nets
- 21.12.2023: time Petri nets

Today's plan: many examples to help us understand the important concepts in Petri nets

Basic Notations

Set of places S: {p1, p2, p3, p4}
Set of transitions T: {t1, t2}
Set of flow relations F: {(p1, t1), (t1, p2), (t1, p3), (p2, t2), (p3, t2), (t2, p4)}

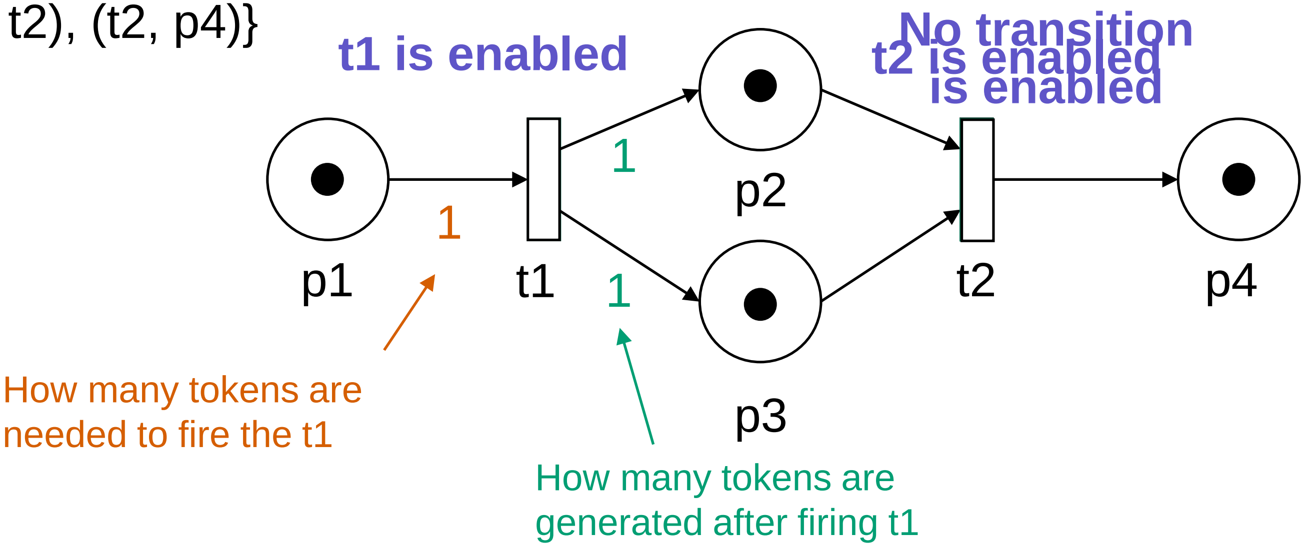
t2’s input set: {p2, p3}
t2’s output set: {p4}



p3’s input set: {t1}
p3’s output set: {t2}

Basic Notations

Set of places S: {p1, p2, p3, p4}
Set of transitions T: {t1, t2}
Set of flow relations F: {(p1, t1), (t1, p2), (t1, p3), (p2, t2), (p3, t2), (t2, p4)}



A transition is said to be **enabled** if all places in the input set have sufficient tokens

A marking M: number of tokens on each place. M = [M(p1), M(p2), M(p3), M(p4)]

[1, 0, 0, 0] —Fire t1→ [0, 1, 1, 0] —Fire t2→ [0, 0, 0, 1]

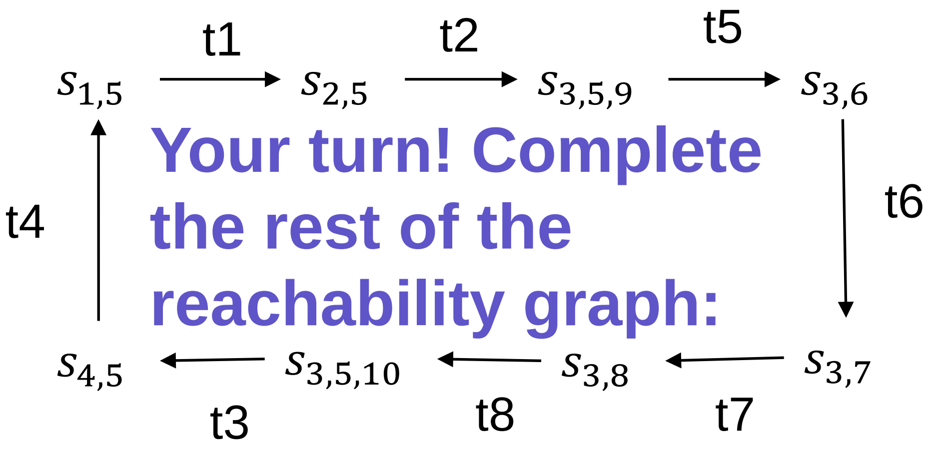
The initial marking M0

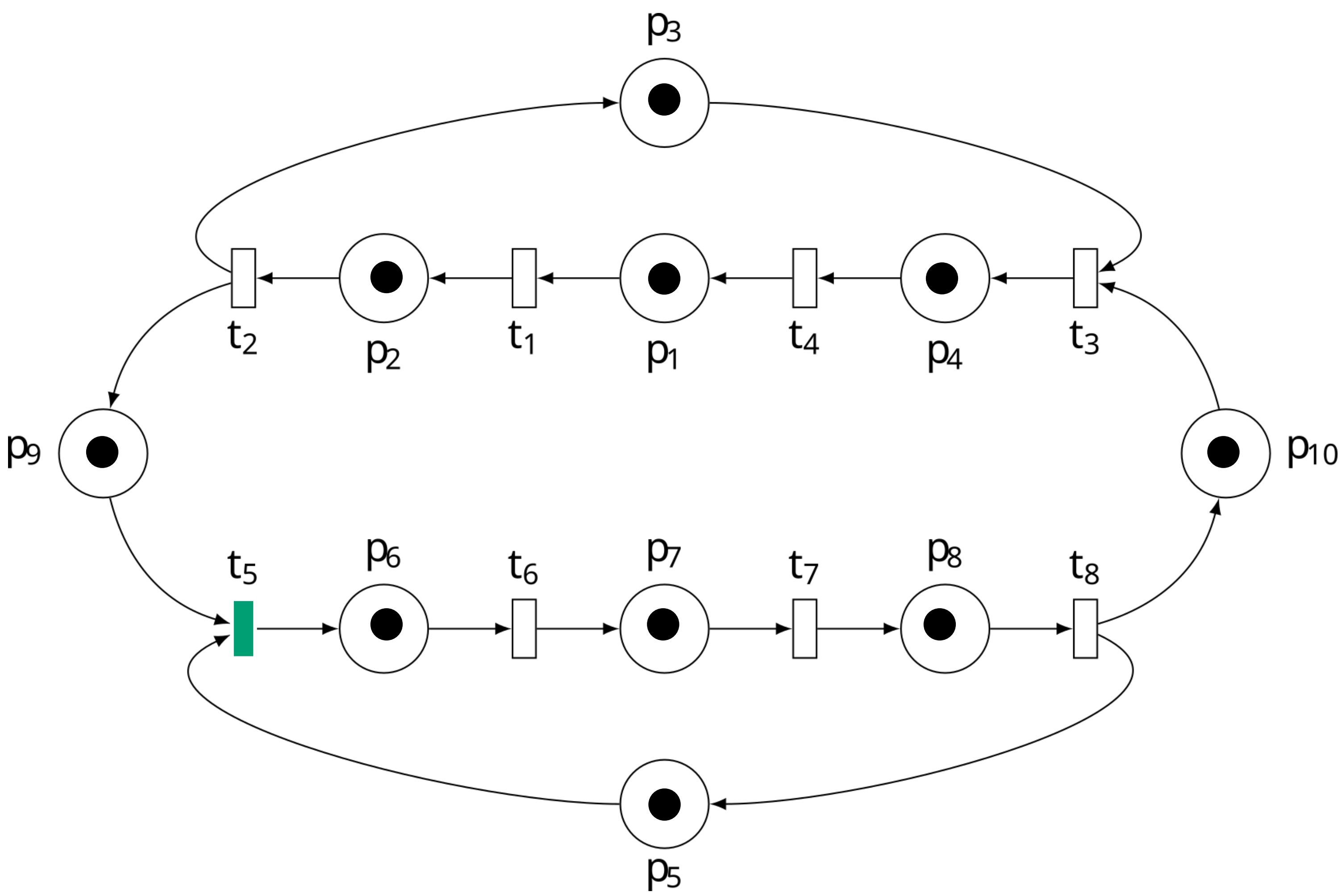
Token game

Notation: $s_{1,5}$ means p1 and p5 are marked.

(b) Starting from the initial marking, which transitions are enabled after we fire t1 and t2?
(c) What is the total number of tokens in the Petri net before and after t2 is fired?
(d) Play a token game for the this Petri net, and draw the reachability graph.

Your turn! Complete the rest of the reachability graph:





(b) t5 is enabled after we fire t1 and t2

Behavioral properties

For different values of k, is the Petri net **deadlock free**? Is the Petri net **N-bounded**? If yes, what is the smallest N? How can we to prove the properties?

k = 0: 1-bounded and not deadlock free.

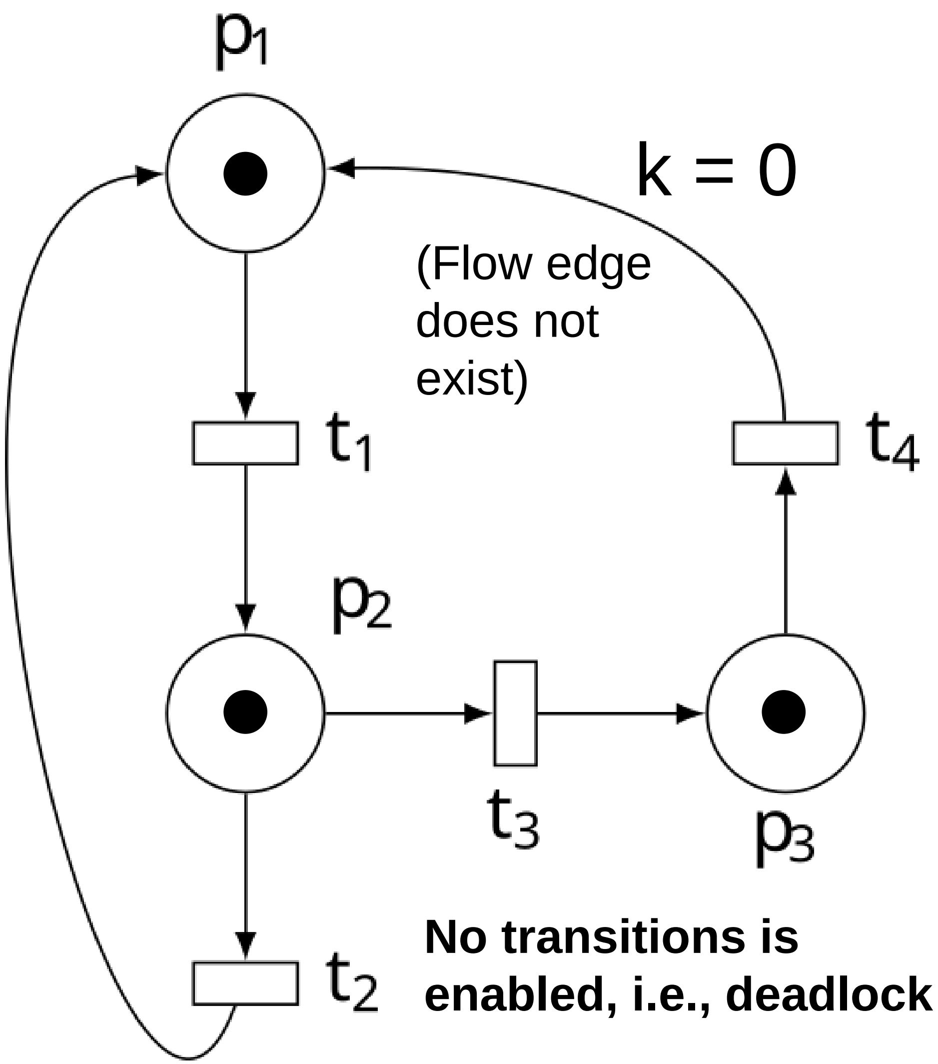
Proof: a reachability graph with all markings 1-bounded (in another exercise)

Proof: a sequence of firing leads to deadlock state

Can you give a different proof?

Fire sequence: t1 t3 t4

Your turn! What happens for other values of k?



No transitions is enabled, i.e., deadlock

N-boundedness: in all reachable markings, no place holds more than N tokens.
Deadlock-freeness: in all reachable markings, at least one transition is enabled.

Behavioral properties

For different values of k, is the Petri net **deadlock free**? Is the Petri net **N-bounded**? If yes, what is the smallest N? How can we to prove the properties?

k = 1: 1-bounded and deadlock free.

No transition changes the total number of tokens.

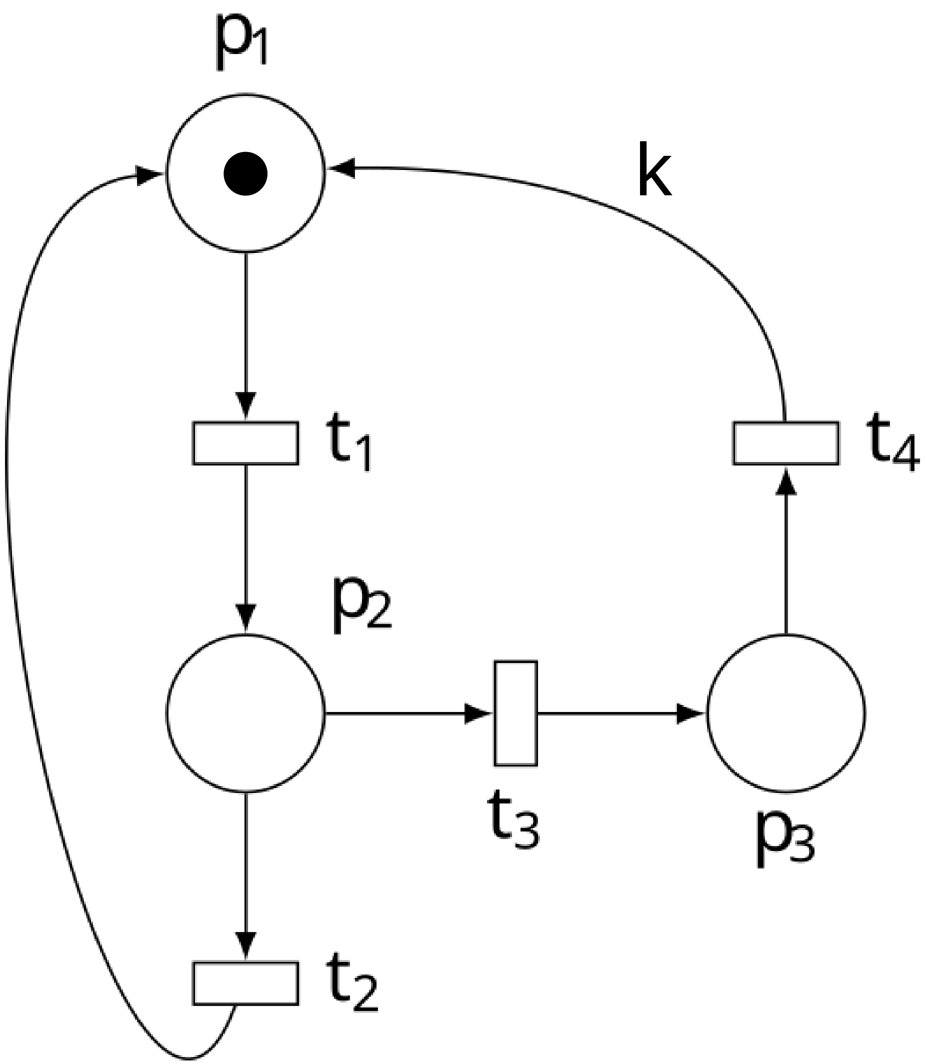
Initial marking has 1 token.

All *reachable* markings have exactly 1 token

The only marking that deadlocks is when no place has token.

The deadlock marking is not reachable, thus, the Petri net is deadlock free

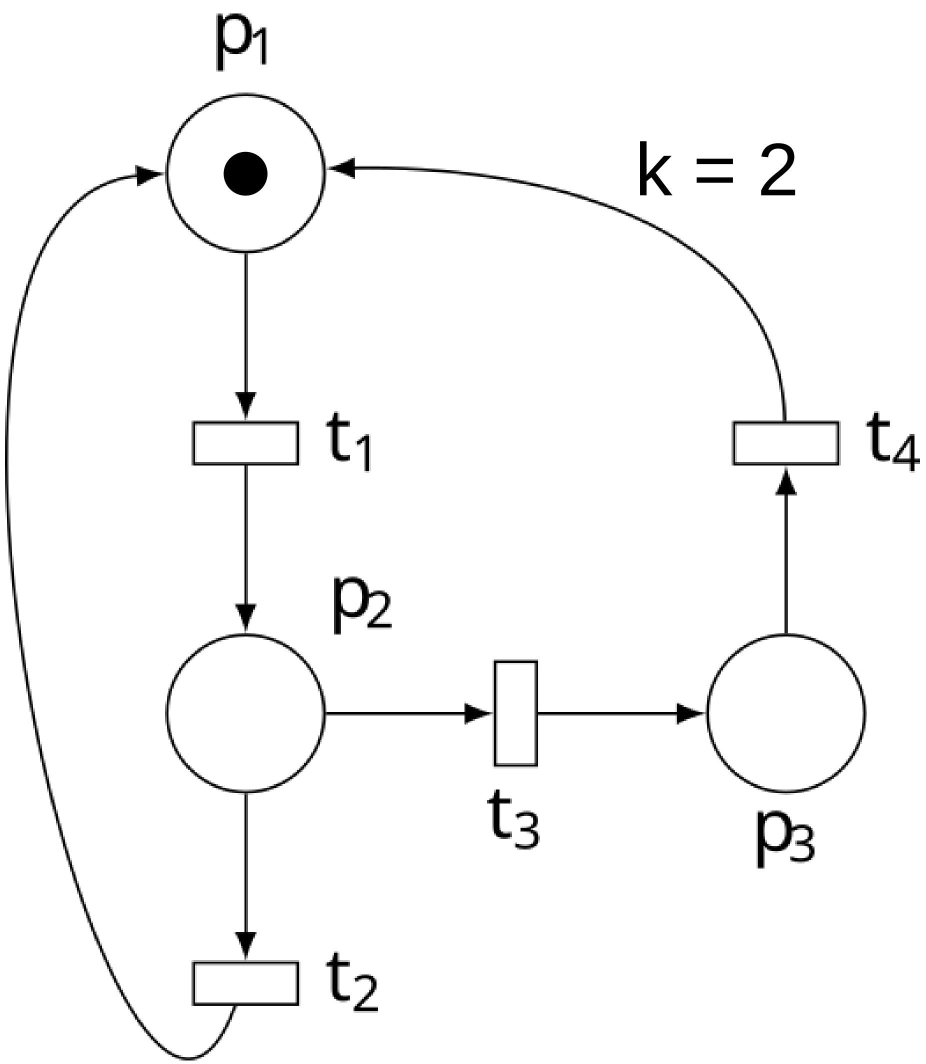
Place invariant ("AG p"): (M(p1) + M(p2) + M(p3) == 1)



N-boundedness: in all reachable markings, no place holds more than N tokens.
Deadlock-freeness: in all reachable markings, at least one transition is enabled.

Behavioral properties

For different values of k, is the Petri net **deadlock free**? Is the Petri net **N-bounded**? If yes, what is the smallest N? How can we to prove the properties?



k > 1: unbounded and deadlock free.

Repeatedly fire t1, t3, t4 will accumulate tokens infinitely at p1

The net always has at least one token, therefore the deadlock marking is unreachable.

N-boundedness: in all reachable markings, no place holds more than N tokens.
Deadlock-freeness: in all reachable markings, at least one transition is enabled.

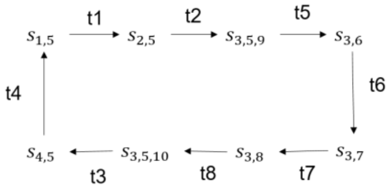
Coverability and Reachability Graph

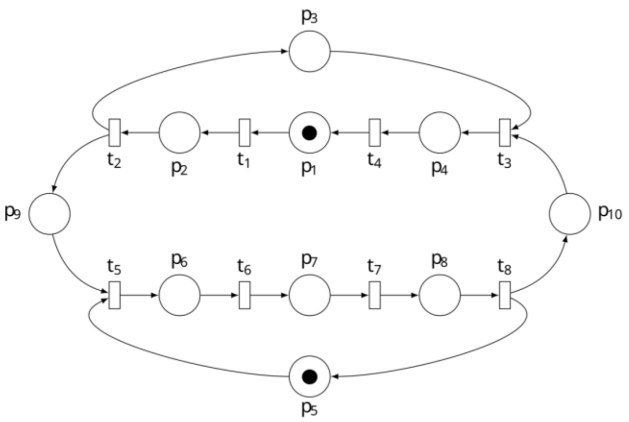
A Petri net that has a finite number of states

Token game

Notation: $s_{1,5}$ means p1 and p5 are marked.

(b) Starting from the initial marking, which transitions are enabled after we fire t1 and t2?
(c) What is the total number of tokens in the Petri net before and after t2 is fired?
(d) Play a token game for the this Petri net, and draw the reachability graph.





Now we also want to characterize state-space with infinitely many states

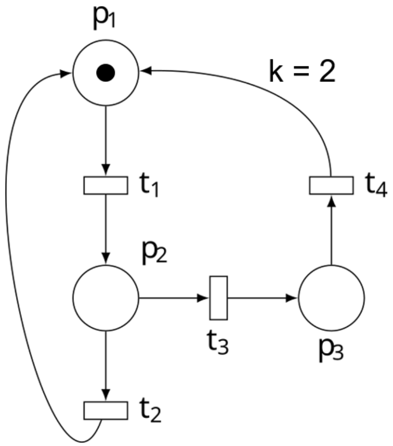
Behavioral properties

For different values of k, is the Petri net **deadlock free**? Is the Petri net **N-bounded**? If yes, what is the smallest N? How can we to prove the properties?

k > 1: unbounded and deadlock free.

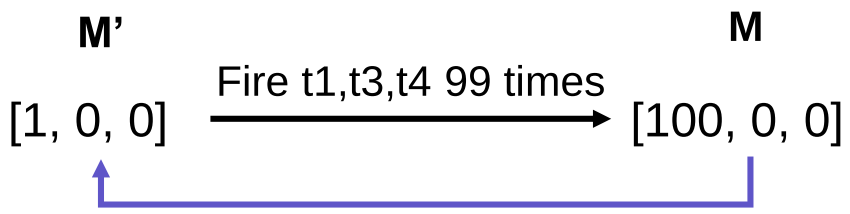
Repeatedly fire t1, t3, t4 will accumulate tokens infinitely at p1

The net always has at least one token, therefore the deadlock marking is unreachable.

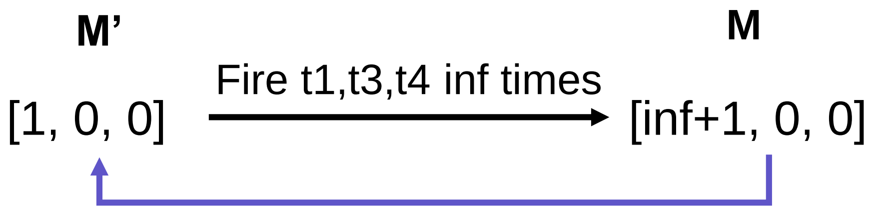


Coverability and Reachability Graph

Cover: for a Petri net, given two markings M and M',
M covers M' if for each place p, M(p) >= M'(p).



Marking M covers M'



Marking M covers M'

Now we also want to characterize state-space with infinitely many states

Behavioral properties

For different values of k, is the Petri net **dea**
the Petri net **N-bounded**? If yes, what is the
How can we to prove the properties?

k > 1: unbounded and deadlock free

Repeatedly fire t1, t3, t4 will accumulate tokens infinitely at p1

The net
token, th
marking

Coverability Tree – Algorithm

Special symbol ω, similar to ∞: ∀n∈**N**: ω > n; ω = ω ± n; ω ≥ ω

Label initial marking M_0 as root and tag it as *new*

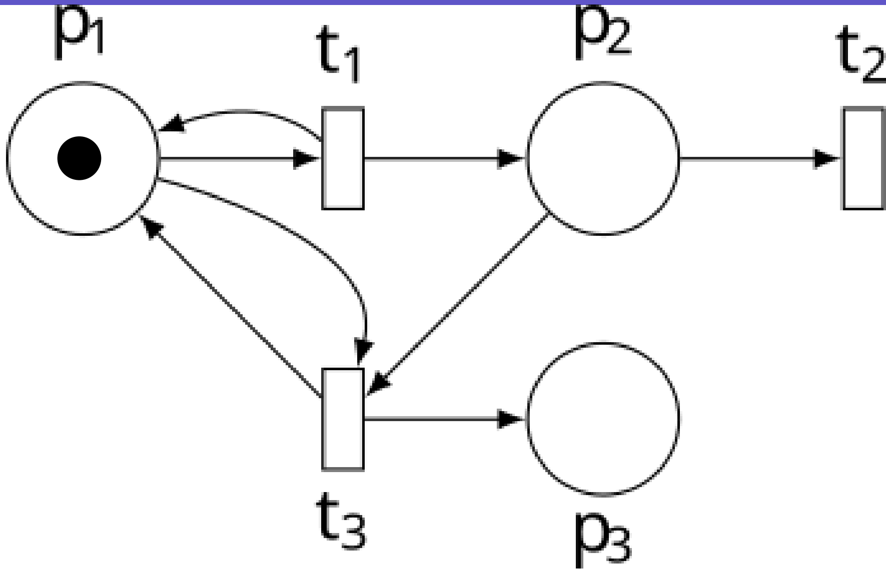
while *new* tags exist, pick one, say M

- Remove tag *new* from M;
- If M is identical to an already existing marking, tag it as *old*; **continue**;
- If no transitions are enabled at M, tag it as *deadlock*; **continue**;
- For each enabled transition t at M do
  - Obtain marking M' = M · t
  - If there exists a marking M'' on the path from the root to M s.t. M'(*p*) ≥ M''(*p*) for each place p and M' ≠ M'', replace M'(*p*) with ω for *p* where M'(*p*) > M''(*p*).
  - Introduce M' as a node, draw an arc with label t from M to M' and tag M' *new*.

Coverability and Reachability Graph

Cover: for a Petri net, given two markings M and M',
M covers M' if for each place p, M(p) >= M'(p).

[1, 0, 0]   new



Coverability Tree – Algorithm

Special symbol ω, similar to ∞: $\forall n \in \mathbf{N}$: $\omega > n$; $\omega = \omega \pm n$; $\omega \geq \omega$

Label initial marking M_0 as root and tag it as *new*

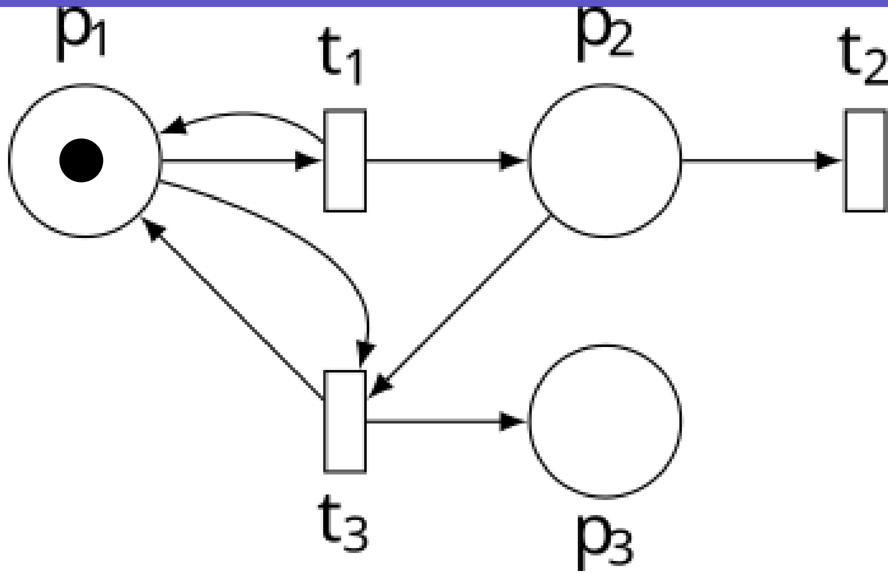
while *new* tags exist, pick one, say M

- Remove tag *new* from M;
- If M is identical to an already existing marking, tag it as *old*; **continue**;
- If no transitions are enabled at M, tag it as *deadlock*; **continue**;
- For each enabled transition t at M do
  - Obtain marking M' = M · t
  - If there exists a marking M'' on the path from the root to M s.t. M'(*p*) ≥ M''(*p*) for each place p and M' ≠ M'', replace M'(*p*) with ω for *p* where M'(*p*) > M''(*p*).
  - Introduce M' as a node, draw an arc with label t from M to M' and tag M' *new*.

Coverability and Reachability Graph

Cover: for a Petri net, given two markings M and M',
M covers M' if for each place p, M(p) >= M'(p).

[1, 0, 0] Enabled: t1



Coverability Tree – Algorithm

Special symbol ω, similar to ∞: $\forall n \in \mathbf{N}$: $\omega > n$; $\omega = \omega \pm n$; $\omega \geq \omega$

Label initial marking M_0 as root and tag it as *new*

while *new* tags exist, pick one, say M

- Remove tag *new* from M;
- If M is identical to an already existing marking, tag it as *old*; **continue**;
- If no transitions are enabled at M, tag it as *deadlock*; **continue**;
- For each enabled transition t at M do
  - Obtain marking M' = M · t
  - If there exists a marking M'' on the path from the root to M s.t. M'(p) ≥ M''(p) for each place p and M' ≠ M'', replace M'(p) with ω for p where M'(p) > M''(p).
  - Introduce M' as a node, draw an arc with label t from M to M' and tag M' *new*.

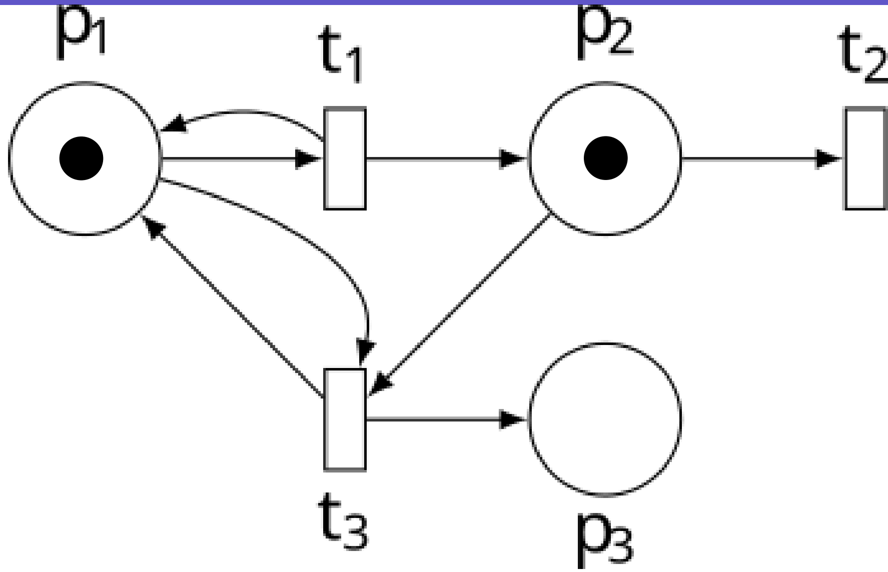
Coverability and Reachability Graph

Cover: for a Petri net, given two markings M and M',
M covers M' if for each place p, M(p) >= M'(p).

[1, 0, 0] Enabled: t1

t1 ↓

[1, 1, 0]



Coverability Tree – Algorithm

Special symbol ω, similar to ∞: $\forall n \in \mathbf{N}$: $\omega > n$; $\omega = \omega \pm n$; $\omega \geq \omega$

Label initial marking M_0 as root and tag it as *new*

while *new* tags exist, pick one, say M

- Remove tag *new* from M;
- If M is identical to an already existing marking, tag it as *old*; **continue**;
- If no transitions are enabled at M, tag it as *deadlock*; **continue**;
- For each enabled transition t at M do
  - Obtain marking M' = M · t
  - If there exists a marking M'' on the path from the root to M s.t. M'(*p*) ≥ M''(*p*) for each place p and M' ≠ M'', replace M'(*p*) with ω for *p* where M'(*p*) > M''(*p*).
  - Introduce M' as a node, draw an arc with label t from M to M' and tag M' *new*.

Coverability and Reachability Graph

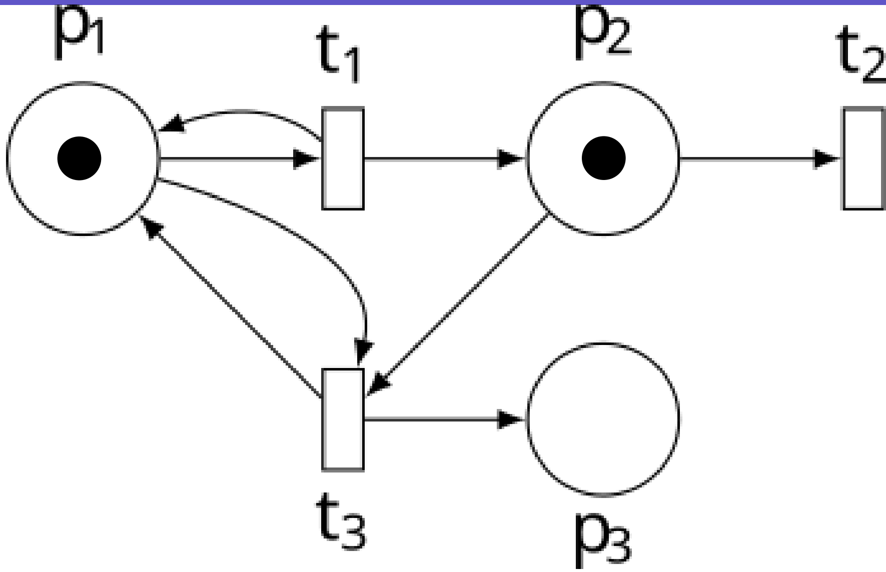
Cover: for a Petri net, given two markings M and M',
M covers M' if for each place p, M(p) >= M'(p).

[1, 0, 0] Enabled: t1

↓ t1

[1, 1, 0]

[1, 1, 0] covers [0, 0, 0] and they are not the same



Coverability Tree – Algorithm

Special symbol ω, similar to ∞: $\forall n \in \mathbf{N}$: $\omega > n$; $\omega = \omega \pm n$; $\omega \geq \omega$

Label initial marking M_0 as root and tag it as *new*

while *new* tags exist, pick one, say M

- Remove tag *new* from M;
- If M is identical to an already existing marking, tag it as *old*; **continue**;
- If no transitions are enabled at M, tag it as *deadlock*; **continue**;
- For each enabled transition t at M do
  - Obtain marking $M' = M \cdot t$
  - If there exists a marking M'' on the path from the root to M s.t. $M'(p) \geq M''(p)$ for each place p and $M' \neq M''$, replace $M'(p)$ with ω for *p* where $M'(p) > M''(p)$.
  - Introduce M' as a node, draw an arc with label t from M to M' and tag M' *new*.

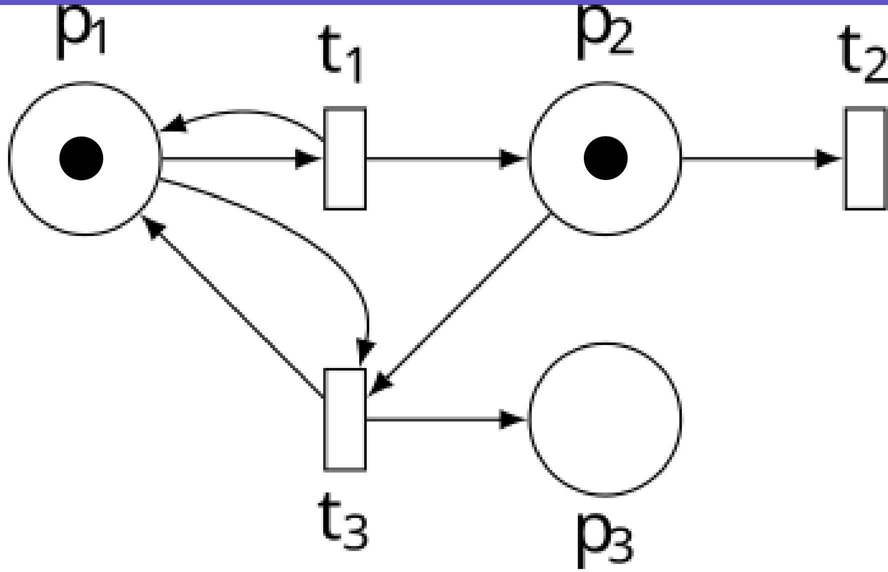
Coverability and Reachability Graph

Cover: for a Petri net, given two markings M and M',
M covers M' if for each place p, M(p) >= M'(p).

[1, 0, 0] Enabled: t1

↓ t1

[1, ω, 0] new



Coverability Tree – Algorithm

Special symbol ω, similar to ∞: $\forall n \in \mathbf{N}$: $\omega > n$; $\omega = \omega \pm n$; $\omega \geq \omega$

Label initial marking M_0 as root and tag it as *new*

while *new* tags exist, pick one, say M

- Remove tag *new* from M;
- If M is identical to an already existing marking, tag it as *old*; **continue**;
- If no transitions are enabled at M, tag it as *deadlock*; **continue**;
- For each enabled transition t at M do
  - Obtain marking $M' = M \cdot t$
  - If there exists a marking M'' on the path from the root to M s.t. $M'(p) \geq M''(p)$ for each place p and $M' \neq M''$, replace $M'(p)$ with ω for p where $M'(p) > M''(p)$.
  - Introduce M' as a node, draw an arc with label t from M to M' and tag M' *new*.

Coverability and Reachability Graph

Cover: for a Petri net, given two markings M and M’,
M covers M’ if for each place p, M(p) >= M’(p).

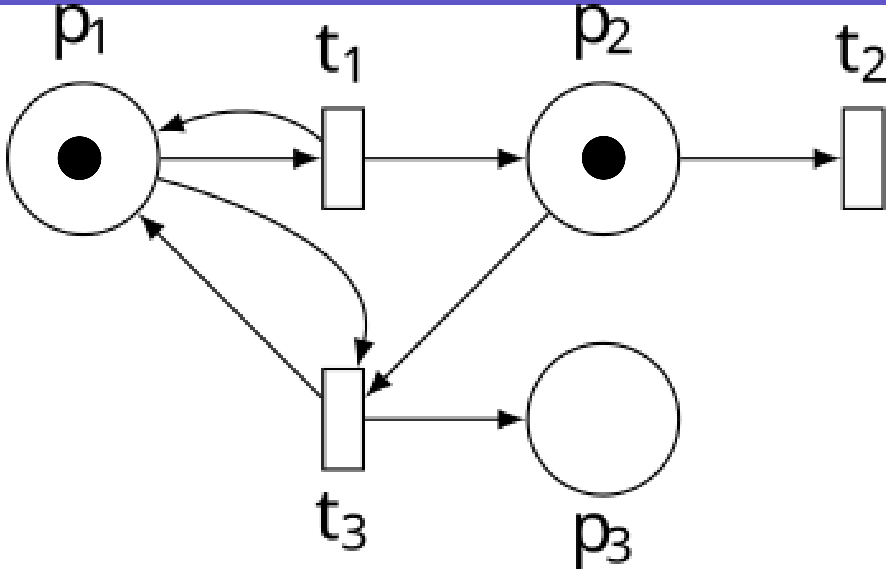
[1, 0, 0]

t1 ↓

Enabled: t1, t2, t3 [1, ω, 0]

↓

Your turn! Complete the rest of the coverability tree



Coverability Tree – Algorithm

Special symbol ω, similar to ∞: $\forall n \in \mathbf{N}$: $\omega > n$; $\omega = \omega \pm n$; $\omega \geq \omega$

Label initial marking M_0 as root and tag it as *new*

while *new* tags exist, pick one, say M

- Remove tag *new* from M;
- If M is identical to an already existing marking, tag it as *old*; **continue**;
- If no transitions are enabled at M, tag it as *deadlock*; **continue**;
- For each enabled transition t at M do
  - Obtain marking M' = M · t
  - If there exists a marking M'' on the path from the root to M s.t. M'(*p*) ≥ M''(*p*) for each place p and M' ≠ M'', replace M'(*p*) with ω for *p* where M'(*p*) > M''(*p*).
  - Introduce M' as a node, draw an arc with label t from M to M' and tag M' *new*.

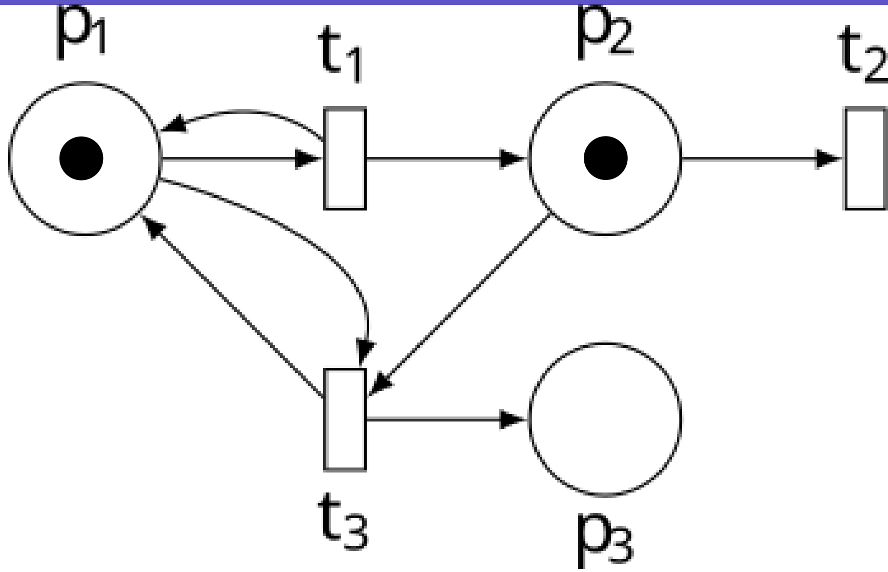
Coverability and Reachability Graph

Cover: for a Petri net, given two markings M and M',
M covers M' if for each place p, M(p) >= M'(p).

[1, 0, 0]

↓ t1

Enabled: t1, t2, t3 — [1, ω, 0]



Coverability Tree – Algorithm

Special symbol ω, similar to ∞: $\forall n \in \mathbf{N}$: $\omega > n$; $\omega = \omega \pm n$; $\omega \geq \omega$

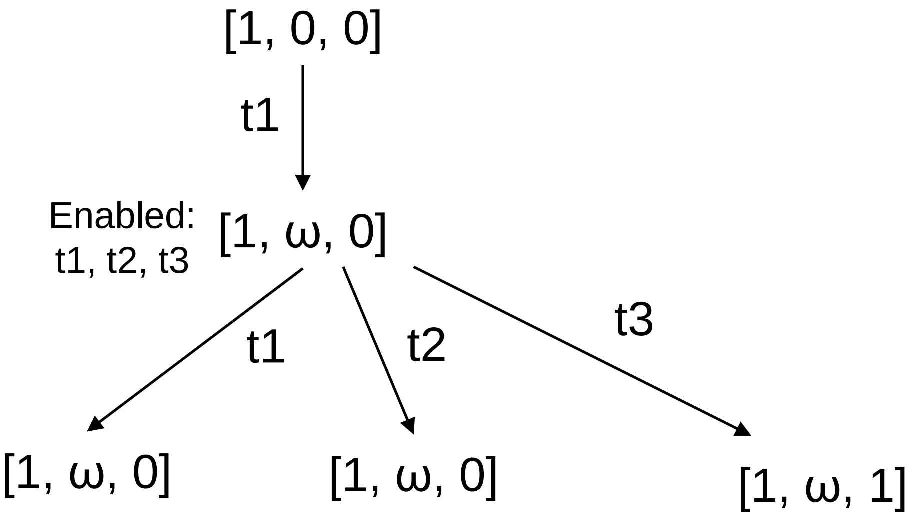
Label initial marking M_0 as root and tag it as *new*

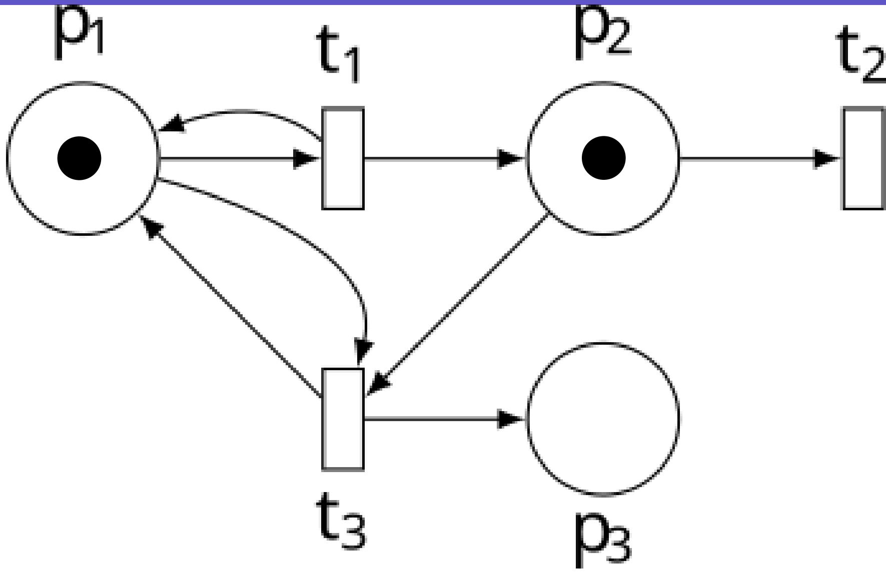
while *new* tags exist, pick one, say M

- Remove tag *new* from M;
- If M is identical to an already existing marking, tag it as *old*; **continue**;
- If no transitions are enabled at M, tag it as *deadlock*; **continue**;
- For each enabled transition t at M do
  - Obtain marking M' = M · t
  - If there exists a marking M'' on the path from the root to M s.t. M'(*p*) ≥ M''(*p*) for each place p and M' ≠ M'', replace M'(*p*) with ω for *p* where M'(*p*) > M''(*p*).
  - Introduce M' as a node, draw an arc with label t from M to M' and tag M' *new*.

Coverability and Reachability Graph

Cover: for a Petri net, given two markings M and M',
M covers M' if for each place p, M(p) >= M'(p).





Coverability Tree – Algorithm

Special symbol ω, similar to ∞: $\forall n \in \mathbf{N}$: $\omega > n$; $\omega = \omega \pm n$; $\omega \geq \omega$

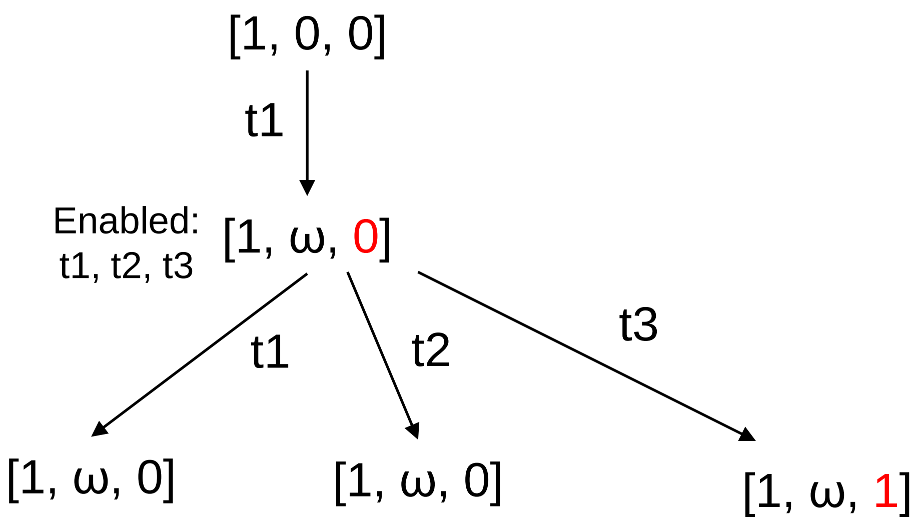
Label initial marking M_0 as root and tag it as *new*

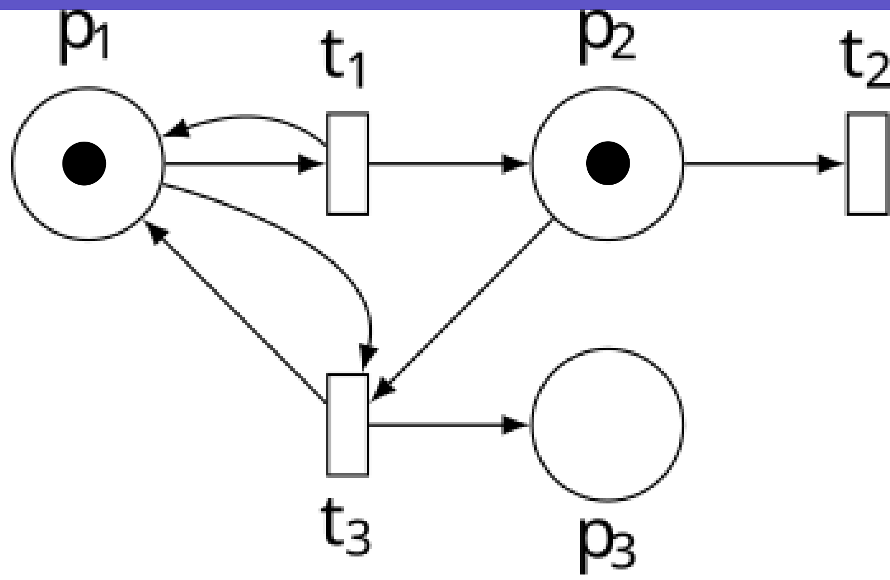
while *new* tags exist, pick one, say M

- Remove tag *new* from M;
- If M is identical to an already existing marking, tag it as *old*; **continue**;
- If no transitions are enabled at M, tag it as *deadlock*; **continue**;
- For each enabled transition t at M do
  - Obtain marking $M' = M \cdot t$
  - If there exists a marking M'' on the path from the root to M s.t. $M'(p) \geq M''(p)$ for each place p and $M' \neq M''$, replace $M'(p)$ with ω for *p* where $M'(p) > M''(p)$.
  - Introduce M' as a node, draw an arc with label t from M to M' and tag M' *new*.

Coverability and Reachability Graph

Cover: for a Petri net, given two markings M and M’, **M covers M’** if for each place p, M(p) >= M’(p).





Coverability Tree – Algorithm

Special symbol ω, similar to ∞: ∀n∈**N**: ω > n; ω = ω ± n; ω ≥ ω

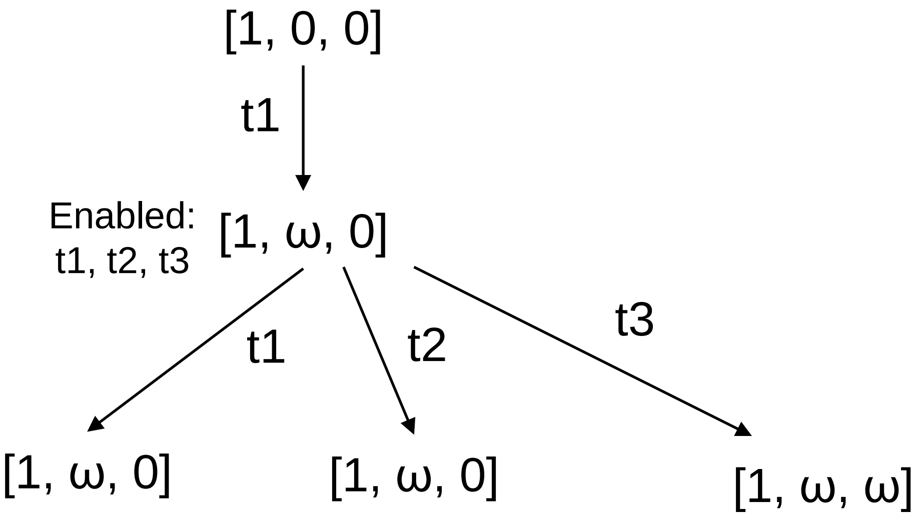
Label initial marking M_0 as root and tag it as *new*

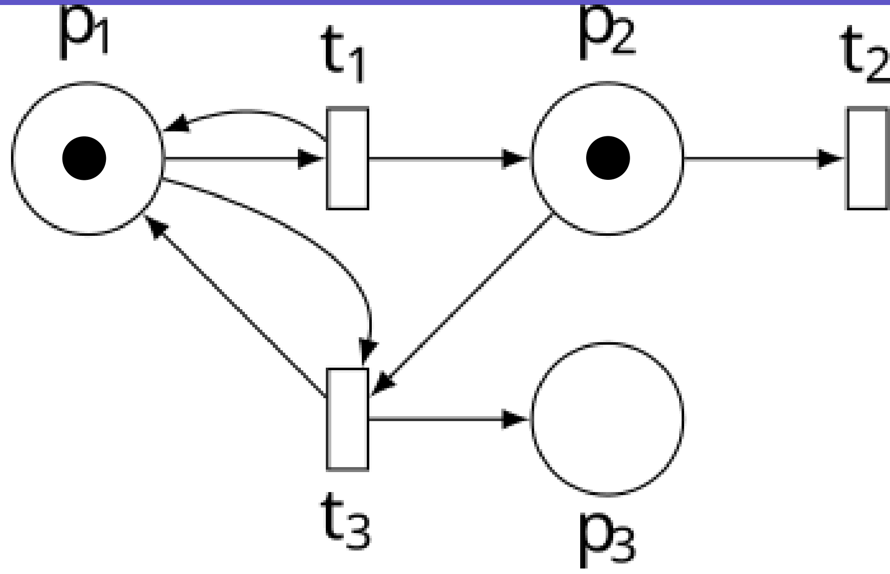
while *new* tags exist, pick one, say M

- Remove tag *new* from M;
- If M is identical to an already existing marking, tag it as *old*; **continue**;
- If no transitions are enabled at M, tag it as *deadlock*; **continue**;
- For each enabled transition t at M do
  - Obtain marking M' = M · t
  - If there exists a marking M'' on the path from the root to M s.t. M'(*p*) ≥ M''(*p*) for each place p and M' ≠ M'', replace M'(*p*) with ω for *p* where M'(*p*) > M''(*p*).
  - Introduce M' as a node, draw an arc with label t from M to M' and tag M' *new*.

Coverability and Reachability Graph

Cover: for a Petri net, given two markings M and M',
M covers M' if for each place p, M(p) >= M'(p).





Coverability Tree – Algorithm

Special symbol ω, similar to ∞: $\forall n \in \mathbf{N}$: $\omega > n$; $\omega = \omega \pm n$; $\omega \geq \omega$

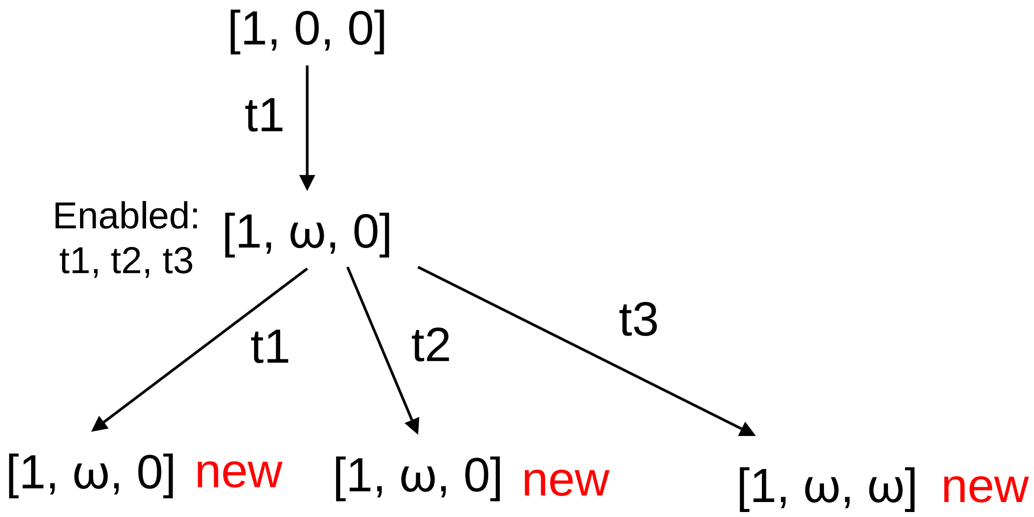
Label initial marking M_0 as root and tag it as *new*

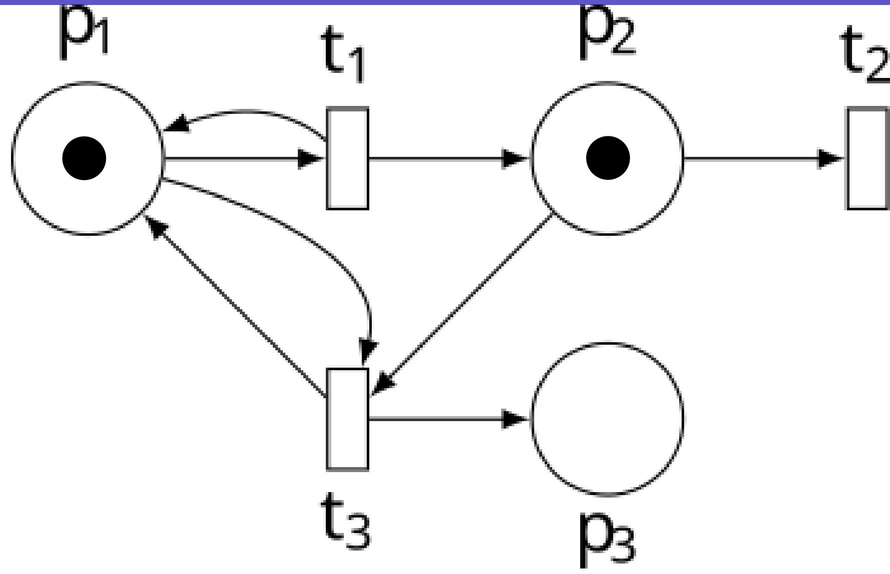
while *new* tags exist, pick one, say M

- Remove tag *new* from M;
- If M is identical to an already existing marking, tag it as *old*; **continue**;
- If no transitions are enabled at M, tag it as *deadlock*; **continue**;
- For each enabled transition t at M do
  - Obtain marking $M' = M \cdot t$
  - If there exists a marking M'' on the path from the root to M s.t. $M'(p) \geq M''(p)$ for each place p and $M' \neq M''$, replace $M'(p)$ with ω for p where $M'(p) > M''(p)$.
  - Introduce M' as a node, draw an arc with label t from M to M' and tag M' *new*.

Coverability and Reachability Graph

Cover: for a Petri net, given two markings M and M’,
M covers M’ if for each place p, M(p) >= M’(p).





Coverability Tree – Algorithm

Special symbol ω, similar to ∞: $\forall n \in \mathbf{N}$: $\omega > n$; $\omega = \omega \pm n$; $\omega \geq \omega$

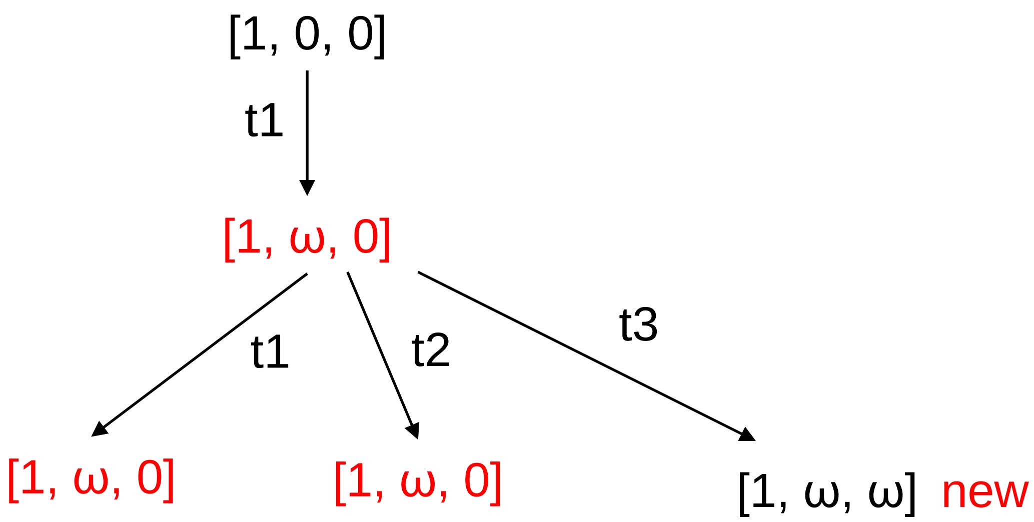
Label initial marking M_0 as root and tag it as *new*

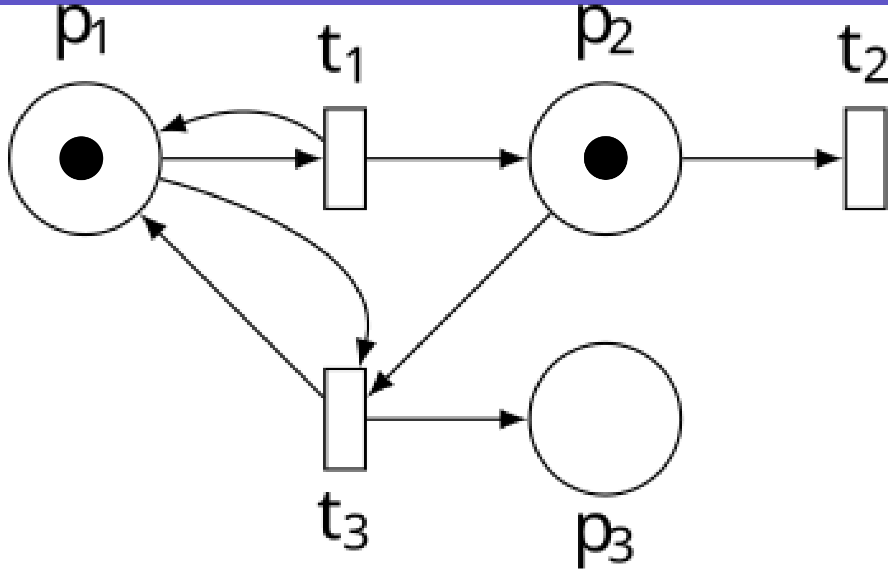
while *new* tags exist, pick one, say M

- Remove tag *new* from M;
- If M is identical to an already existing marking, tag it as *old*; **continue**;
- If no transitions are enabled at M, tag it as *deadlock*; **continue**;
- For each enabled transition t at M do
  - Obtain marking M' = M · t
  - If there exists a marking M'' on the path from the root to M s.t. M'(*p*) ≥ M''(*p*) for each place p and M' ≠ M'', replace M'(*p*) with ω for *p* where M'(*p*) > M''(*p*).
  - Introduce M' as a node, draw an arc with label t from M to M' and tag M' *new*.

Coverability and Reachability Graph

Cover: for a Petri net, given two markings M and M',
M covers M' if for each place p, M(p) >= M'(p).





Coverability Tree – Algorithm

Special symbol ω, similar to ∞: $\forall n \in \mathbf{N}$: $\omega > n$; $\omega = \omega \pm n$; $\omega \geq \omega$

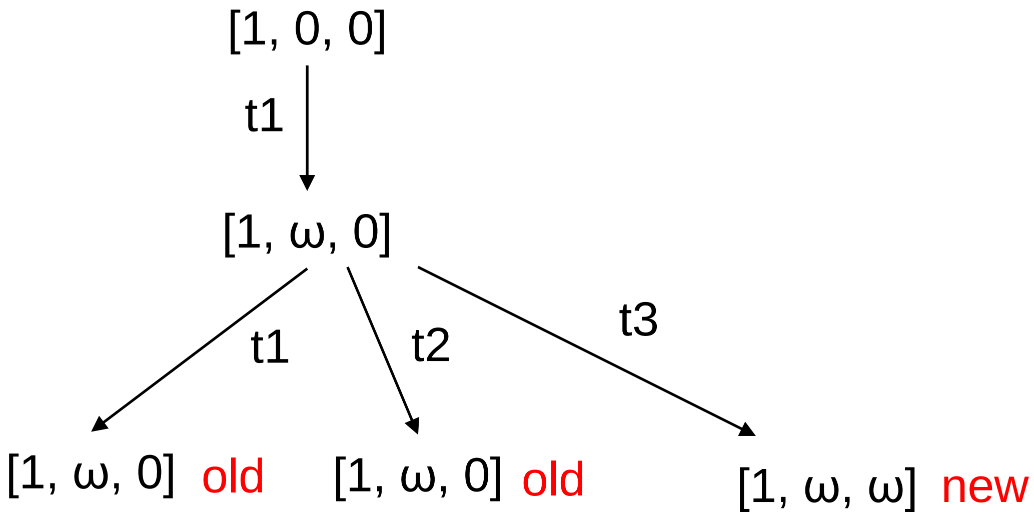
Label initial marking M_0 as root and tag it as *new*

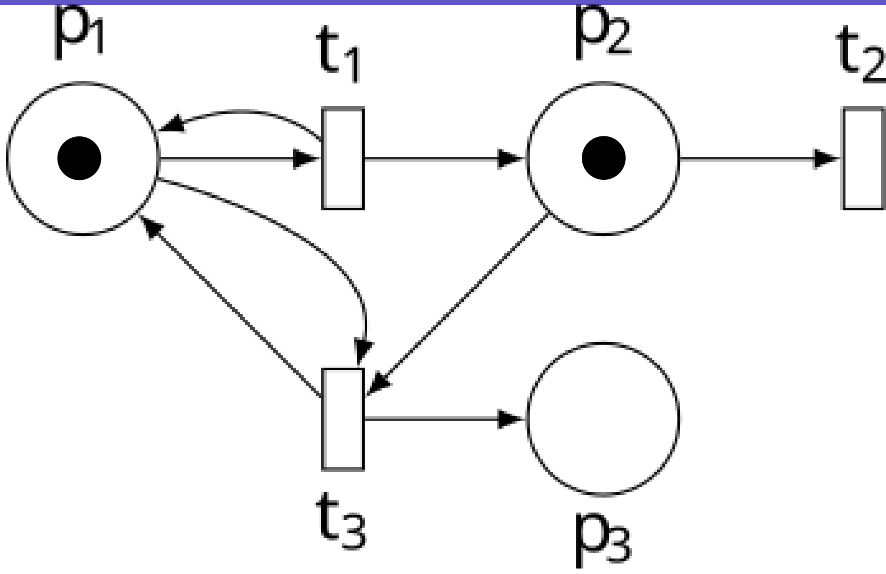
while *new* tags exist, pick one, say M

- Remove tag *new* from M;
- If M is identical to an already existing marking, tag it as *old*; **continue**;
- If no transitions are enabled at M, tag it as *deadlock*; **continue**;
- For each enabled transition t at M do
  - Obtain marking M' = M · t
  - If there exists a marking M'' on the path from the root to M s.t. M'(*p*) ≥ M''(*p*) for each place p and M' ≠ M'', replace M'(*p*) with ω for *p* where M'(*p*) > M''(*p*).
  - Introduce M' as a node, draw an arc with label t from M to M' and tag M' *new*.

Coverability and Reachability Graph

Cover: for a Petri net, given two markings M and M',
M covers M' if for each place p, M(p) >= M'(p).





Coverability Tree – Algorithm

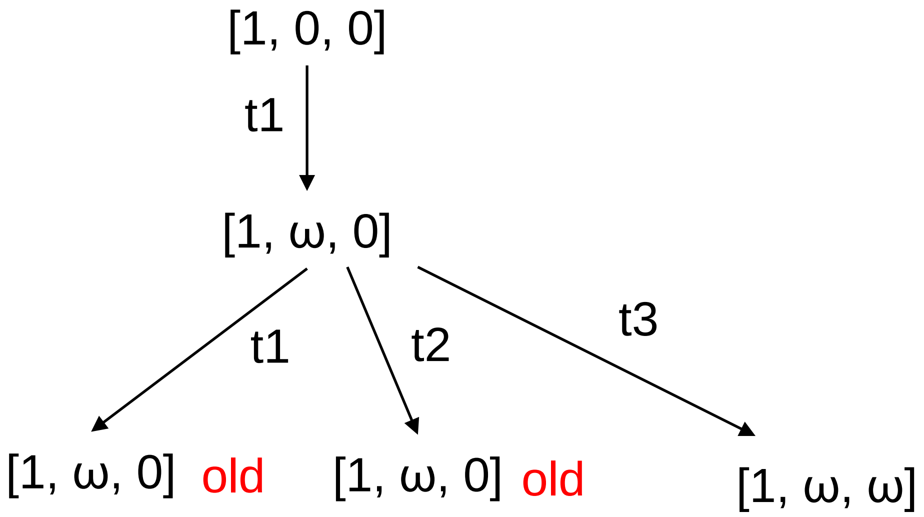
Special symbol ω, similar to ∞: $\forall n \in \mathbf{N}$: $\omega > n$; $\omega = \omega \pm n$; $\omega \geq \omega$

Label initial marking M_0 as root and tag it as *new*

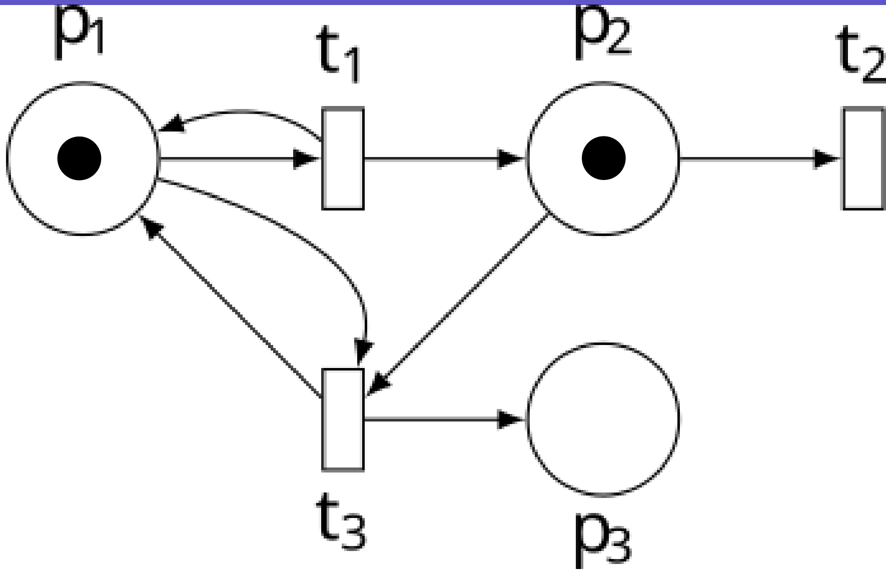
while *new* tags exist, pick one, say M

- Remove tag *new* from M;
- If M is identical to an already existing marking, tag it as *old*; **continue**;
- If no transitions are enabled at M, tag it as *deadlock*; **continue**;
- For each enabled transition t at M do
  - Obtain marking M' = M · t
  - If there exists a marking M'' on the path from the root to M s.t. M'(*p*) ≥ M''(*p*) for each place p and M' ≠ M'', replace M'(*p*) with ω for *p* where M'(*p*) > M''(*p*).
  - Introduce M' as a node, draw an arc with label t from M to M' and tag M' *new*.

Cover: for a Petri net, given two markings M and M',
M covers M' if for each place p, M(p) >= M'(p).



Enabled:
t1, t2, t3



Coverability Tree – Algorithm

Special symbol ω, similar to ∞: $\forall n \in \mathbf{N}$: ω > n; ω = ω ± n; ω ≥ ω

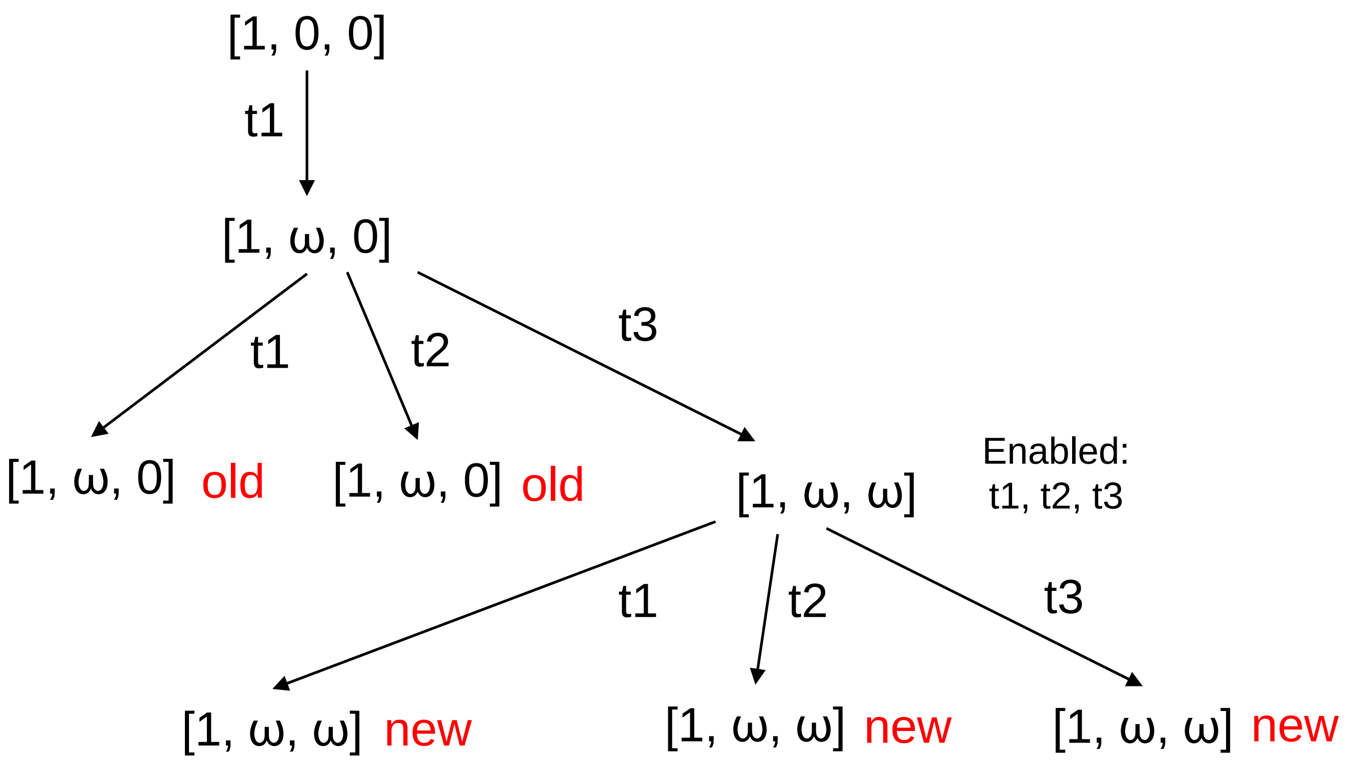
Label initial marking M_0 as root and tag it as *new*

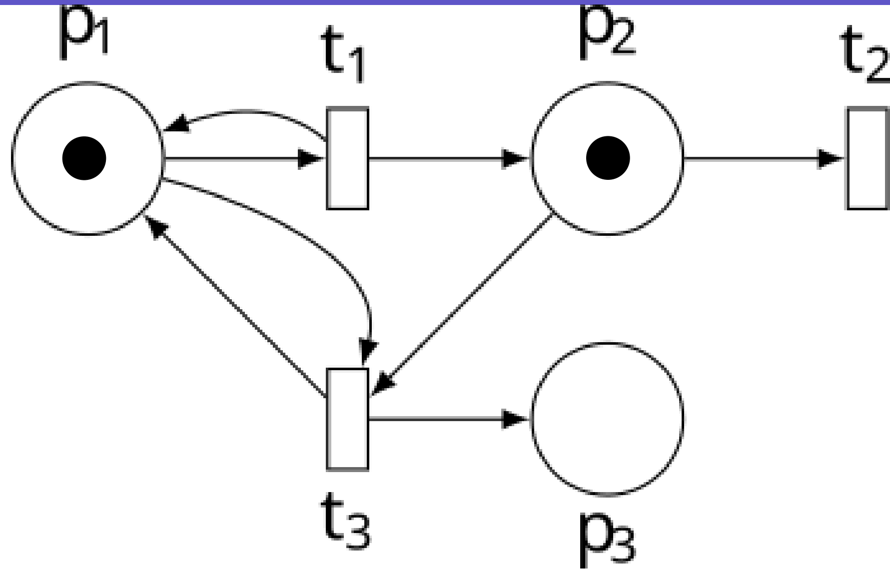
while *new* tags exist, pick one, say M

- Remove tag *new* from M;
- If M is identical to an already existing marking, tag it as *old*; **continue**;
- If no transitions are enabled at M, tag it as *deadlock*; **continue**;
- For each enabled transition t at M do
  - Obtain marking M' = M · t
  - If there exists a marking M'' on the path from the root to M s.t. M'(p) ≥ M''(p) for each place p and M' ≠ M'', replace M'(p) with ω for p where M'(p) > M''(p).
  - Introduce M' as a node, draw an arc with label t from M to M' and tag M' *new*.

Coverability and Reachability Graph

Cover: for a Petri net, given two markings M and M',
M covers M' if for each place p, M(p) >= M'(p).





Coverability Tree – Algorithm

Special symbol ω, similar to ∞: ∀n∈**N**: ω > n; ω = ω ± n; ω ≥ ω

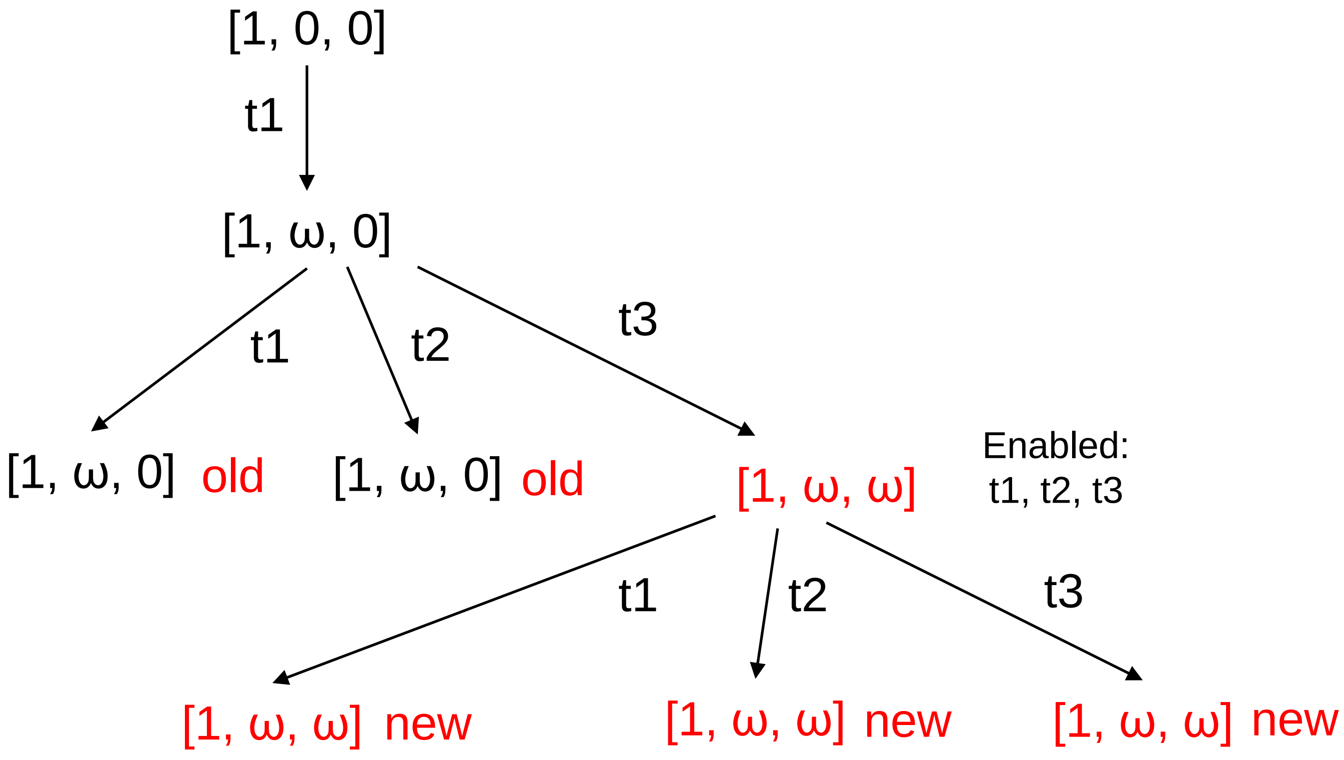
Label initial marking M_0 as root and tag it as *new*

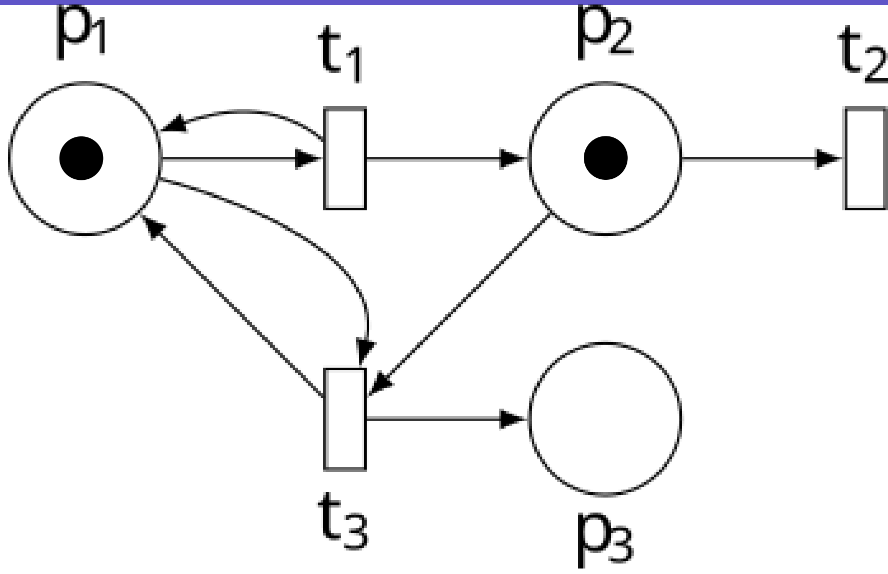
while *new* tags exist, pick one, say M

- Remove tag *new* from M;
- If M is identical to an already existing marking, tag it as *old*; **continue**;
- If no transitions are enabled at M, tag it as *deadlock*; **continue**;
- For each enabled transition t at M do
  - Obtain marking M' = M · t
  - If there exists a marking M'' on the path from the root to M s.t. M'(*p*) ≥ M''(*p*) for each place p and M' ≠ M'', replace M'(*p*) with ω for *p* where M'(*p*) > M''(*p*).
  - Introduce M' as a node, draw an arc with label t from M to M' and tag M' *new*.

Coverability and Reachability Graph

Cover: for a Petri net, given two markings M and M’,
M covers M’ if for each place p, M(p) >= M’(p).





Coverability Tree – Algorithm

Special symbol ω, similar to ∞: $\forall n \in \mathbf{N}$: $\omega > n$; $\omega = \omega \pm n$; $\omega \geq \omega$

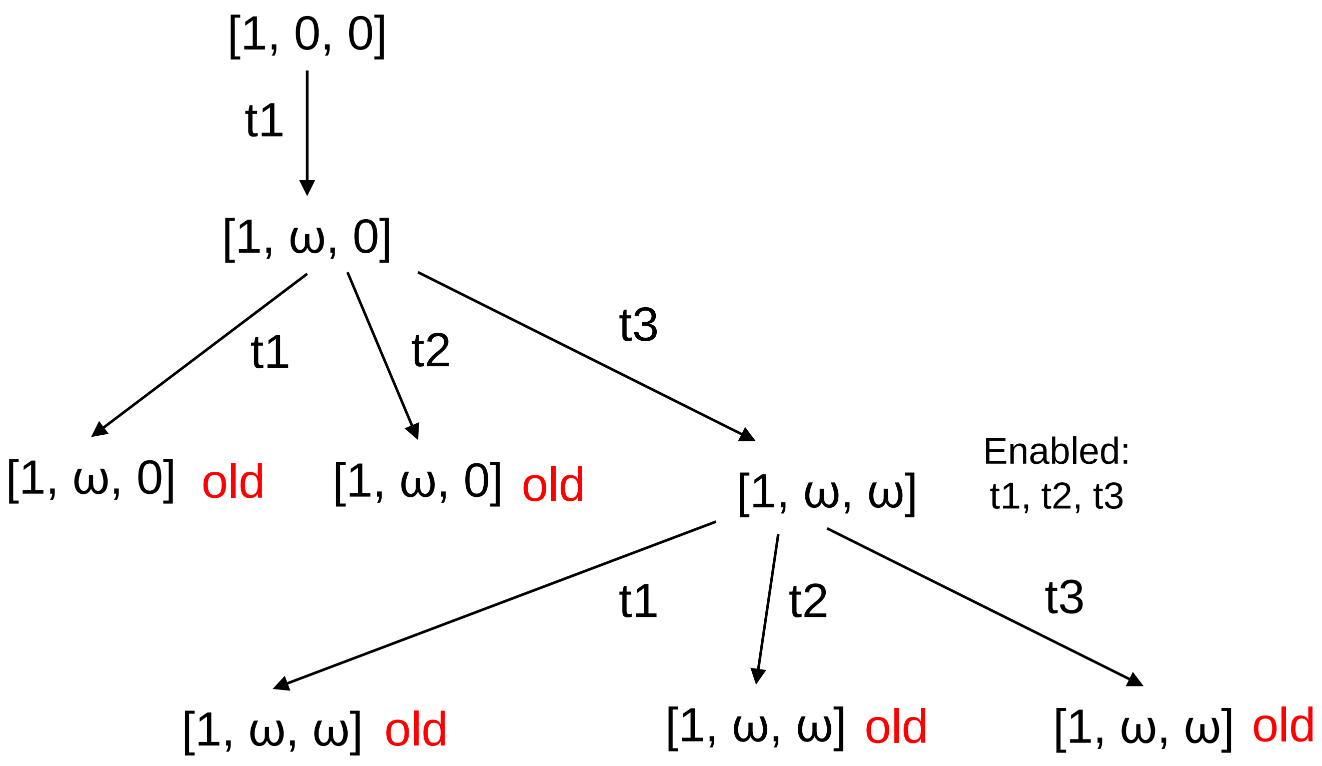
Label initial marking M_0 as root and tag it as *new*

while *new* tags exist, pick one, say M

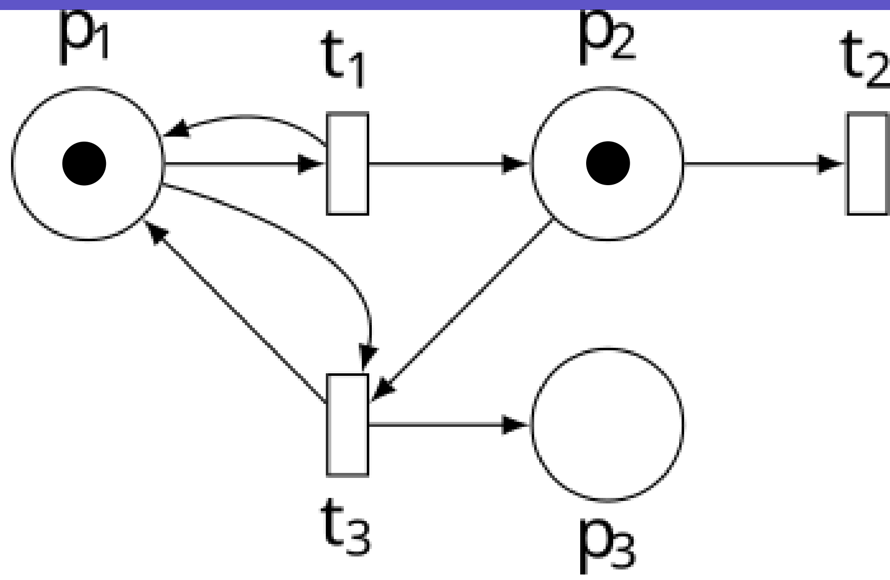
- Remove tag *new* from M;
- If M is identical to an already existing marking, tag it as *old*; **continue**;
- If no transitions are enabled at M, tag it as *deadlock*; **continue**;
- For each enabled transition t at M do
  - Obtain marking M' = M · t
  - If there exists a marking M'' on the path from the root to M s.t. M'(*p*) ≥ M''(*p*) for each place p and M' ≠ M'', replace M'(*p*) with ω for *p* where M'(*p*) > M''(*p*).
  - Introduce M' as a node, draw an arc with label t from M to M' and tag M' *new*.

Coverability and Reachability Graph

Cover: for a Petri net, given two markings M and M', **M covers M'** if for each place p, M(p) >= M'(p).



The coverability tree



Coverability Tree – Algorithm

Special symbol ω, similar to ∞: $\forall n \in \mathbf{N}$: $\omega > n$; $\omega = \omega \pm n$; $\omega \geq \omega$

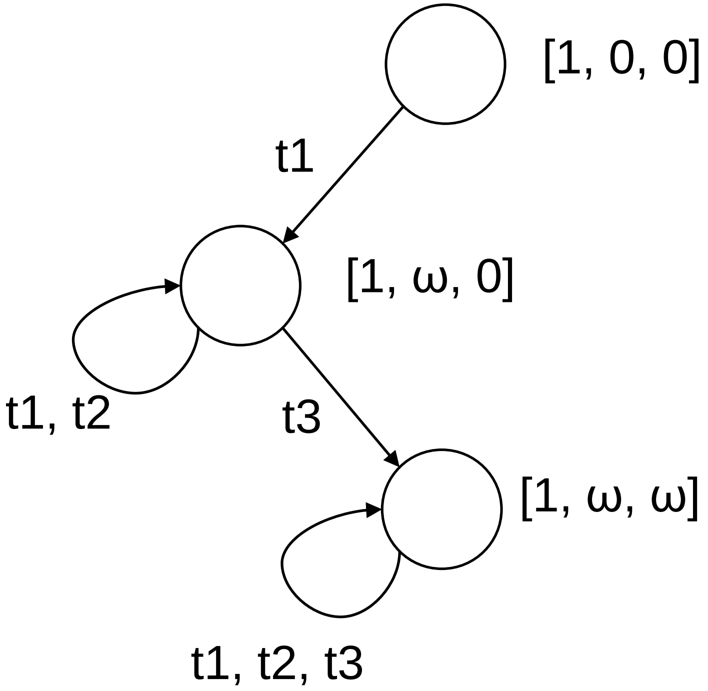
Label initial marking M_0 as root and tag it as *new*

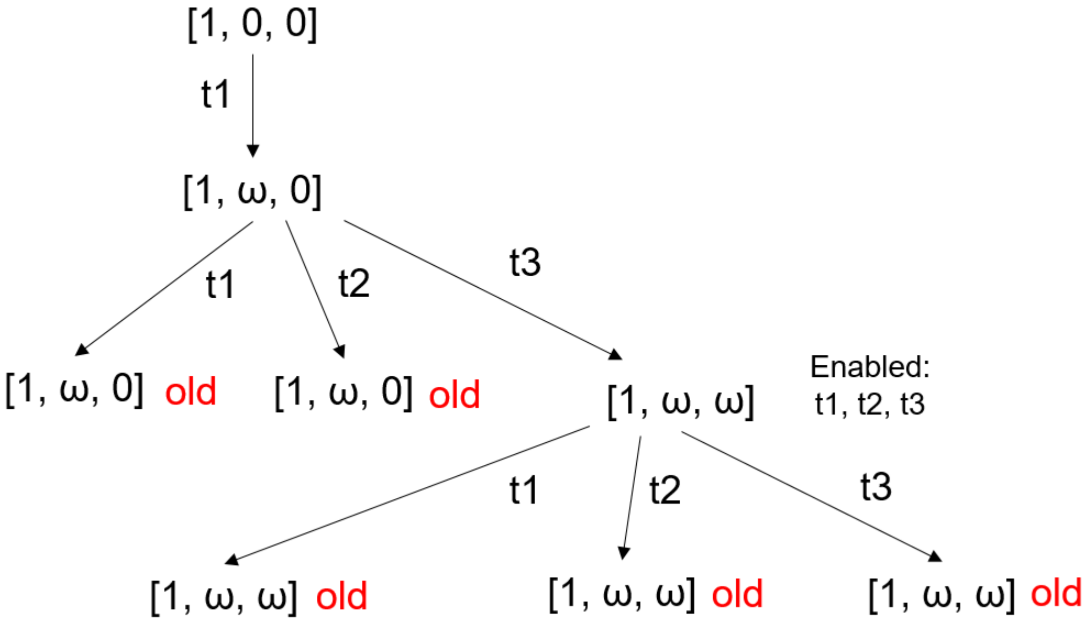
while *new* tags exist, pick one, say M

- Remove tag *new* from M;
- If M is identical to an already existing marking, tag it as *old*; **continue**;
- If no transitions are enabled at M, tag it as *deadlock*; **continue**;
- For each enabled transition t at M do
  - Obtain marking M' = M · t
  - If there exists a marking M'' on the path from the root to M s.t. M'(*p*) ≥ M''(*p*) for each place p and M' ≠ M'', replace M'(*p*) with ω for *p* where M'(*p*) > M''(*p*).
  - Introduce M' as a node, draw an arc with label t from M to M' and tag M' *new*.

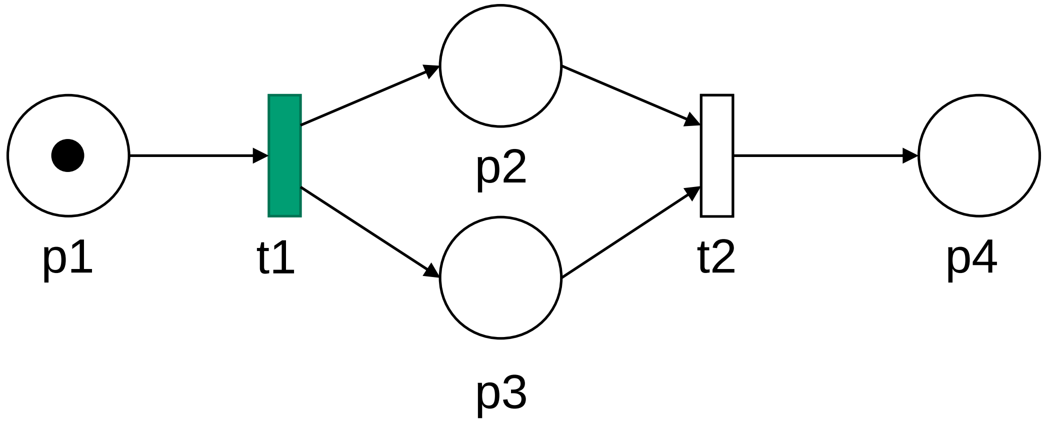
Coverability and Reachability Graph

Coverability graph: obtained by merging identical nodes and combine edges that connect the same nodes in the tree





Incidence Matrix



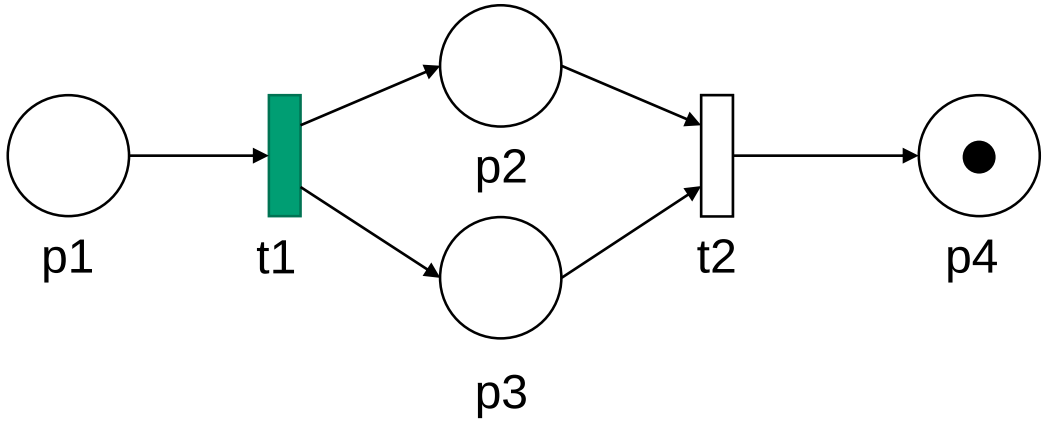
Use vector addition to represent "we fire t1" once

$$M_0 + u_1 = \begin{bmatrix} 1 \\ 0 \\ 0 \\ 0 \end{bmatrix} + \begin{bmatrix} -1 \\ +1 \\ +1 \\ 0 \end{bmatrix}$$

Transition vector u1

Both marking and firing can be represented using vectors

Incidence Matrix



Use vector addition to represent "we fire t1" once

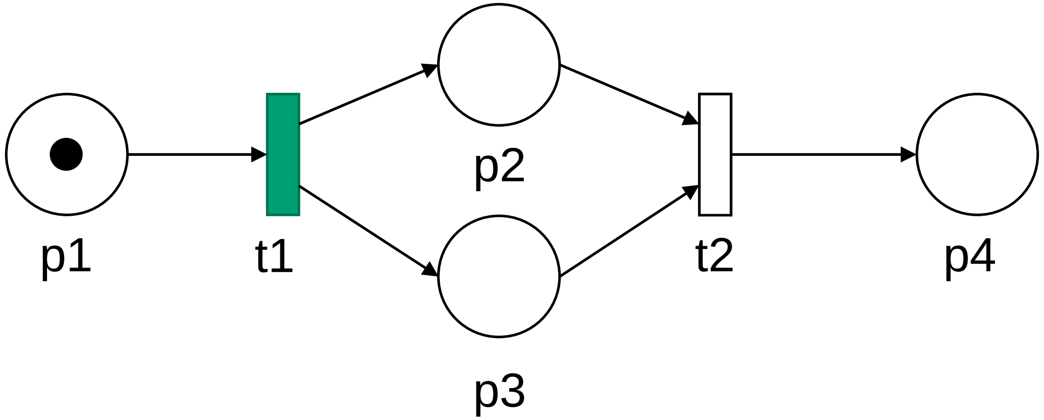
$$M_2 + u_1 = \begin{bmatrix} 0 \\ 0 \\ 0 \\ 1 \end{bmatrix} + \begin{bmatrix} -1 \\ +1 \\ +1 \\ 0 \end{bmatrix}$$

Transition vector u2

We can write it out as additions between vectors, but does it make sense?

Both marking and firing can be represented using vectors

Incidence Matrix

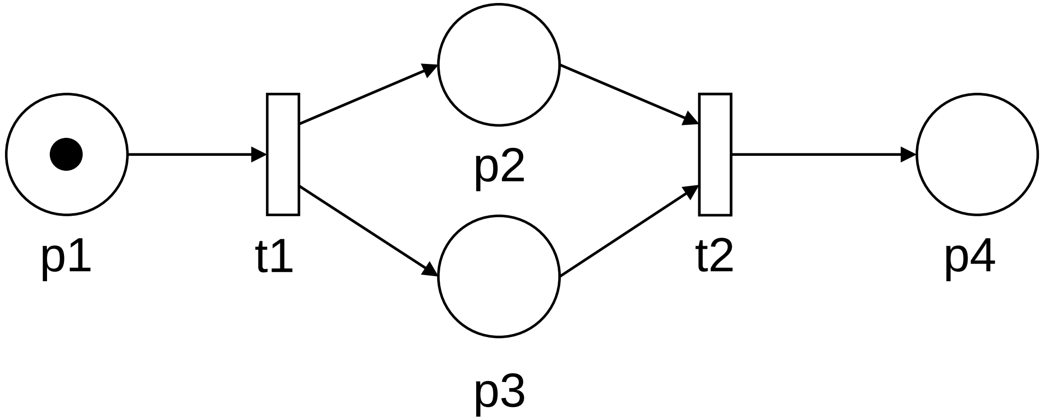


Use vector addition to represent
"we fire t1" x1 times

$$M_0 + u_1 \cdot x_1 = \begin{bmatrix} 1 \\ 0 \\ 0 \\ 0 \end{bmatrix} + \begin{bmatrix} -1 \\ +1 \\ +1 \\ 0 \end{bmatrix} \cdot x_1$$

Both marking and firing can be represented using vectors

Incidence Matrix



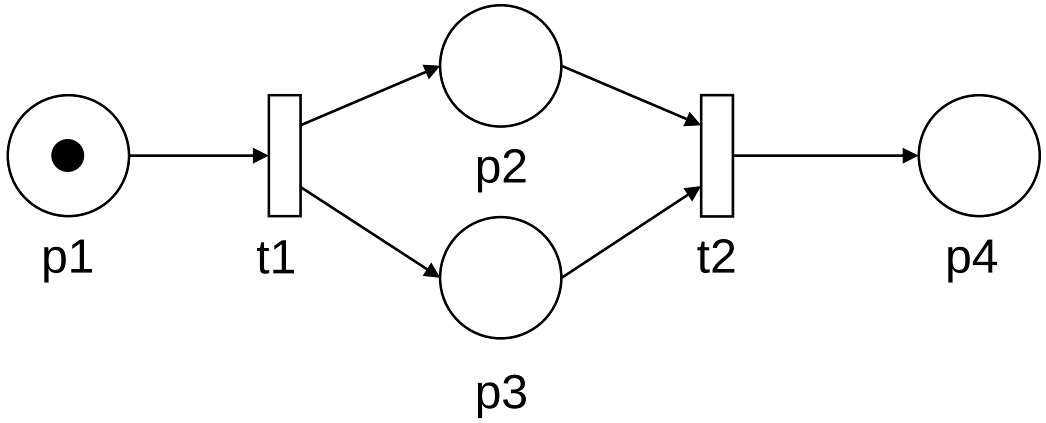
Use vector addition to represent "we fire t1" x1 times and "fire t2" x2 times

$$M_0 + u_1 \cdot x_1 + u_2 \cdot x_2 = \begin{bmatrix} 1 \\ 0 \\ 0 \\ 0 \end{bmatrix} + \begin{bmatrix} -1 \\ +1 \\ +1 \\ 0 \end{bmatrix} \cdot x_1 + \begin{bmatrix} 0 \\ -1 \\ -1 \\ +1 \end{bmatrix} \cdot x_2$$

Linear combinations of vectors can be denoted as a matrix

Both marking and firing can be represented using vectors

Incidence Matrix

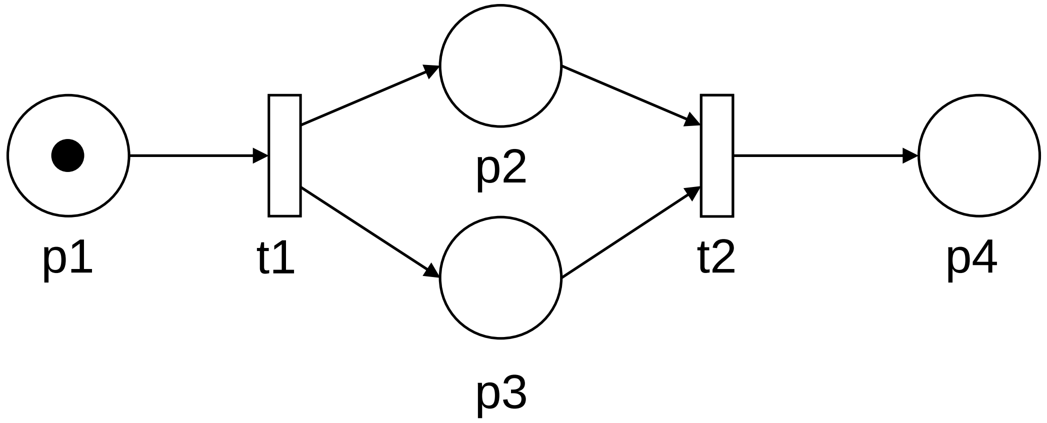


Use vector addition to represent "we fire t1" x1 times and "fire t2" x2 times

$$M_0 + A \cdot \vec{x} = \begin{bmatrix} 1 \\ 0 \\ 0 \\ 0 \end{bmatrix} + \underbrace{\begin{bmatrix} -1 & 0 \\ +1 & -1 \\ +1 & -1 \\ 0 & +1 \end{bmatrix} \cdot \begin{bmatrix} x_1 \\ x_2 \end{bmatrix}}_{\text{The incidence matrix}}$$

Both marking and firing can be represented using vectors

Incidence Matrix



Use vector addition to represent "we fire t1" x1 times and "fire t2" x2 times

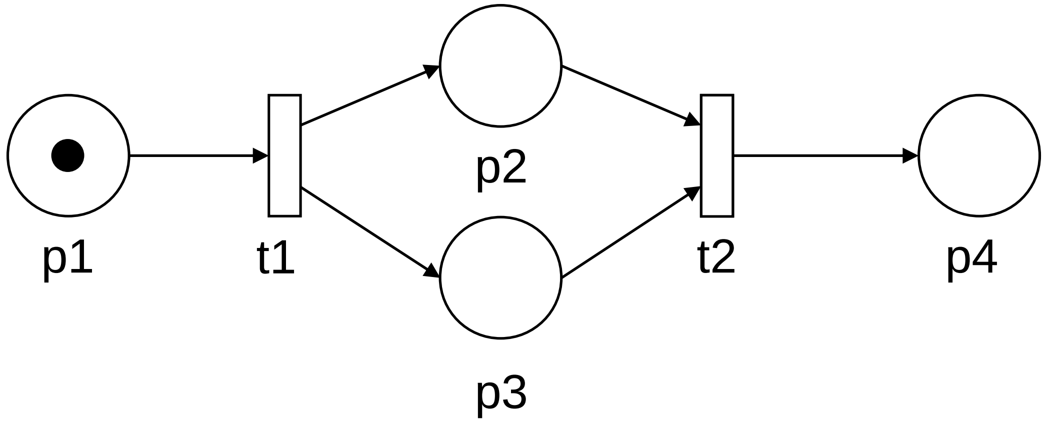
Columns: the effect of firing the transitions

$$M_0 + A \cdot \vec{x} = \begin{bmatrix} 1 \\ 0 \\ 0 \\ 0 \end{bmatrix} + \begin{bmatrix} -1 & 0 \\ +1 & -1 \\ +1 & -1 \\ 0 & +1 \end{bmatrix} \cdot \begin{bmatrix} x_1 \\ x_2 \end{bmatrix}$$

The incidence matrix

Both marking and firing can be represented using vectors

Incidence Matrix



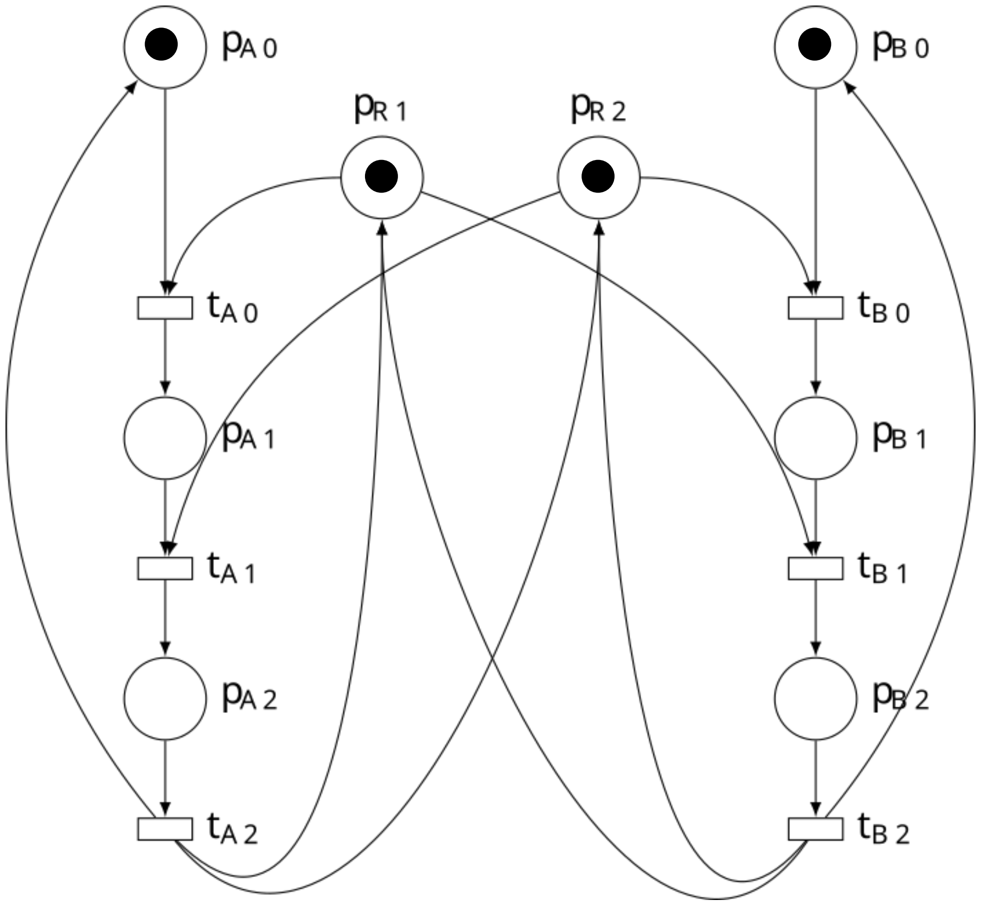
$$M_0 + A^+ \cdot \vec{x} - A^- \cdot \vec{x} = \begin{bmatrix}1\\0\\0\\0\end{bmatrix} + \begin{bmatrix}0 & 0\\1 & 0\\1 & 0\\0 & 1\end{bmatrix} \cdot \begin{bmatrix}x_1\\x_2\end{bmatrix} - \begin{bmatrix}1 & 0\\0 & 1\\0 & 1\\0 & 0\end{bmatrix} \cdot \begin{bmatrix}x_1\\x_2\end{bmatrix}$$

The downstream incidence matrix A^+

The upstream incidence matrix A^-

Both marking and firing can be represented using vectors

Incidence Matrix

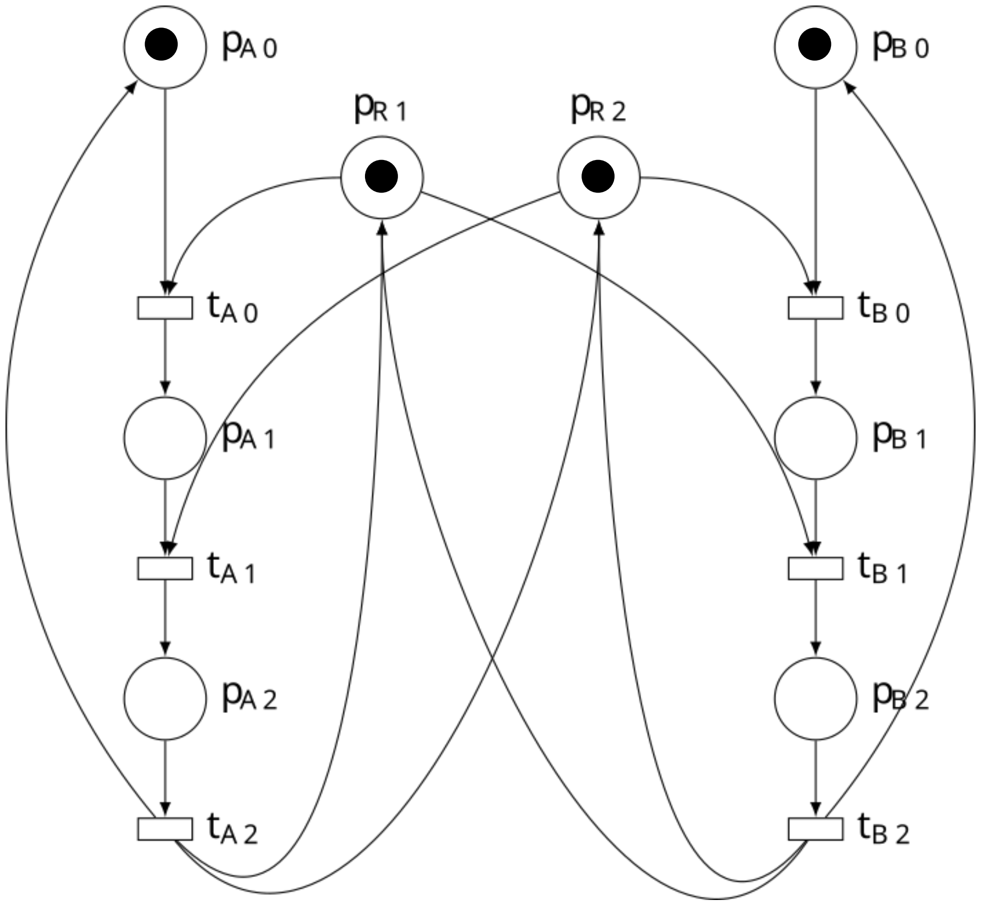


M $=(P_{A0}, P_{A1}, P_{A2}, P_{R1}, P_{R2}, P_{B0}, P_{B1}, P_{B2})$

x $= (t_{A0}, t_{A1}, t_{A2}, t_{B0}, t_{B1}, t_{B2})$

Incidence Matrix

(a) Construct a reachability graph, and determine the deadlock state (no transition is enabled in a deadlock state).
(b) Determine the upstream and downstream incidence matrices A^+ and A^- and the incidence matrix A. What is the marking you obtain by firing $t_{A0}$and t_{B0}?
(c) Can you show why the marking after firing $t_{A0}$and t_{B0} is a deadlock state by using the upstream incidence matrix A^-?
(d) Can we make the Petri net deadlock-free by adding one place and a few transitions?



$$M_0 + A^+ \cdot \vec{x} - A^- \cdot \vec{x} = \begin{bmatrix}1\\0\\0\\0\end{bmatrix} + \underbrace{\begin{bmatrix}0 & 0\\1 & 0\\1 & 0\\0 & 1\end{bmatrix}}_{\text{The upstream incidence matrix } A^+} \cdot \begin{bmatrix}x_1\\x_2\end{bmatrix} - \underbrace{\begin{bmatrix}1 & 0\\0 & 1\\0 & 1\\0 & 0\end{bmatrix}}_{\text{The downstream incidence matrix } A^-} \cdot \begin{bmatrix}x_1\\x_2\end{bmatrix}$$

Both marking and firing can be represented using vectors

M $=(P_{A0}, P_{A1}, P_{A2}, P_{R1}, P_{R2}, P_{B0}, P_{B1}, P_{B2})$

x $= (t_{A0}, t_{A1}, t_{A2}, t_{B0}, t_{B1}, t_{B2})$

Incidence Matrix

matrix A^+

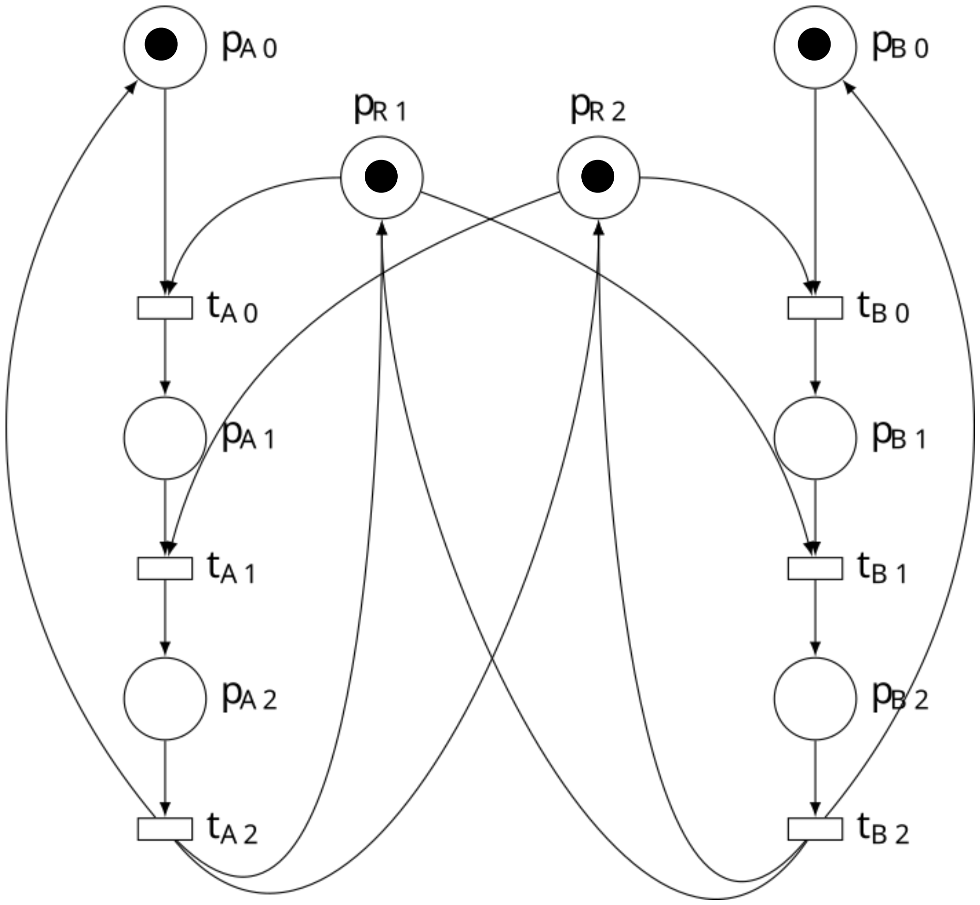
$$
\begin{bmatrix}
0 & 0 & 1 & 0 & 0 & 0 \\
1 & 0 & 0 & 0 & 0 & 0 \\
0 & 1 & 0 & 0 & 0 & 0 \\
0 & 0 & 1 & 0 & 0 & 1 \\
0 & 0 & 1 & 0 & 0 & 1 \\
0 & 0 & 0 & 0 & 0 & 1 \\
0 & 0 & 0 & 1 & 0 & 0 \\
0 & 0 & 0 & 0 & 1 & 0
\end{bmatrix}
$$

matrix A^-

$$
\begin{bmatrix}
1 & 0 & 0 & 0 & 0 & 0 \\
0 & 1 & 0 & 0 & 0 & 0 \\
0 & 0 & 1 & 0 & 0 & 0 \\
1 & 0 & 0 & 0 & 1 & 0 \\
0 & 1 & 0 & 1 & 0 & 0 \\
0 & 0 & 0 & 1 & 0 & 0 \\
0 & 0 & 0 & 0 & 1 & 0 \\
0 & 0 & 0 & 0 & 0 & 1
\end{bmatrix}
$$

matrix A

$$
\begin{bmatrix}
-1 & 0 & 1 & 0 & 0 & 0 \\
1 & -1 & 0 & 0 & 0 & 0 \\
0 & 1 & -1 & 0 & 0 & 0 \\
-1 & 0 & 1 & 0 & -1 & 1 \\
0 & -1 & 1 & -1 & 0 & 1 \\
0 & 0 & 0 & -1 & 0 & 1 \\
0 & 0 & 0 & 1 & -1 & 0 \\
0 & 0 & 0 & 0 & 1 & -1
\end{bmatrix}
$$



M $=(P_{A0}, P_{A1}, P_{A2}, P_{R1}, P_{R2}, P_{B0}, P_{B1}, P_{B2})$

x $= (t_{A0}, t_{A1}, t_{A2}, t_{B0}, t_{B1}, t_{B2})$

Incidence Matrix

matrix A^+

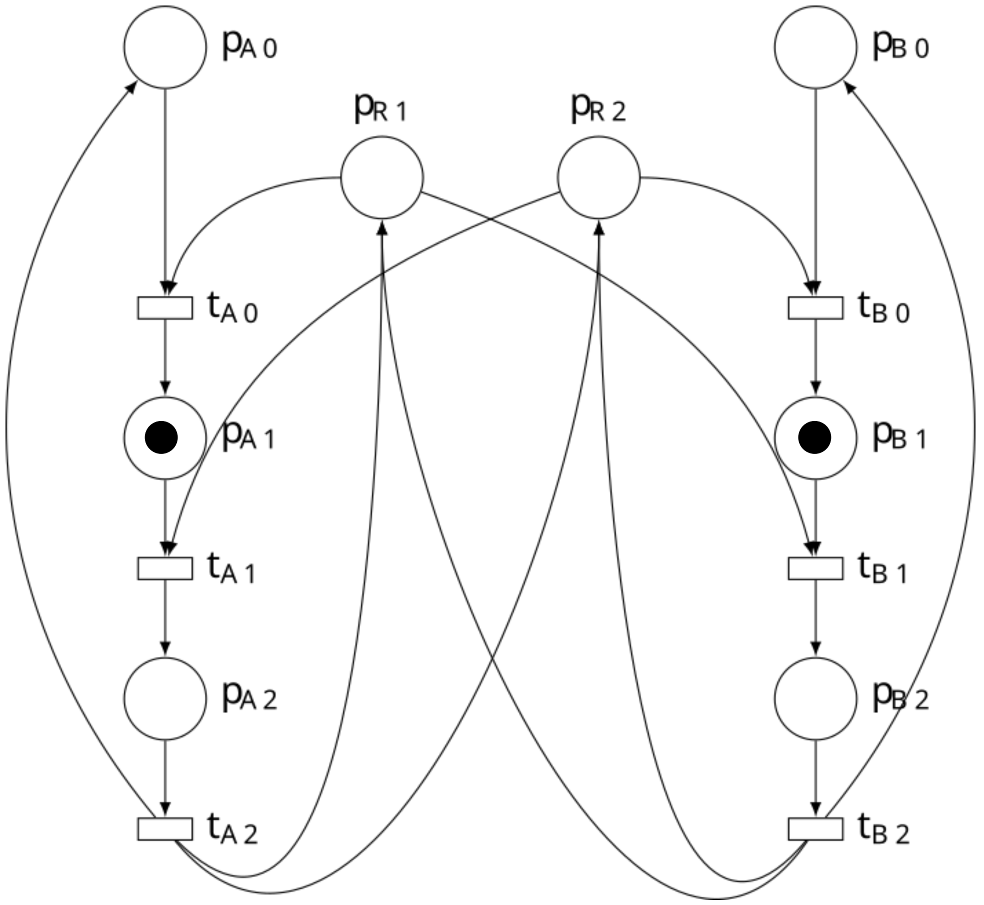
$$\begin{bmatrix} 0 & 0 & 1 & 0 & 0 & 0 \\ 1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 & 1 \\ 0 & 0 & 1 & 0 & 0 & 1 \\ 0 & 0 & 0 & 0 & 0 & 1 \\ 0 & 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 0 \end{bmatrix}$$

matrix A^-

$$\begin{bmatrix} 1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 & 0 \\ 1 & 0 & 0 & 0 & 1 & 0 \\ 0 & 1 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 \end{bmatrix}$$

matrix A

$$\begin{bmatrix} -1 & 0 & 1 & 0 & 0 & 0 \\ 1 & -1 & 0 & 0 & 0 & 0 \\ 0 & 1 & -1 & 0 & 0 & 0 \\ -1 & 0 & 1 & 0 & -1 & 1 \\ 0 & -1 & 1 & -1 & 0 & 1 \\ 0 & 0 & 0 & -1 & 0 & 1 \\ 0 & 0 & 0 & 1 & -1 & 0 \\ 0 & 0 & 0 & 0 & 1 & -1 \end{bmatrix}$$



$$\begin{bmatrix} 0 \\ 1 \\ 0 \\ 0 \\ 0 \\ 0 \\ 1 \\ 0 \end{bmatrix}$$

Deadlock state

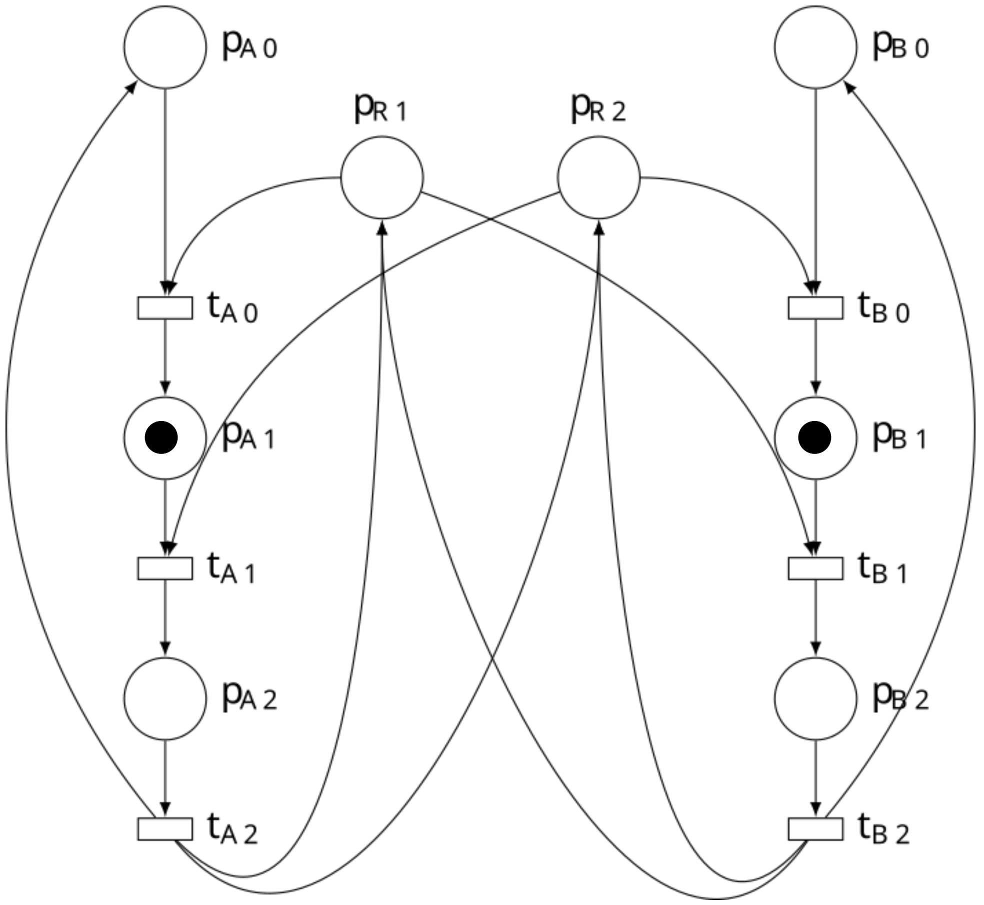
This state is a deadlock state because it is not covered by any vectors in A

M $=(P_{A0}, P_{A1}, P_{A2}, P_{R1}, P_{R2}, P_{B0}, P_{B1}, P_{B2})$

x $= (t_{A0}, t_{A1}, t_{A2}, t_{B0}, t_{B1}, t_{B2})$

Incidence Matrix

This marking has to be avoided



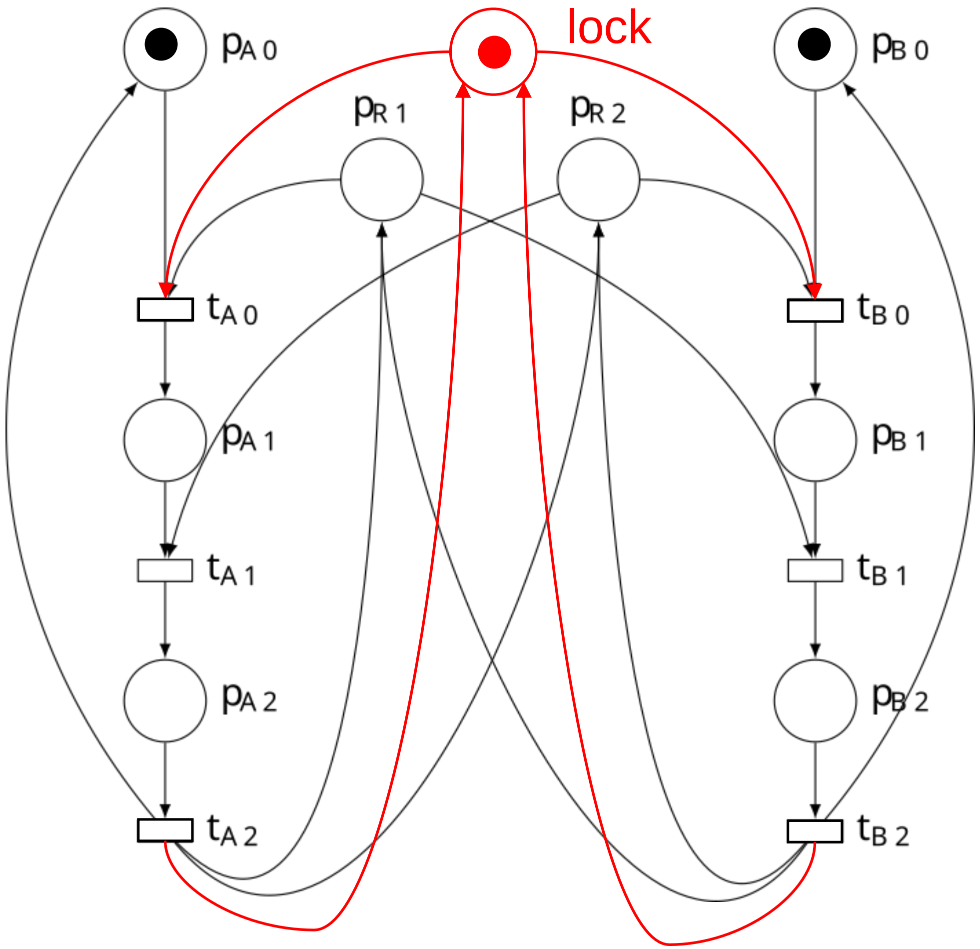
M $=(P_{A0}, P_{A1}, P_{A2}, P_{R1}, P_{R2}, P_{B0}, P_{B1}, P_{B2})$

x $= (t_{A0}, t_{A1}, t_{A2}, t_{B0}, t_{B1}, t_{B2})$

Incidence Matrix

This marking has to be avoided

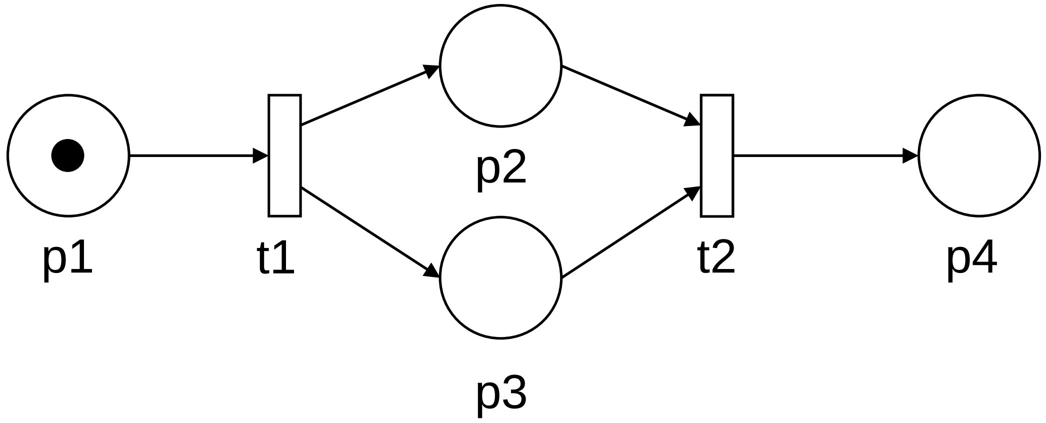
Solution: add a lock



M =$(P_{A0}, P_{A1}, P_{A2}, P_{R1}, P_{R2}, P_{B0}, P_{B1}, P_{B2})$

x = $(t_{A0}, t_{A1}, t_{A2}, t_{B0}, t_{B1}, t_{B2})$

How to prove that a marking is reachable?



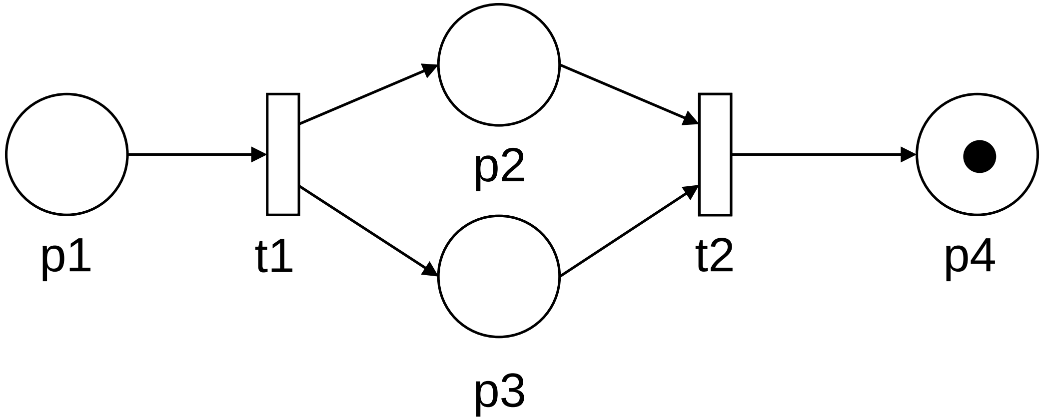
$$M_1 = M_0 + A \cdot \vec{x} = \begin{bmatrix} 1 \\ 0 \\ 0 \\ 0 \end{bmatrix} + \begin{bmatrix} -1 & 0 \\ +1 & -1 \\ +1 & -1 \\ 0 & +1 \end{bmatrix} \cdot \begin{bmatrix} x_1 \\ x_2 \end{bmatrix}$$

If a marking M1 is reachable, then the state equation has a non-negative solution $\vec{x}$

If the state equation has no non-negative solution $\vec{x}$, then marking M1 is not reachable

But not true vice versa!

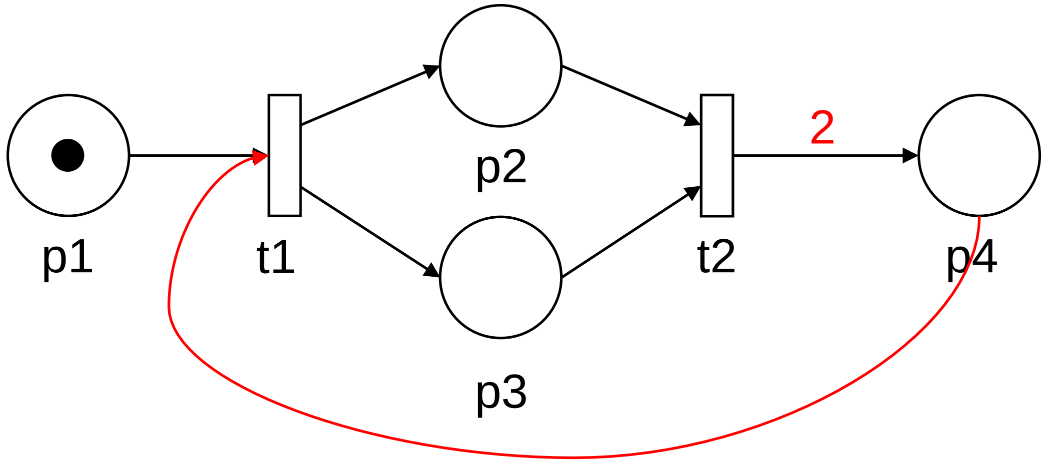
How to prove that a marking is reachable?



$$M_1 = \begin{bmatrix} 0 \\ 0 \\ 0 \\ 1 \end{bmatrix} = M_0 + A \cdot \vec{x} = \begin{bmatrix} 1 \\ 0 \\ 0 \\ 0 \end{bmatrix} + \begin{bmatrix} -1 & 0 \\ +1 & -1 \\ +1 & -1 \\ 0 & +1 \end{bmatrix} \cdot \begin{bmatrix} x_1 \\ x_2 \end{bmatrix}$$

This marking is reachable, and the state equation has a solution $\begin{bmatrix} x_1 \\ x_2 \end{bmatrix} = \begin{bmatrix} 1 \\ 1 \end{bmatrix}$

How to prove that a marking is reachable?



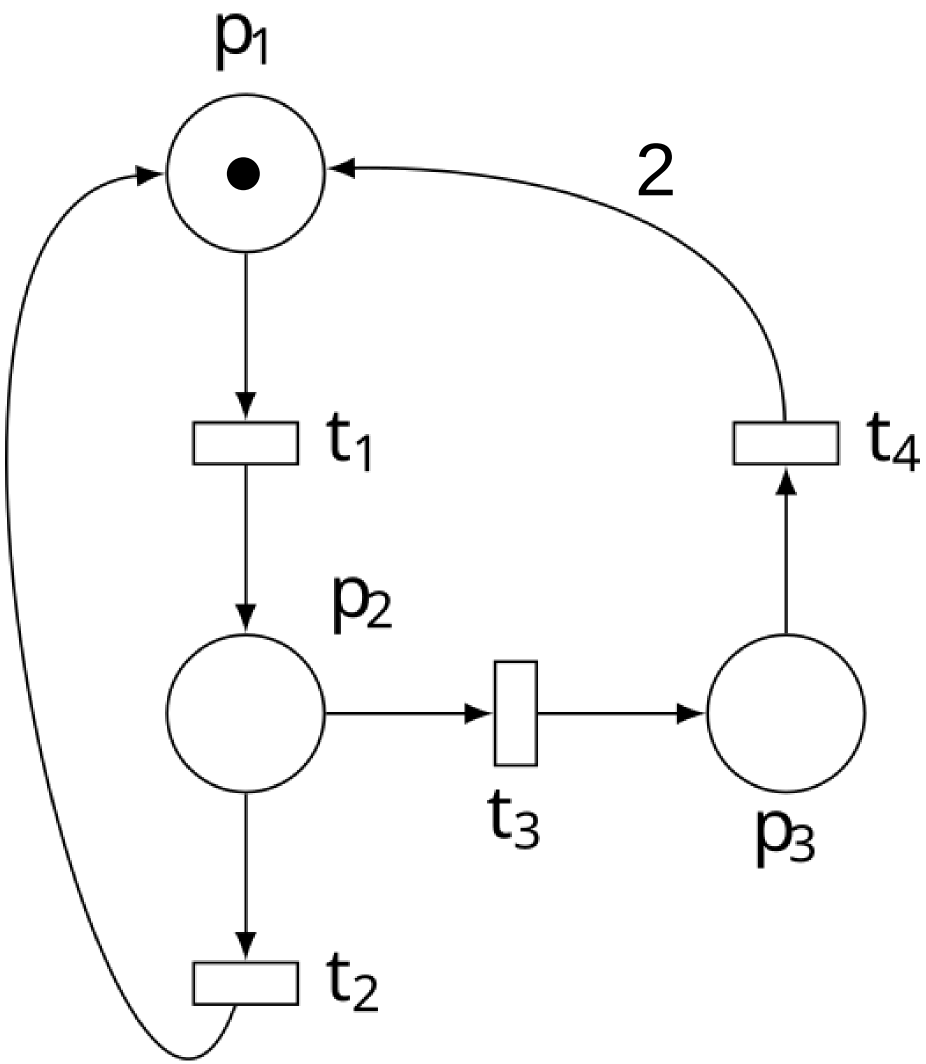
$$M_1 = \begin{bmatrix} 0 \\ 0 \\ 0 \\ 1 \end{bmatrix} = M_0 + A \cdot \vec{x} = \begin{bmatrix} 1 \\ 0 \\ 0 \\ 0 \end{bmatrix} + \begin{bmatrix} -1 & 0 \\ +1 & -1 \\ +1 & -1 \\ -1 & +2 \end{bmatrix} \cdot \begin{bmatrix} x_1 \\ x_2 \end{bmatrix}$$

This marking is **not reachable**, but the **state equation has a solution** $\begin{bmatrix} \boldsymbol{x_1} \\ \boldsymbol{x_2} \end{bmatrix} = \begin{bmatrix} \mathbf{1} \\ \mathbf{1} \end{bmatrix}$

Take away message: state equation has a solution is not a proof for reachability!

How to prove that a marking is reachable?

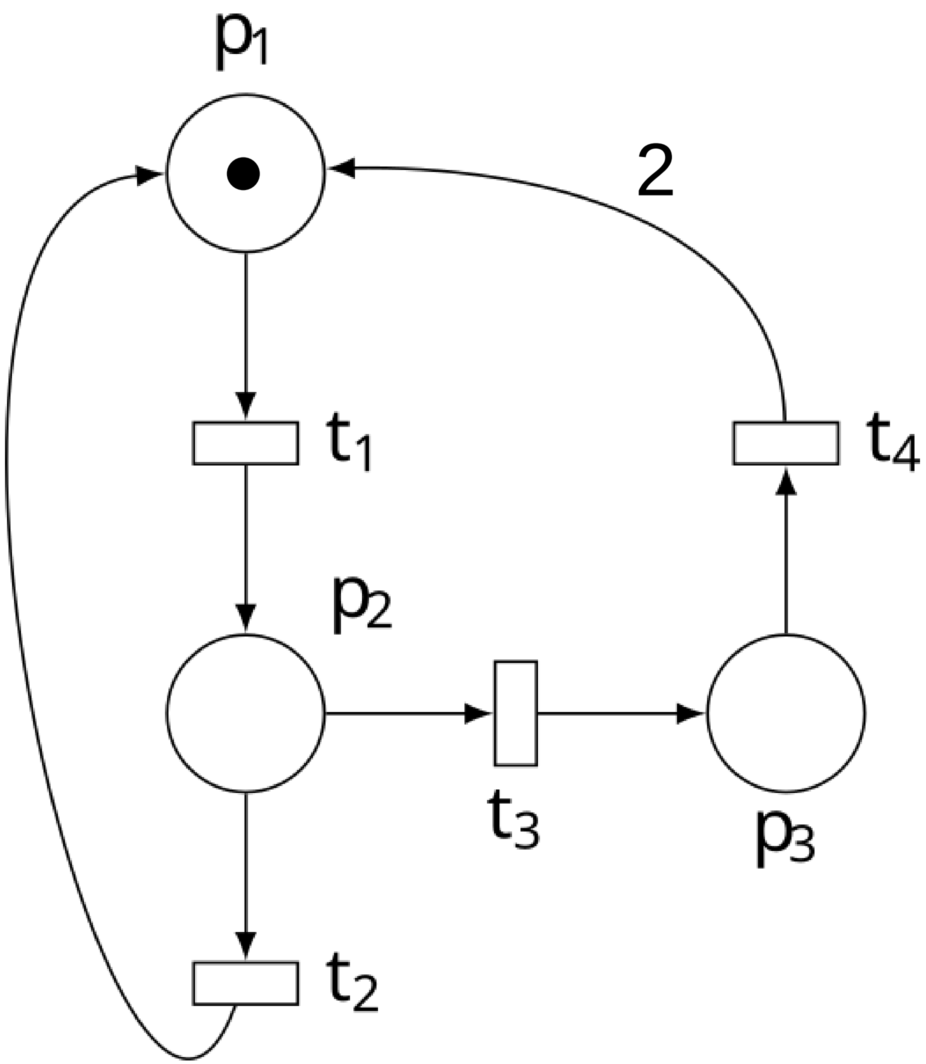
Your turn! How can you prove that the marking [101, 99, 4] is reachable or not reachable?



How to prove that a marking is reachable?

Your turn! How can you prove that the marking [101, 99, 4] is reachable or not reachable?

The state equation has a solution [100, 99, 4], but it does not prove the reachability of the marking



How to prove that a marking is reachable?

Your turn! How can you prove that the marking [101, 99, 4] is reachable or not reachable?

The state equation has a solution [306, 0, 207, 203], but it does not prove the reachability of the marking

The only way to prove reachability is a firing sequence: