Petri nets and their twofold representation to model DES

From observability to privacy and security in discrete event systems

Prof. Gianmaria DE TOMMASI
Email: detommas@unina.it

December 2020

DIETI UNIVERSITÀ DEGLI STUDI DI NAPOLI FEDERICO II
DIPARTIMENTO DI INGEGNERIA ELETTRICA E DELLE TECNOLOGIE DELL'INFORMAZIONE

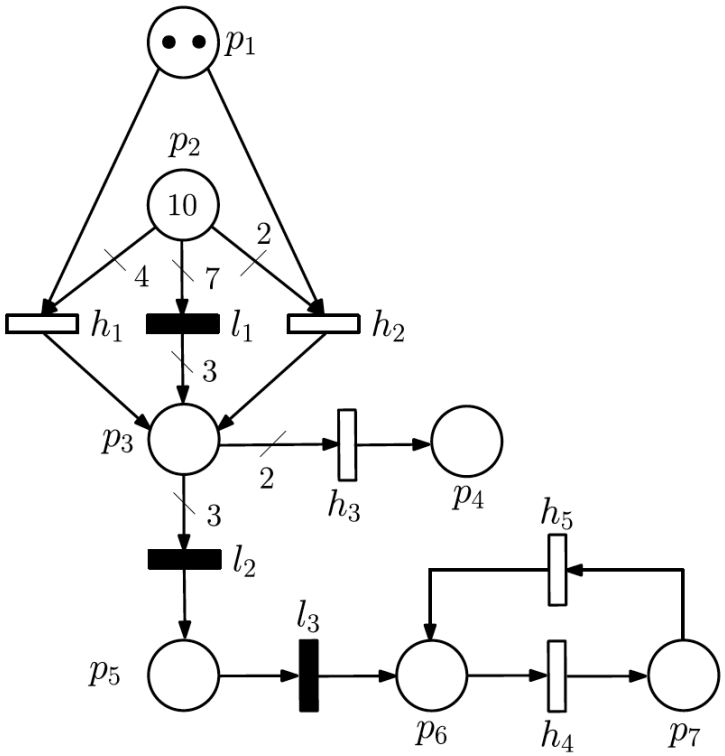
Course syllabus

1. Discrete Event Systems (DES), Languages and Automata
2. **Petri nets (PNs) and their twofold representation to model DES**
3. MILP and ILP formulations: logical conditions, binary variables “do everything", and variable connecting
4. Adding uncertainty: unobservable events and observers for finite state automata and PNs
5. Augmenting the observers: diagnosability of prefix-closed languages, diagnosers and the fault detection for finite state automata
6. Diagnosability and fault detection in PNs - Part I: graph-based approaches
7. Diagnosability and fault detection in PNs - Part II: algebraic approaches for bounded systems
8. Security issues in DES: non-interference and opacity
9. Non-interference and opacity enforcement
10. Open issues

Outline

1. Petri nets systems
   - Basic definitions
   - Language and reachability set
   - State equation
   - Reachability and coverability graphs
2. Behavioral properties
3. Structural properties
4. Labeled Petri net systems
   - Languages of a labeled net system
   - Regular language, Petri net languages and Chomsky grammars
5. Linear programming techniques for the analysis of Petri net systems
6. Software tools

Introduction



- Introduced by Carl Adam Petri in his PhD thesis in 1962 (about concurrent programming)
- Tool introduced first in Computer Science $\rightarrow$ Automatic Control $\rightarrow$ Operations Research
- In the field of Industrial Automation Petri nets *inspired* the **Grafcet** programming language for Programmable Logic Controllers (PLCs)

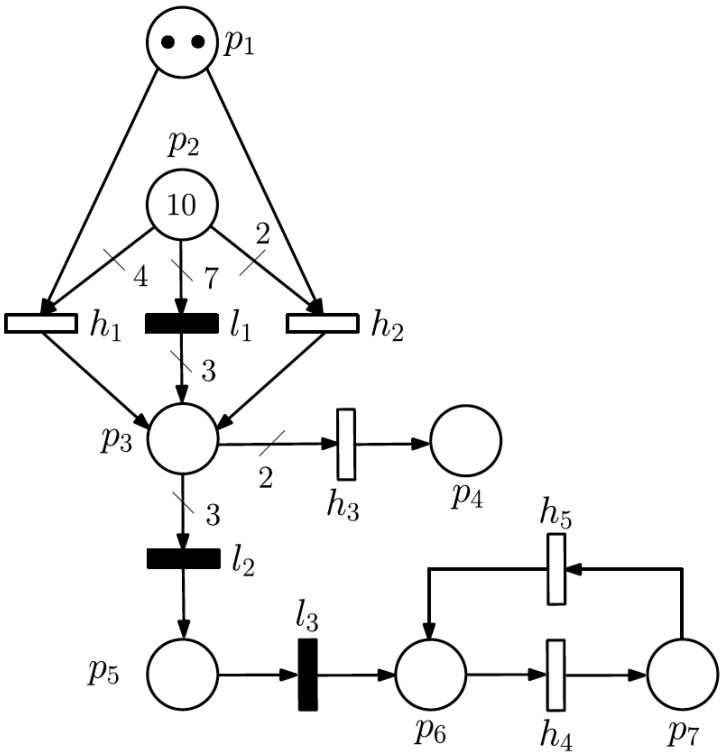
M. Silva

Half a century after Carl Adam Petri's Ph.D. thesis: A perspective on the field
Annual Reviews in Control, 2013

R. Alla

Grafcet: a powerful tool for specification of logic controllers
IEEE Transactions on Control System Technology, 1995

P/T nets



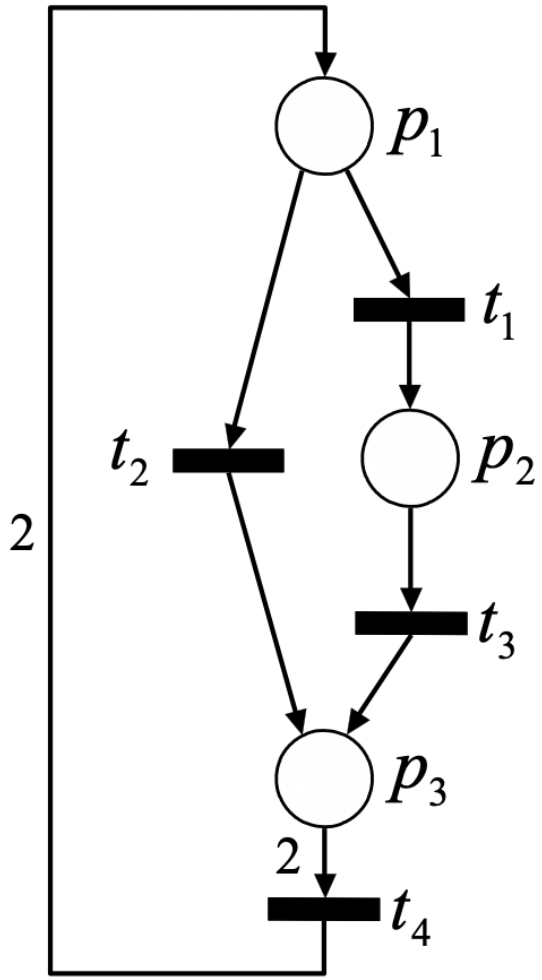
A place/transition (P/T) or Petri net is an oriented bipartite graph. Formally it can be defined as the 4-ple

$$N = (P, T, Pre, Post)$$

where

- P is a set of m *places*
- T is a set of n *transitions*
- $Pre : P \times T \mapsto \mathbb{N}$ is the *pre-incidence* function (but we will represent it as a $m \times n$ matrix)
- $Post : P \times T \mapsto \mathbb{N}$ is the *post-incidence* function

P/T net – Example



$$P = \{p_1 , p_2 , p_3\}$$

$$T = \{t_1 , t_2 , t_3 , t_4\}$$

$$\mathbf{Pre} = \begin{pmatrix} 1 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 2 \end{pmatrix}$$

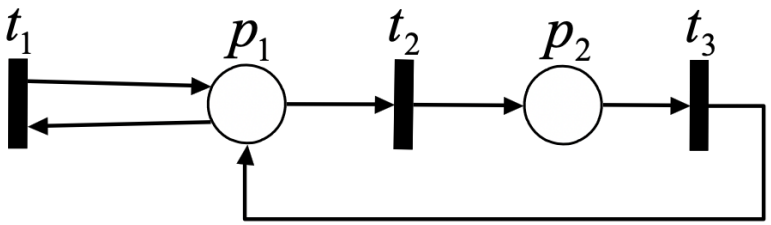
$$\mathbf{Post} = \begin{pmatrix} 0 & 0 & 0 & 2 \\ 1 & 0 & 0 & 0 \\ 0 & 1 & 1 & 0 \end{pmatrix} .$$

The incidence matrix $\boldsymbol{C}$

Given a P/T net $N = \{P, T, \mathbf{Pre}, \mathbf{Post}\}$ with m places and n transitions, the incidence matrix $\boldsymbol{C} \in \mathbb{Z}^{m \times n}$ is given by

$$\boldsymbol{C} = \mathbf{Post} - \mathbf{Pre}$$

In general, the incidence matrix does not contain sufficient information to reconstruct the net structure ($\rightarrow$ this will have an impact on the use of the so-called *state equation*)



Pre and post sets

Given a transition $t \in T$ we denote with

- ${}^\bullet t = \{p \in P \mid \mathbf{Pre}(p, t) > 0\}$, the *preset* of t (*input places*)
- $t^\bullet = \{p \in P \mid \mathbf{Post}(p, t) > 0\}$, the *postset* of t (*output places*)

Given a place $p \in P$ we denote with

- ${}^\bullet p = \{t \in T \mid \mathbf{Post}(p, t) > 0\}$, the *preset* of p (*input transitions*)
- $p^\bullet = \{t \in T \mid \mathbf{Pre}(p, t) > 0\}$, the *postset* of p (*output transitions*)

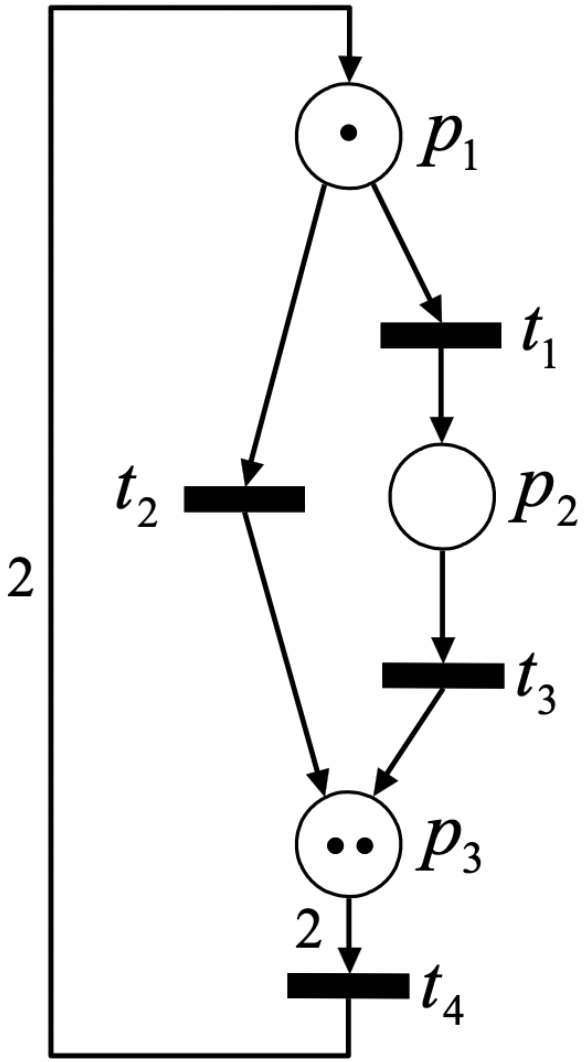
From a graph to a model for DES
The net system $\mathcal{S}$

- P/T nets are *just* graphs
- In order to use them to model dynamic systems, we need to introduce the concept of *state*
- The **marking** is the way we have to define a discrete state space
- The marking can be defined as a function (other definition can be found in literature, like marking as a multiset)

$$\boldsymbol{m} : P \mapsto \mathbb{N},$$

  that assigns to each place a nonnegative integer number of **tokens**
- The marking is usually represented as a vector $\boldsymbol{m} \in \mathbb{N}^m$
- A P/T net N with its initial marking $\boldsymbol{m}_0$ is called **net system**, and is denoted with $\langle N, \boldsymbol{m}_0 \rangle$

Marking – example



$$\boldsymbol{m}_0 = (1\ 0\ 2)^T$$

How the state evolves in $\mathcal{S}$?

Enabling condition

A transition t is enabled at the marking $\boldsymbol{m}$ if

$$\boldsymbol{m} \geq \mathbf{Pre}(\cdot\,, t)$$

i.e., if each place $p \in P$ contains a number of tokens greater than or equal to $\mathbf{Pre}(p\,, t)$

- $\boldsymbol{m}\left[t\right\rangle$ denotes that t is enabled at $\boldsymbol{m}$
- $\boldsymbol{m}\neg\left[t\right\rangle$ dentoes that t is not enabled at $\boldsymbol{m}$

How the state evolves in $\mathcal{S}$?

Firing of a transition

- A transition t enabled at $\boldsymbol{m}$ *can* fire
- The firing of t removes $\mathbf{Pre}(p, t)$ tokens from each place $p \in P$ and adds $\mathbf{Post}(p, t)$ tokens in each place $p \in P$
- Hence, the firing of t yields the new marking

$$\boldsymbol{m}' = \boldsymbol{m} - \mathbf{Pre}(\cdot, t) + \mathbf{Post}(\cdot, t) = \boldsymbol{m} + \boldsymbol{C}(\cdot, t)$$

- $\boldsymbol{m}\left[t\right\rangle\boldsymbol{m}'$ denotes that the firing of t from $\boldsymbol{m}$ leads to $\boldsymbol{m}'$

Firing of a sequence

A firing sequence $\sigma = t^1 t^2 \dots t^r \in T^*$ is enabled at $\boldsymbol{m}$ if and only if

- $\boldsymbol{m}\left[t^1\right\rangle$ and $\boldsymbol{m}\left[t^1\right\rangle \boldsymbol{m}_1$
- $\boldsymbol{m}_1\left[t^2\right\rangle$ and $\boldsymbol{m}_1\left[t^2\right\rangle \boldsymbol{m}_2$
- $\dots$
- $\boldsymbol{m}_{r-1}\left[t^r\right\rangle$ and $\boldsymbol{m}_{r-1}\left[t^r\right\rangle \boldsymbol{m}_r$

$\boldsymbol{m}\left[\sigma\right\rangle$ denotes that σ is enabled at $\boldsymbol{m}$, while $\boldsymbol{m}\left[\sigma\right\rangle \boldsymbol{m}'$ denotes that the firing sequence σ strarting from $\boldsymbol{m}$ yields the marking $\boldsymbol{m}'$

The empty sequence

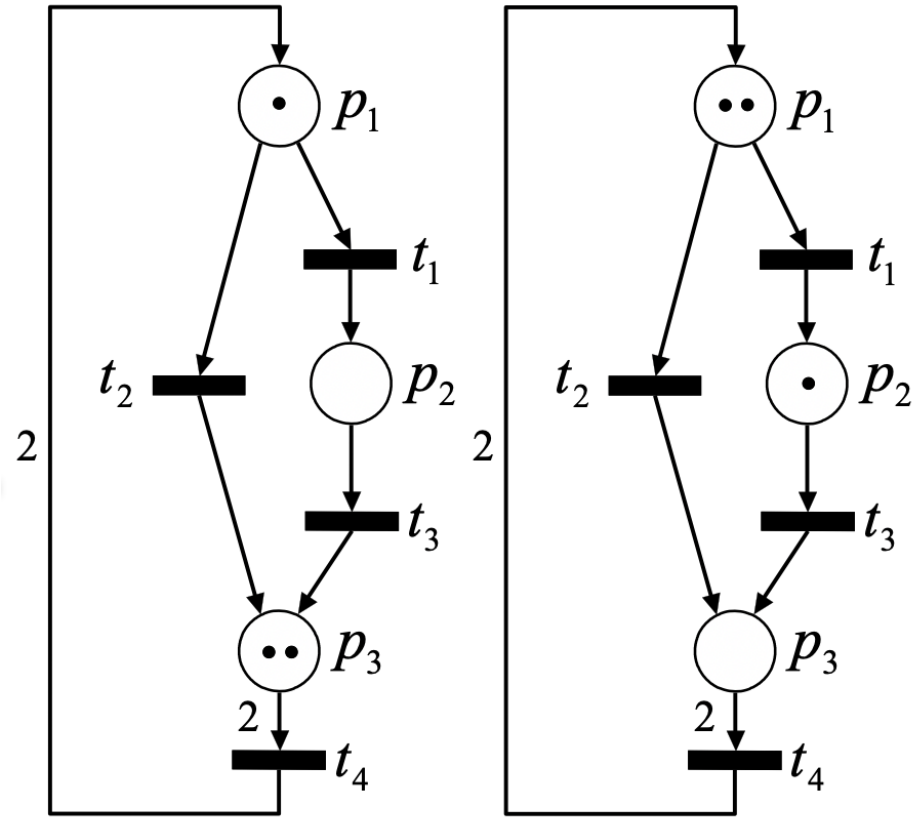
ε is the empty sequence and it is

- $\boldsymbol{m}\left[\varepsilon\right\rangle \quad \forall\, \boldsymbol{m} \in \mathbb{N}^m$
- $\boldsymbol{m}\left[\varepsilon\right\rangle \boldsymbol{m}$

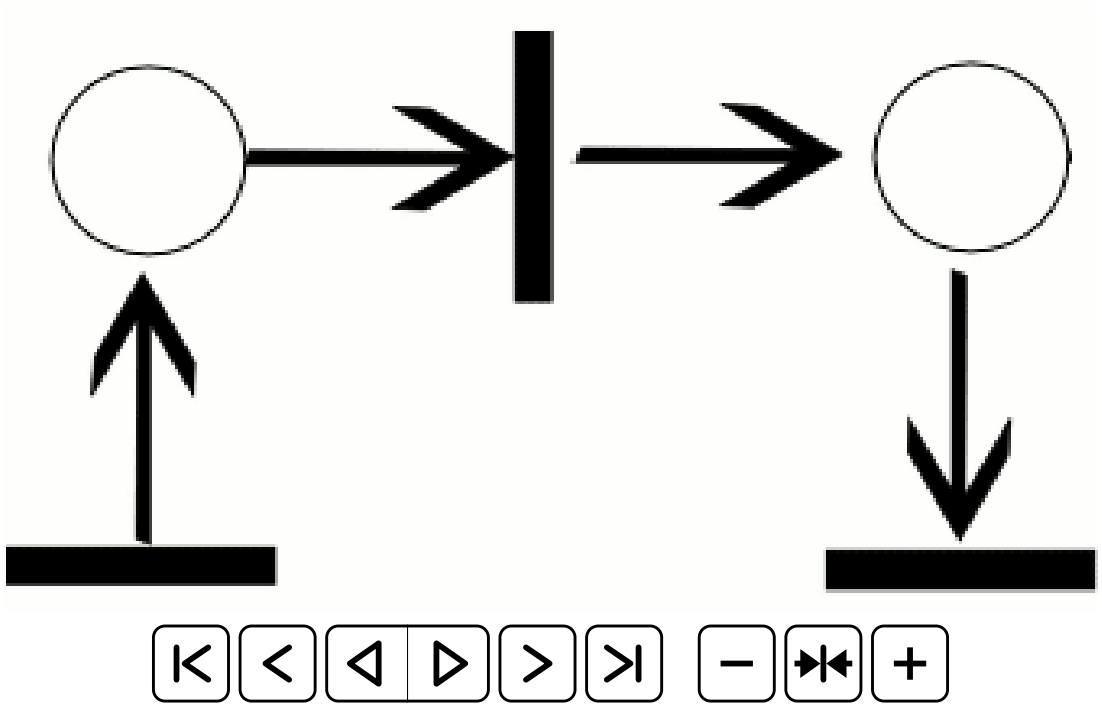
Firing sequence – Example

- The firing sequence $\sigma = t_4 t_1$ is enable at $\boldsymbol{m} = (1\ 0\ 2)^T$
- Its firing yields

$$\boldsymbol{m}' = (2\ 1\ 0)^T$$



Firing sequence – Example



Firing count **vector**

Given a net system $\mathcal{S}$ and a firing sequence $\sigma \in T^*$, it is possible to introduce the **firing count vector** $\boldsymbol{\sigma} \in \mathbb{N}^n$ whose entry $\boldsymbol{\sigma}(t_i) = \boldsymbol{\sigma}_i$ denotes how many times transition t_i appears in the sequence σ

Example

- $T = \{t_1 , t_2 , t_3\}$
- $\sigma = t_3 t_1 t_1 t_3 t_2$
- $\boldsymbol{\sigma} = (2\ 1\ 2)^T$

Language of a net system $\mathcal{S}$

- So far we have dealt with the so called unlabeled Petri nets
- The set of transitions somehow corresponds to the set of events that drive the system dynamic
- It is then possible to associate a language to $\mathcal{S} = \langle N, \boldsymbol{m}_0 \rangle$ defined as follows

$$L(N, \boldsymbol{m}_0) = \left\{ \sigma \in T^* \mid \boldsymbol{m}_0 \left[\sigma \right\rangle \right\}$$

- It readily follows that $\varepsilon \in L(N, \boldsymbol{m}_0)$

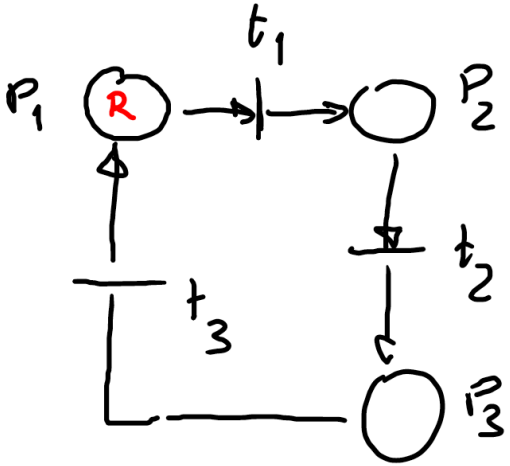
Reachability set $R(N, \boldsymbol{m}_0)$

- Given a net system $\mathcal{S} = \langle N, \boldsymbol{m}_0 \rangle$ a marking $\boldsymbol{m}$ is said to be reachable (from $\boldsymbol{m}_0$) if there exists a firing sequence $\sigma \in L(N, \boldsymbol{m}_0)$ such that $\boldsymbol{m}_0 \left[\sigma\right\rangle \boldsymbol{m}$
- It is then possible to introduce the **reachability set** $R(N, \boldsymbol{m}_0)$ of $\mathcal{S}$ as follows

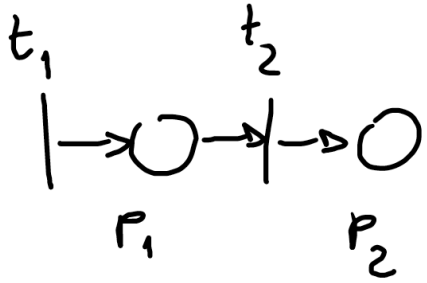
$$R(N, \boldsymbol{m}_0) = \left\{ \boldsymbol{m} \in \mathbb{N}^m \mid \exists\, \sigma \in L(N, \boldsymbol{m}_0) \text{ s.t. } \boldsymbol{m}_0 \left[\sigma\right\rangle \boldsymbol{m} \right\}$$

- Obviously it is $\boldsymbol{m}_0 \in R(N, \boldsymbol{m}_0)$

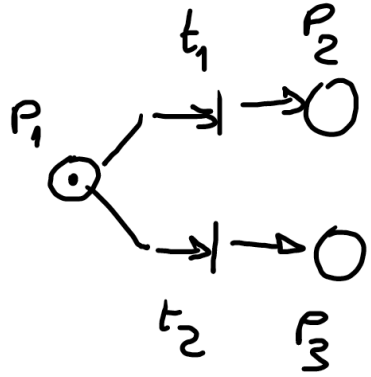
Reachability set – Examples



$m_0 = (R \;\; 0 \;\; 0)^T$



$m_0 = (0 \;\; 0)^T$



$m_0 = (1 \;\; 0 \;\; 0)$

WHAT IS THE CARDINALITY OF $R(N, m_0)$?

State equation

Given a net system $\mathcal{S} = \langle N, \boldsymbol{m}_0 \rangle$ and a reachable marking $\boldsymbol{m} \in R(N, \boldsymbol{m}_0)$ such that $\boldsymbol{m}_0 \left[\sigma\right\rangle \boldsymbol{m}$ with $\sigma \in L(N, \boldsymbol{m}_0)$.

If $\boldsymbol{\sigma}$ is the firing count vector associated to firing sequence σ then the following **state equation** holds

$$\boldsymbol{m} = \boldsymbol{m}_0 + \boldsymbol{C} \cdot \boldsymbol{\sigma}$$

Reachability graph

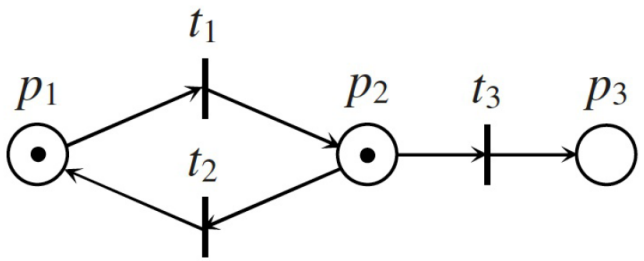
If a net system $\mathcal{S}$ has a reachability set $R(N, \boldsymbol{m}_0)$ with finite cardinality, then it is possible to explicitly represent the whole state space by means of an automaton, called **reachability graph**

Algorithm 10.2. (Reachability Graph).

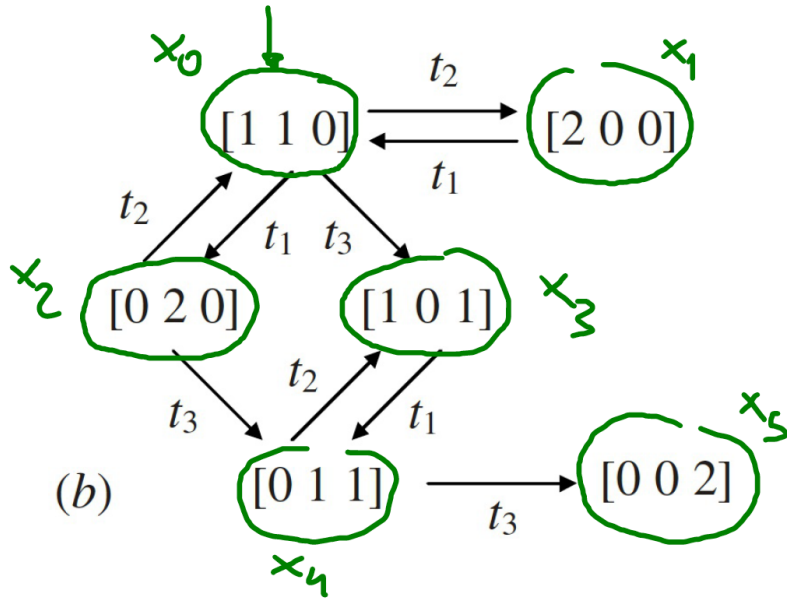
1. *The initial node of the graph is the initial marking* $\boldsymbol{m}_0$. *This node is initially unlabeled.*
2. *Consider an unlabeled node* $\boldsymbol{m}$ *of the graph.*
   - a *For each transition* t *enabled at* $\boldsymbol{m}$, *i.e., such that* $\boldsymbol{m} \geq \boldsymbol{Pre}[\cdot, t]$:
     - i. *Compute the marking* $\boldsymbol{m}' = \boldsymbol{m} + \boldsymbol{C}[\cdot, t]$ *reached from* $\boldsymbol{m}$ *firing* t.
     - ii. *If no node* $\boldsymbol{m}'$ *is on the graph, add a new node* $\boldsymbol{m}'$ *to the graph.*
     - iii. *Add an arc* t *from* $\boldsymbol{m}$ *to node* $\boldsymbol{m}'$.
   - b *Label node* $\boldsymbol{m}$ *"old".*
3. *If there exist nodes with no label, goto Step 2.*

Figure: Algorithm taken from Cabasino *et al.*, "Introduction to Petri nets"

Reachability graph – Example



(a)



(b)

THE REACHABILITY GRAPH AS AN AUTOMATA

$E = \{t_1, t_2, t_3\}$

Coverability graph - 1/2

- If the cardinality of $R(N, \boldsymbol{m}_0)$ is infinite, then the reachability graph cannot be constructed
- In order to represent marking whose components can arbitrarily grow, the symbol ω is adopted to denote an arbitrarily large component in a marking vector

Coverability tree

1 The root node of the tree is the initial marking $\boldsymbol{m}_0$. This node is initially unlabeled.

2 Consider an unlabeled node $\boldsymbol{m}$ of the tree.

a For each transition t enabled at $\boldsymbol{m}$, i.e., such that $\boldsymbol{m} \geq \boldsymbol{Pre}[\cdot, t]$:

i. Compute the marking $\boldsymbol{m}' = \boldsymbol{m} + \boldsymbol{C}[\cdot, t]$ reached from $\boldsymbol{m}$ firing t.

ii. For all markings $\tilde{\boldsymbol{m}} \lneq \boldsymbol{m}'$ on the path from the root node $\boldsymbol{m}_0$ to node $\boldsymbol{m}$ and for all $p \in P$,
if $\tilde{m}[p] < m'[p]$ then let $m'[p] = \omega$.

iii. Add a new node $\boldsymbol{m}'$ to the tree.

iv. Add an arc t from $\boldsymbol{m}$ to the new node $\boldsymbol{m}'$.

v. If there already exists a node $\boldsymbol{m}'$ in the tree, label the new node $\boldsymbol{m}'$ "duplicated".

b Label node $\boldsymbol{m}$ "old".

3 If there exist nodes with no label, goto Step 2.

Coverability graph - 2/2

Algorithm 10.4. (Coverability graph).

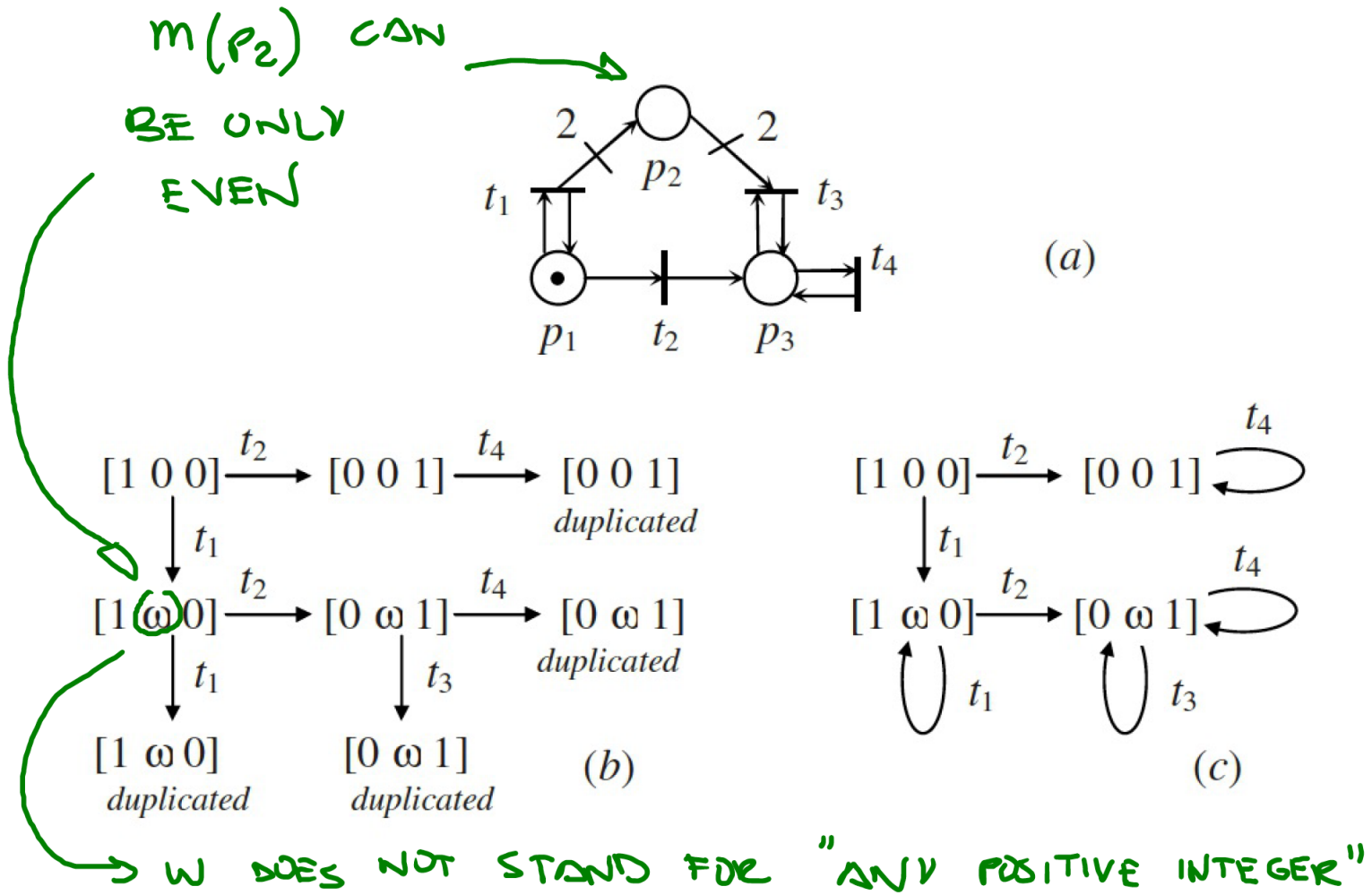
1 If the tree contains no nodes with label "duplicated" goto Step 4.
2 Consider a node $\mathbf{m}$ *of the graph with label "duplicated".*
Such a node has no output arcs but an input arc t from node $\mathbf{m}'$.
Moreover, there surely exists in the graph another node $\mathbf{m}$ *with label "old".*

a Remove arc t from node $\mathbf{m}'$ *to node* $\mathbf{m}$ *"duplicated".*
b Add an arc t from node $\mathbf{m}'$ *to node* $\mathbf{m}$ *"old".*
c Remove node $\mathbf{m}$ *"duplicated".*

3 If there still exist nodes with label "duplicated" goto Step 2.
4 Remove labels from nodes.

Figure: Algorithm taken from Cabasino *et al.*, "Introduction to Petri nets"

Coverability graph – Example



Behavioral properties of Petri net systems

The behavioral properties of Petri net systems are those ones that depends (also) on the initial marking $\boldsymbol{m}_0$

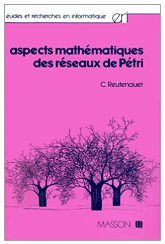
- **reachability**
- **boundedness**
- **conservativeness**
- **repetitiveness**
- **reversibility**
- **liveness**

The reachability problem

Given a net systems $\mathcal{S} = \langle N, \boldsymbol{m}_0 \rangle$ and a generic marking $\boldsymbol{m} \in \mathbb{N}^m$, is $\boldsymbol{m} \in R(N, \boldsymbol{m}_0)$?

- it is straightforward to see that if $R(N, \boldsymbol{m}_0)$ has finite cardinality, i.e. the net system is bounded, then the problem is **decidable**. **In this case the reachability problem is equivalent to the decidability of regular languages**
- it is also straightforward to show that in the general case the problem is at least **semi-decidible**
- In the 1980s it has been proved the the reachability problem is decidible, but the corresponding algorithm has a very high complexity

C. Reutenauer
Aspects Mathématiques des Réseaux de Petri
Prentice Hall, 1989



Boundedness

k-boundedness

- A place p is k-bounded in the system $\mathcal{S} = \langle N, \boldsymbol{m}_0 \rangle$ if for all the reachable markings $\boldsymbol{m} \in R(N, \boldsymbol{m}_0)$ it holds $\boldsymbol{m}(p) \leq k$
- A system $\mathcal{S}$ is k-bounded if all its places are k-bounded

- When we are not interested in any specific k, then the system $\mathcal{S}$ is simply called **bounded**
- Obviously the following holds

$$\mathcal{S} \text{ is bounded} \Leftrightarrow R(N, \boldsymbol{m}_0) \text{ is finite}$$

Reversibility

Reversibility

A net system $\mathcal{S}$ is reversibile if for all $\boldsymbol{m} \in R(N, \boldsymbol{m}_0)$ it holds that $\boldsymbol{m}_0 \in R(N, \boldsymbol{m})$

- Reversibility implies that a system $\mathcal{S}$ can always be reinitialized to its initial marking $\boldsymbol{m}_0$
- Non reversible systems may exhibit **home states**

Home state

A marking $\widetilde{\boldsymbol{m}} \in R(N, \boldsymbol{m}_0)$ is a **home state** if for all $\boldsymbol{m} \in R(N, \boldsymbol{m}_0)$ it holds that $\widetilde{\boldsymbol{m}} \in R(N, \boldsymbol{m})$

Liveness of a transition

Given a net system $\mathcal{S}$ a transition t is said to be

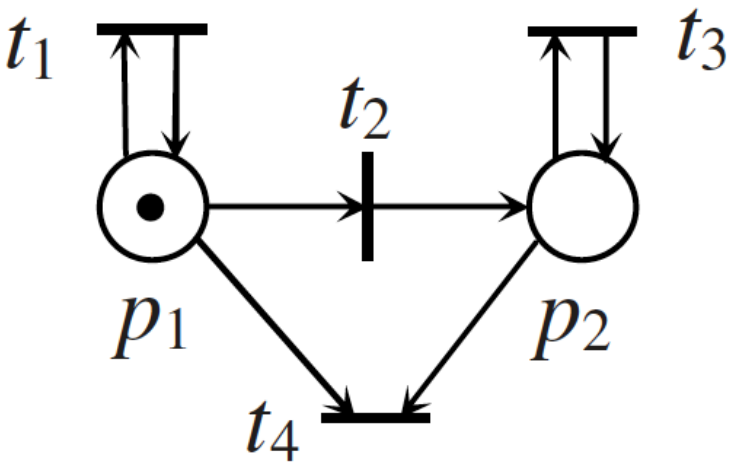
- dead if no reachable marking enables t
- quasi-live if t is enabled by some reachable marking, i.e. $\exists\, \boldsymbol{m} \in R(N, \boldsymbol{m}_0)$ s.t. $\boldsymbol{m}[t\rangle$
- live if for all reachable markings $\boldsymbol{m} \in R(N, \boldsymbol{m}_0)$, t is *quasi-live* in $\langle N, \boldsymbol{m}\rangle$

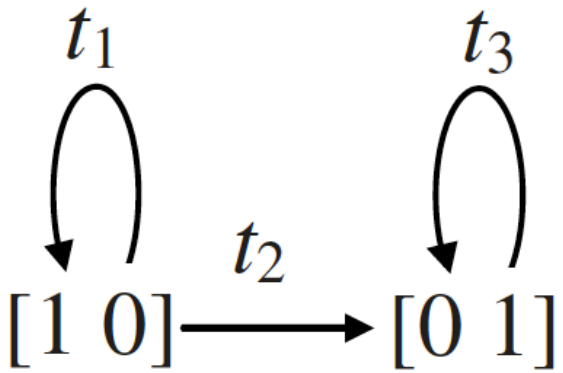
Liveness of a net system $\mathcal{S}$

- $\mathcal{S}$ is **dead** if all its transitions are dead
- $\mathcal{S}$ is **not quasi-live** if some of its transitions are dead and some quasi-live
- $\mathcal{S}$ is **quasi-live** if all its transitions are quasi-live
- $\mathcal{S}$ is **live** if all its transitions are live

Sometimes a net system is considered live if it has some live transitions (not necessarily all)

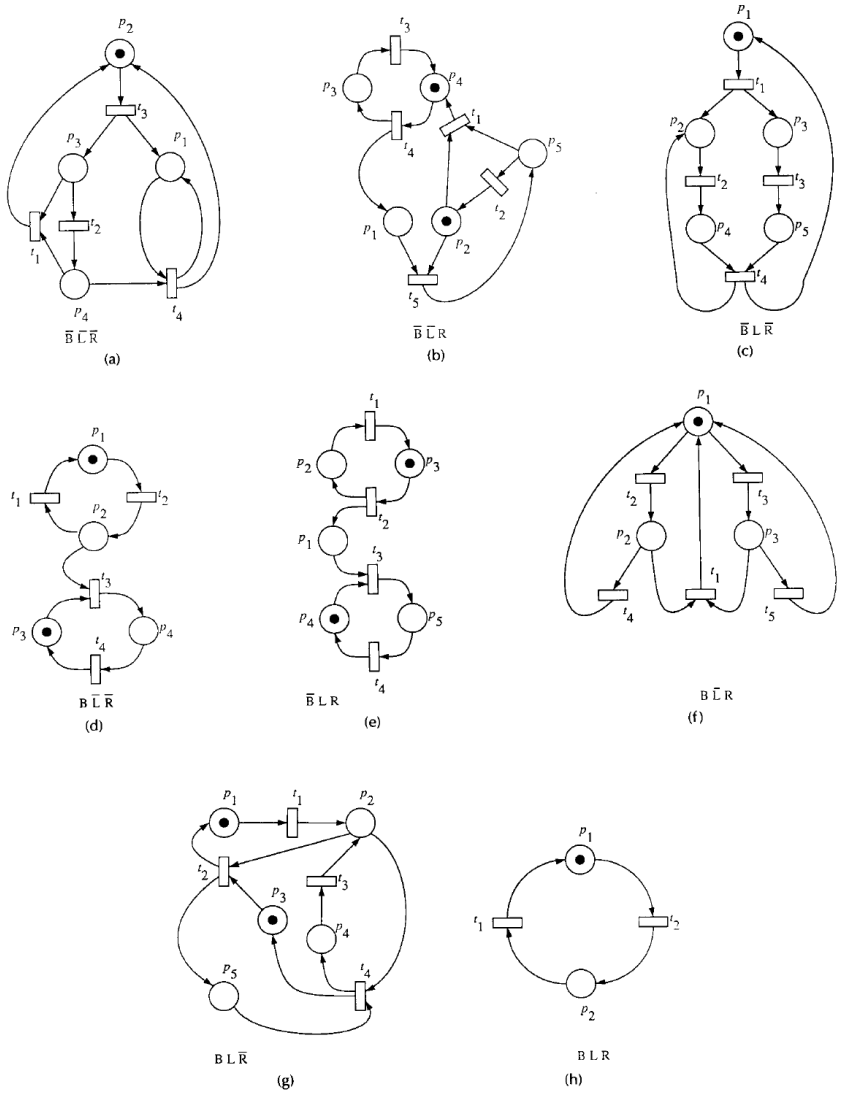
Liveness – Example





Boundedness, liveness and reversibility

Boundedness, liveness and reversibility are independent concepts (image taken from Murata, *Proc. IEEE*, 1989)



Use the graphs to check behavioral properties

- Both the reachability (for bounded systems) and coverability graph (for unbounded systems) can be used to assess the behavioral properties
- Reachability graph permits to formulate necessary and sufficient conditions; there is a link with decidability of regular languages
- Coverability graph permits formulate either necessary or sufficient conditions

Structural properties of a Petri net

The structural properties of Petri net systems are those ones that depends only on the graph and not on the specific initial marking $\boldsymbol{m}_0$

- **analysis based on the state equation**
- **analysis based on invariants**
- **analysis based on siphons and traps**

Use state equation for reachability

- the **satisfaction of the state equation is only a necessary condition for reachability**
  - Given a marking $\boldsymbol{m} \in \mathbb{N}^m$, if $\boldsymbol{m} \in R(N, \boldsymbol{m}_0) \Rightarrow \exists \boldsymbol{\sigma} \in \mathbb{N}^n$ such that $\boldsymbol{m} = \boldsymbol{m}_0 + \boldsymbol{C} \cdot \boldsymbol{\sigma}$
- Potentially reachable marking $PR(N, \boldsymbol{m}_0)$

$$PR(N, \boldsymbol{m}_0) = \{\boldsymbol{m} \in \mathbb{N}^m \mid \exists\, \boldsymbol{y} \in \mathbb{N}^n \text{ s.t. } \boldsymbol{m} = \boldsymbol{m}_0 + \boldsymbol{C} \cdot \boldsymbol{y}\}$$

- **It is** $R(N, \boldsymbol{m}_0) \subseteq PR(N, \boldsymbol{m}_0)$
- The markings in $PR(N, \boldsymbol{m}_0) \setminus R(N, \boldsymbol{m}_0)$ are called spurious markings, since they satisfy the state equation but are not reachable

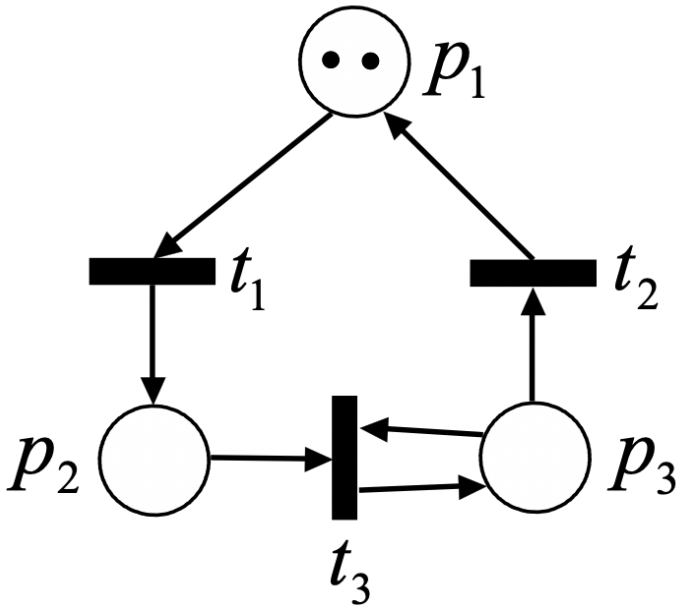
Spurious markings – Examples

- $\boldsymbol{m}_0 = (2\ 0\ 0)^T$
- If $\boldsymbol{y} = (1\ 0\ 1)^T$ then

$$\boldsymbol{m} = (1\ 0\ 0)^T \in PR(N, \boldsymbol{m}_0)$$

while

$$\boldsymbol{m} = (1\ 0\ 0)^T \notin R(N, \boldsymbol{m}_0)$$



When $PR(N, \boldsymbol{m}_0) = R(N, \boldsymbol{m}_0)$?

- There are cases where the satisfaction of the state equation becomes necessary **and sufficient** for reachability, i.e. $PR(N, \boldsymbol{m}_0) = R(N, \boldsymbol{m}_0)$
  - Acyclic nets
  - Some classes of ordinary nets (nets with all arcs that have unitary multiplicity)
    - State machines (only choices and convergences)
    - Marked graphs (only parallelisms and synchronizations)
- When $PR(N, \boldsymbol{m}_{,0}) = R(N, \boldsymbol{m}_0)$, the following feasibility problem with integer unknowns and linear constraints can be solved to assess reachability

  $\boldsymbol{m} \in \mathbb{N}^m$ is reachable **if and only if** it is possible to find a $\boldsymbol{\sigma} \in \mathbb{N}^n$ such that

  $$\boldsymbol{C} \cdot \boldsymbol{\sigma} = \boldsymbol{m} - \boldsymbol{m}_0$$

  which is NP-hard

Invariants

P-invariants

Given a net N a vector $\boldsymbol{x} \in \mathbb{N}^m$ with $\boldsymbol{x} \neq \mathbf{0}$ is called a **P-invariant** if

$$\boldsymbol{x}^T \cdot \boldsymbol{C} = \mathbf{0}^\mathbf{T}$$

T-invariants

Given a net N a vector $\boldsymbol{y} \in \mathbb{N}^n$ with $\boldsymbol{y} \neq \mathbf{0}$ is called a **T-invariant** if

$$\boldsymbol{C} \cdot \boldsymbol{y} = \mathbf{0}$$

P-invariants

- If a marking $\boldsymbol{m}$ is reachable then it satisfies the state equation

$$\boldsymbol{m} = \boldsymbol{m}_0 + \boldsymbol{C} \cdot \sigma$$

- by multiplying by a P-vector $\boldsymbol{x}$

$$\boldsymbol{x}^T \cdot \boldsymbol{m} = \boldsymbol{x}^T \cdot \boldsymbol{m}_0 + \boldsymbol{x}^T \cdot \boldsymbol{C} \cdot \sigma$$

hence, if $\boldsymbol{m} \in R(N, \boldsymbol{m}_0)$ then

$$\boldsymbol{x}^T \cdot \boldsymbol{m} = \boldsymbol{x}^T \cdot \boldsymbol{m}_0$$

T-invariants

- Let σ be a firing sequence such that $\boldsymbol{m}_0 \left[\sigma\right\rangle \boldsymbol{m}$. If the corresponding firing count vector $\boldsymbol{\sigma}$ is a T-invariant, then it is $\boldsymbol{m} = \boldsymbol{m}_0$
- All sequences σ whose corresponding firing count vector $\boldsymbol{\sigma}$ is a T-invariant are repetitive and stationary sequences

Siphons and traps

Siphons and traps are usually introduced for *ordinary nets*, i.e. **nets with all arcs with unitary multiplicity**

Siphons

A siphon of an ordinary net N is a set of places $\mathbb{S} \subseteq P$ such that the set of input transitions of $\mathbb{S}$ is included in the set of output transitions of $\mathbb{S}$, that is

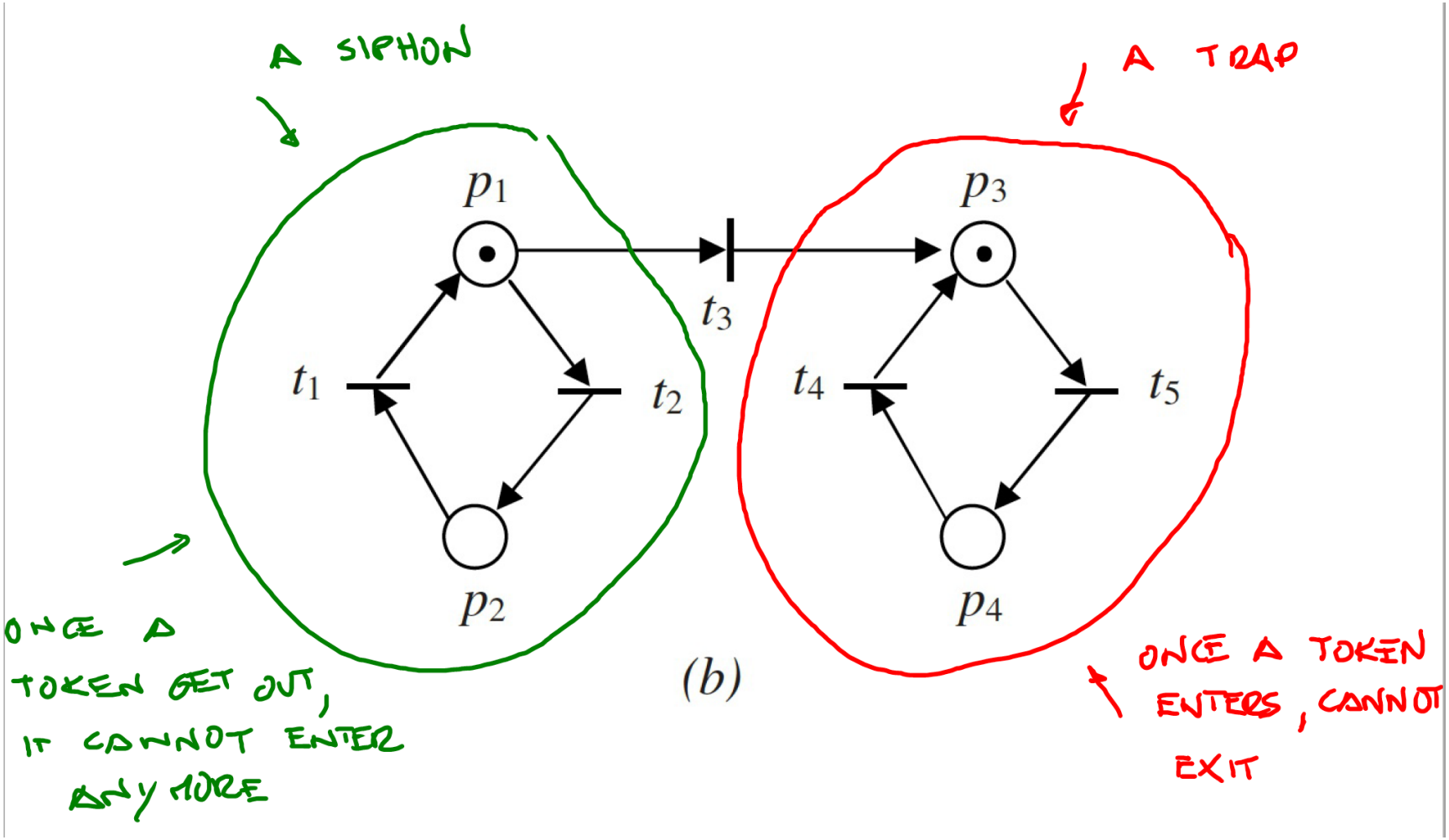
$$\bigcup_{p \in \mathbb{S}} {}^\bullet p \subseteq \bigcup_{p \in \mathbb{S}} p^\bullet$$

Traps

A trap of an ordinary net N is a set of places $\mathbb{T} \subseteq P$ such that the set of output transitions of $\mathbb{T}$ is included in the set of input transitions of $\mathbb{T}$, that is

$$\bigcup_{p \in \mathbb{T}} p^\bullet \subseteq \bigcup_{p \in \mathbb{T}} {}^\bullet p$$

Siphon and traps – Example



Use P-invariants for reachability

The X-invariant set $I_X(N, \boldsymbol{m}_0)$

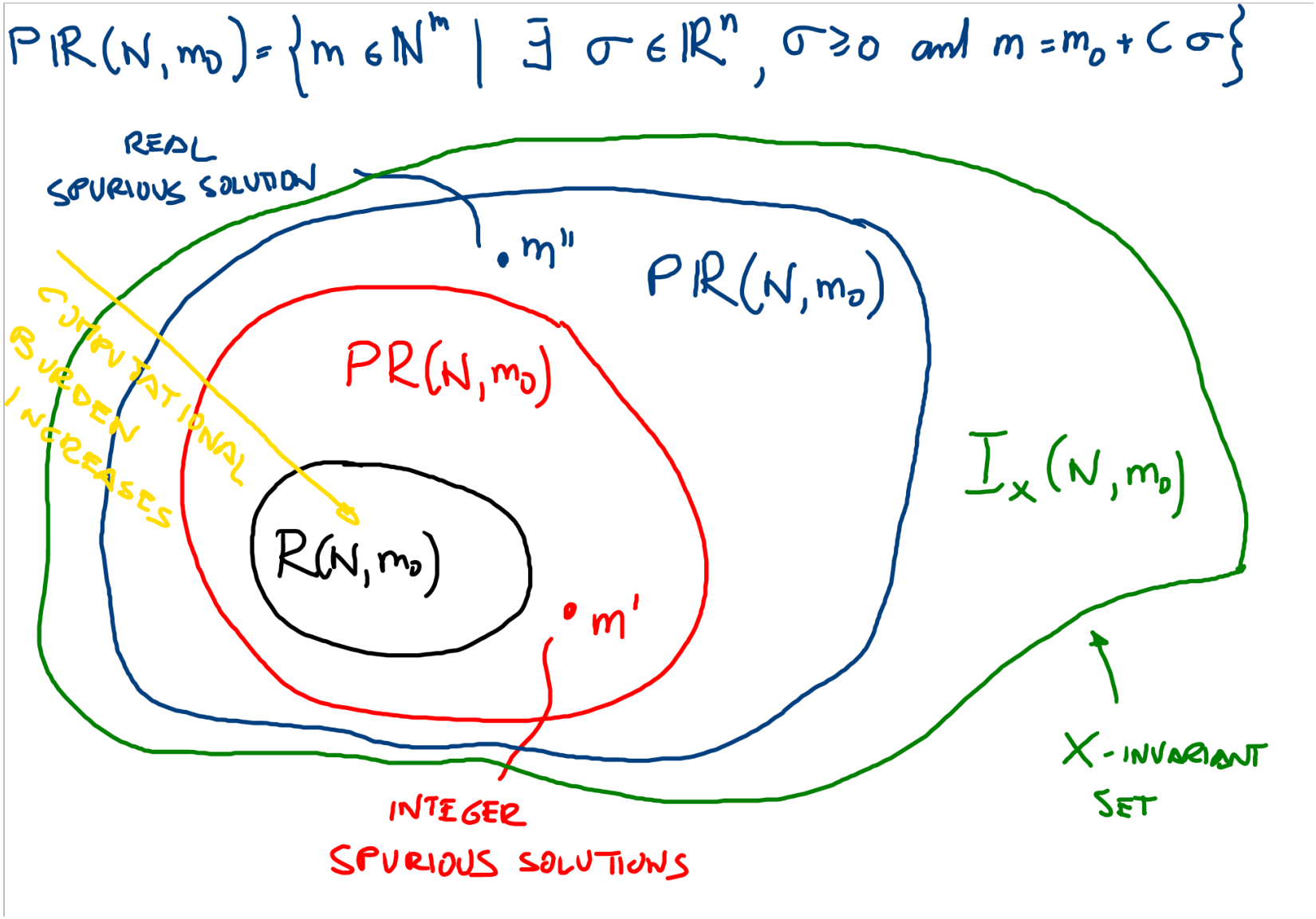
Let $\mathcal{S} = \langle N, \boldsymbol{m}_0 \rangle$ be a net system with m places and consider the matrix $\boldsymbol{X} = \{\boldsymbol{x}_1, \boldsymbol{x}_2, \ldots, \boldsymbol{x}_k\} \in \mathbb{N}^{m \times k}$, whose generic column $\boldsymbol{x}_i$ is a P-invariant of N. The X-invariant set of $\mathcal{S}$ is

$$I_X(N, \boldsymbol{m}_0) = \left\{ \boldsymbol{m} \in \mathbb{N}^m \mid \boldsymbol{X}^T \cdot \boldsymbol{m} = \boldsymbol{X}^T \cdot \boldsymbol{m}_0 \right\}$$

- It can be easily proved that

$$R(N, \boldsymbol{m}_0) \subseteq PR(N, \boldsymbol{m}_0) \subseteq I_X(N, \boldsymbol{m}_0)$$

Estimates of the reachability set ($\mathbf{I}, \boldsymbol{m}_0$)



Labeled Petri net systems

Given an **alphabet (of events)** E, a **labeled Petri net system** is the 4-ple

$$\mathcal{S}_\ell = (N, \ell, \boldsymbol{m}_0, F)$$

where

- $\ell : T \mapsto E \cup \{\varepsilon\}$ is the labeling function that associates an event $e \in E \cup \{\varepsilon\}$ to every transition $t \in T$
- F is the set of **final** markings

NOTE: an event can be associated to more than one transition $\rightarrow$ source of nondeterminism

Types of labeling functions

Three different types of labeling functions can be considered

- **free labeling** – all transitions are distinctly labeled and none is labeled as the silent event ε; the labeling function does not add any relevant information, i.e. we can consider $T = E$
- **ε–free labeling** (usually referred to also as λ–free)– no transition is labeled with the silent event ε (source of nondeterminism)
- **arbitrarily labeling** – no restriction posed on the labeling function (further source of nondeterminism)

From sequences to words

The labeling function $\ell(\cdot)$ can be recursively extended to firing sequences σ so to associate them words w

- $\ell : T^* \mapsto E^*$ with
  - $\ell(\varepsilon) = \varepsilon$
  - $\ell(\sigma t) = \ell(\sigma)\ell(t)$
- The adopted notation will be
  - $e = \ell(t)$, with $t \in T$ and $e \in E$
  - $w = \ell(\sigma)$, with $\sigma \in T^*$ and $w \in E^*$

Languages of a labeled net system

Similarly to what has been done in the case of automata, also in the case of labeled Petri net system, the following two different languages can be introduced

- The generated language (sometimes referred to as the *prefix language*)

$$\mathcal{L}(\mathcal{S}_\ell) = \left\{ w = \ell(\sigma) \in E^* \text{ s.t. } \boldsymbol{m}_0 \left[\sigma\right\rangle \right\}$$

- The marked language (sometimes referred to as the *terminal language*)

$$\mathcal{L}_m(\mathcal{S}_\ell) = \left\{ w = \ell(\sigma) \in E^* \text{ s.t. } \boldsymbol{m}_0 \left[\sigma\right\rangle \boldsymbol{m}_f \in F \right\}$$

- $\mathcal{L}(\mathcal{S}_\ell)$ and $\mathcal{L}_m(\mathcal{S}_\ell)$ are used as in the automata context for supervisory control based on language specifications

Let $\mathcal{S}_1 = (N_1, \ell_1, \boldsymbol{m}_{0_1}, F_1)$ and $\mathcal{S}_2 = (N_2, \ell_2, \boldsymbol{m}_{0_2}, F_2)$ two labeled systems. Their parallel composition (concurrent composition) $\mathcal{S} = \mathcal{S}_1 \| \mathcal{S}_2$ that generates the language

$$\mathcal{L}(\mathcal{S}) = \mathcal{L}(\mathcal{S}_1) \| \mathcal{L}(\mathcal{S}_2)$$

and marks the language

$$\mathcal{L}_m(\mathcal{S}) = \mathcal{L}_m(\mathcal{S}_1) \| \mathcal{L}_m(\mathcal{S}_2)$$

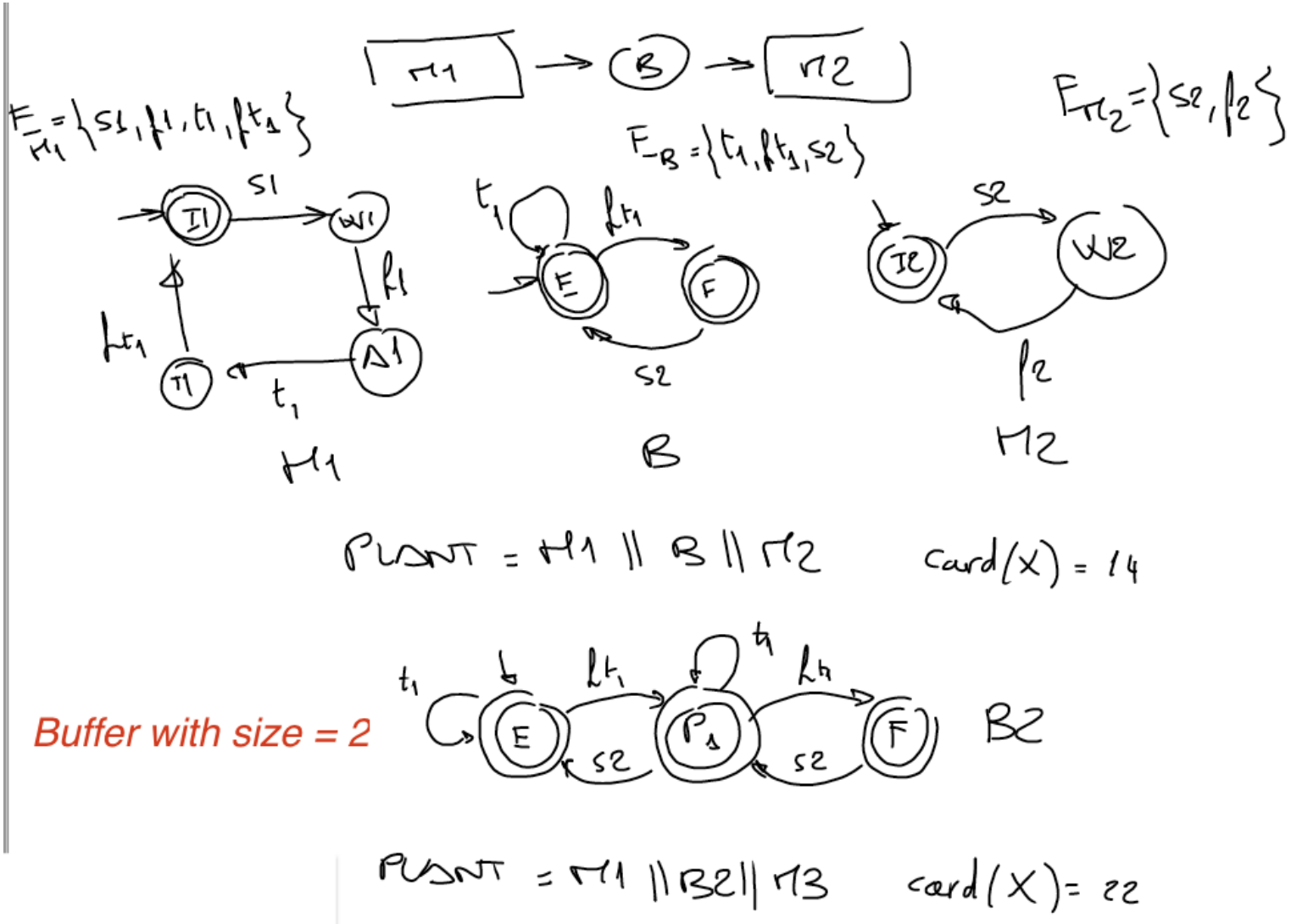
and his given by the following algorithm.

Algorithm for parallel composition of labeled PNs

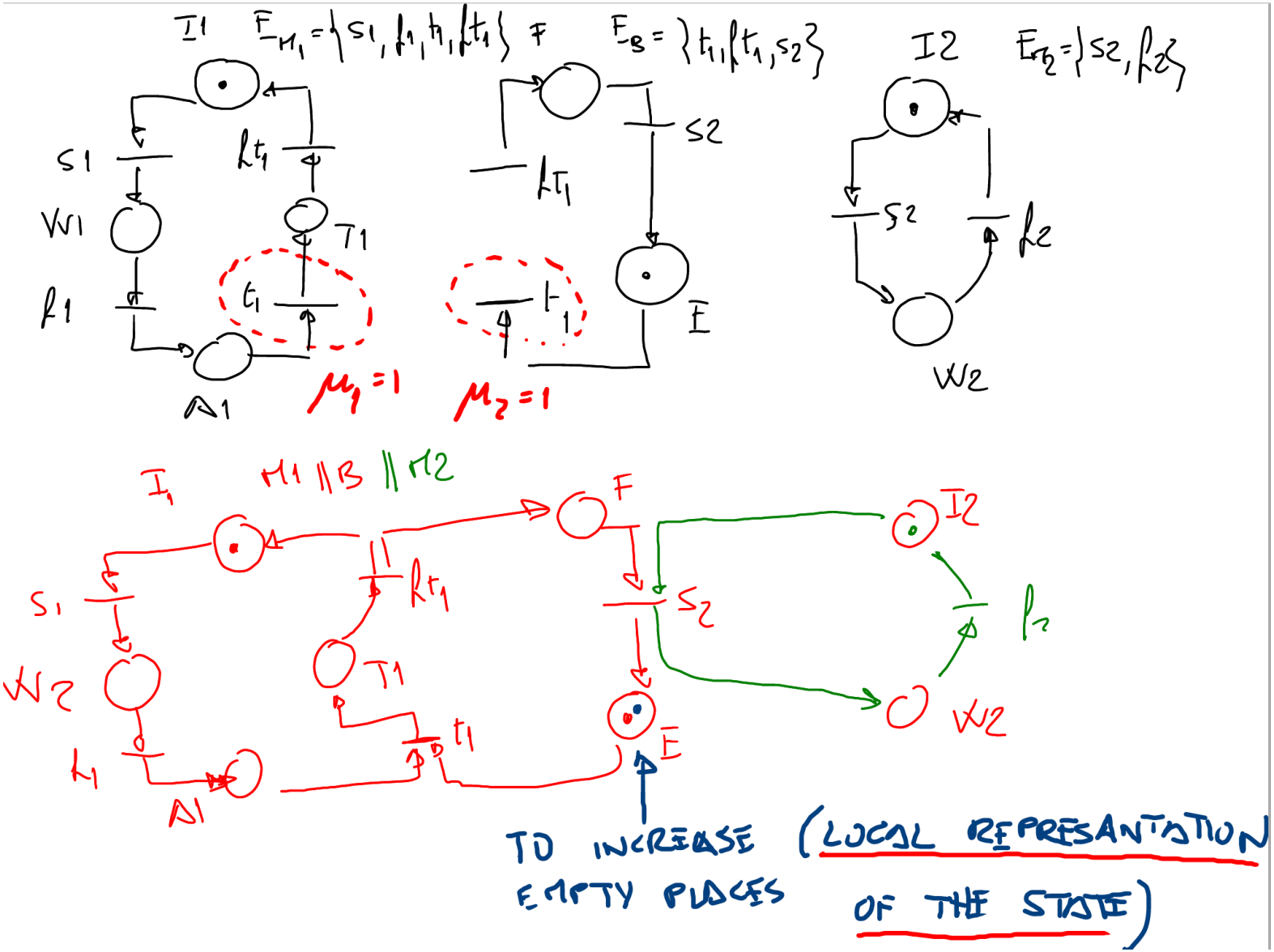
Let P_i, T_i and E_i ($i = 1, 2$) be the place set, transition set, and the alphabet of $\mathcal{S}_i$.

- The place set P of N is the union of the place sets of N_1 and N_2, i.e. $P = P_1 \cup P_2$
- The transition set T of N and the corresponding labels are computed as follows
  - For each transition $t \in T_1 \cup T_2$ labeled ε, a transition with the same ${}^\bullet t$ and $t^\bullet$ and labeled ε belongs to T
  - For each transition $t \in T_1 \cup T_2$ labeled $e \in (E_1 \setminus E_2) \cup (E_2 \setminus E_1)$, a transition with the same ${}^\bullet t$ and $t^\bullet$ and labeled e belongs to T
  - Consider a symbol $e \in E_1 \cap E_2$ and assume it labels μ_1 transitions $T_{e,1} \subseteq T_1$ and μ_2 transitions $T_{e,2} \subseteq T_2$. Then $\mu_1 \times \mu_2$ transitions labeled e belong to T. The input (output) bag of each of these transitions is the sum of the input (output) bags of one transition in $T_{e,1}$ and of one transition in $T_{e,2}$
- $\boldsymbol{m}_0 = \left(\boldsymbol{m}_{0_1}^T \ \boldsymbol{m}_{0_2}^T\right)^T$
- $F = \left\{ \left(\boldsymbol{m}_1^T \ \boldsymbol{m}_2^T\right)^T \mid \boldsymbol{m}_1 \in F_1 \text{ and } \boldsymbol{m}_2 \in F_2 \right\}$

Parallel composition – The FMS example



The FMS example with PNs

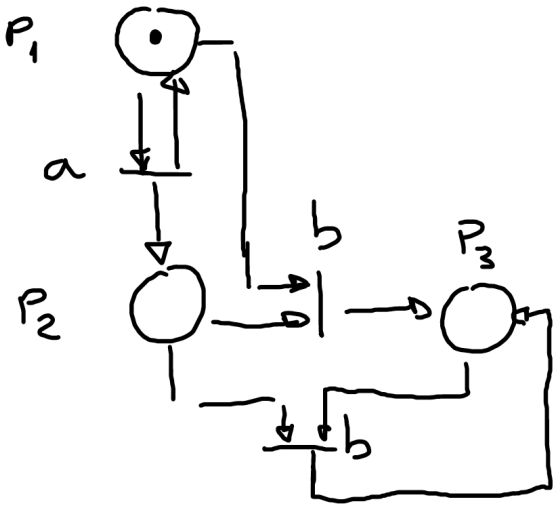


What are the Petri net languages?

- The languages recognized by the finite state automata are the regular languages (Kleene theorem)
- What about *finite* Petri nets, i.e. Petri nets with a finite number of places and transitions?

$$E = \{a, b\} \qquad L = \{w \in E^* \mid w = a^n b^n \text{ with } n > 0\}$$

THIS IS NOT A REGULAR LANGUAGE



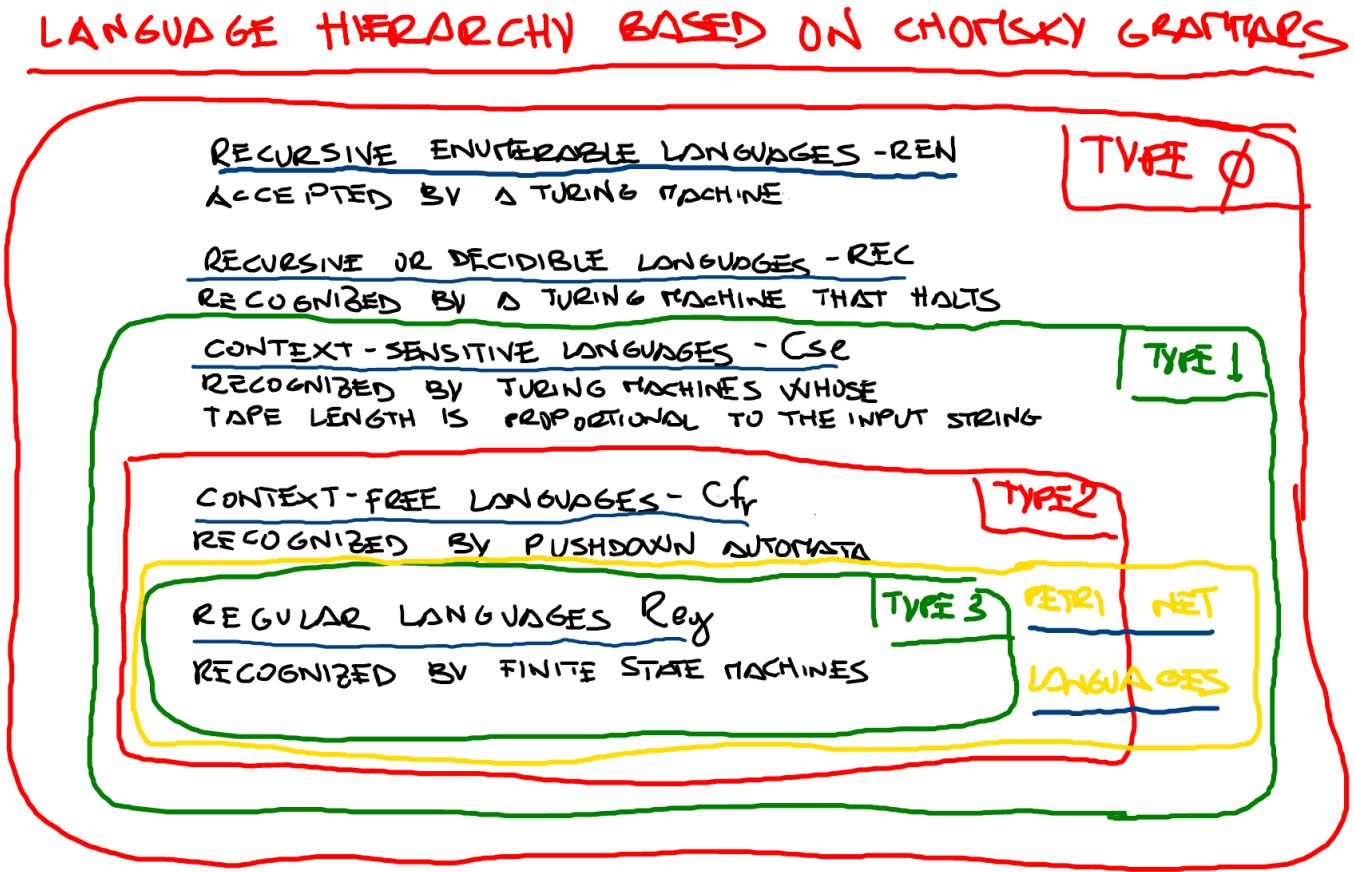
$$m_0 = (1\ 0\ 0)^T$$

FINAL MARKINGS

$$F = \{(0\ 0\ 1)^T\}$$

$\leftarrow \mathcal{L}_m(S_1) = L$

Can Petri nets recognize any language?



1. $Rey \subsetneq Cfr \subsetneq Cse \subsetneq REC \subsetneq REN \neq 2^{\Sigma^*}$
2. $Rey \subset \text{PETRI NET LANGUAGES} \subset Cse$
3. $(\text{PETRI NET LANGUAGES}) \cap Cfr \neq \emptyset$

Twofold representation

- Given their twofold representation – both graphical and algebraic – **linear programming techniques** can be used to assess both structural and behavioral properties in Petri nets systems
- This is one of the main advantages when compared to automata

Example – Reachability

Given a net system $\mathcal{S} = \langle N, \boldsymbol{m}_0 \rangle$ with m places, and a marking $\boldsymbol{m} \in \mathbb{N}^m$, if the following problem is infeasible, then $\boldsymbol{m}$ is unreachable, i.e. $\boldsymbol{m} \notin R(N, \boldsymbol{m}_0)$

$$
\begin{aligned}
&\max \mathbf{0}^{\mathbf{T}} \cdot \boldsymbol{\sigma} \\
&\text{s.t.} \\
&\quad \boldsymbol{C} \cdot \boldsymbol{\sigma} = \boldsymbol{m} - \boldsymbol{m}_0 \\
&\quad \boldsymbol{\sigma} \geq \mathbf{0} \\
&\quad \boldsymbol{\sigma} \in \mathbb{R}^n
\end{aligned}
$$

M. Silva, E. Teruel and J. M. Colom

Linear Algebraic and Linear Programming Techniques for the Analysis of Place/Transition Net Systems

Lectures notes in computer science, 1998

Sufficient condition for boundedness

Given a net system $\mathcal{S} = \langle N, \boldsymbol{m}_0 \rangle$ with m places and $p \in P$, if the solution of the following ILP problem (NP-hard)

$$
\begin{aligned}
& sb(p) = \max \boldsymbol{m}(p) \\
& \text{s.t.} \\
& \boldsymbol{m} = \boldsymbol{m}_0 + \boldsymbol{C} \cdot \boldsymbol{\sigma} \\
& \boldsymbol{m} \geq \mathbf{0} \\
& \boldsymbol{\sigma} \geq \mathbf{0} \\
& \boldsymbol{m} \in \mathbb{N}^m \\
& \boldsymbol{\sigma} \in \mathbb{N}^n
\end{aligned}
$$

is such that

$$sb(p) \leq k$$

then the place p is k-bounded

Example of linear programming relaxation

If the solution of the following LP problem (polynomial time)

$$
\begin{aligned}
& sb(p) = \max \boldsymbol{m}(p) \\
& \text{s.t.} \\
& \boldsymbol{m} = \boldsymbol{m}_0 + \boldsymbol{C} \cdot \boldsymbol{\sigma} \\
& \boldsymbol{m} \geq \mathbf{0} \\
& \boldsymbol{\sigma} \geq \mathbf{0} \\
& \boldsymbol{m} \in \mathbb{R}^m \\
& \boldsymbol{\sigma} \in \mathbb{R}^n
\end{aligned}
$$

is such that

$$\lfloor sb(p) \rfloor \leq k$$

then the place p is k-bounded

Structural boundedness - cont'd

Theorem

Given a net N, the following statements are equivalents

- N is structurally bounded, i.e. every place is bounded for every initial marking $\boldsymbol{m}_0$
- There exists $\boldsymbol{x} > \mathbf{0}$ such that $\boldsymbol{x}^T \cdot \boldsymbol{C} \leq \mathbf{0}$ (integer feasibility problem with linear constraints)
- There does not exists any $\boldsymbol{y} \geq \mathbf{0}$ such that $\boldsymbol{C} \cdot \boldsymbol{y} \gneq \mathbf{0}$ (integer feasibility problem with linear constraints)

Theorem – sufficient condition

Given a net system $\mathcal{S} = \langle N, \boldsymbol{m}_0 \rangle$, if the following feasibility problem **does not** admit any solution $\boldsymbol{m} \in \mathbb{N}^m, \boldsymbol{\sigma} \in \mathbb{N}^n$

$$\begin{aligned} \boldsymbol{m} &= \boldsymbol{m}_0 + \boldsymbol{C} \cdot \boldsymbol{\sigma} \\ \boldsymbol{\sigma} &\geq \mathbf{0} \\ \boldsymbol{m}(p) &\geq 1 \\ \boldsymbol{m}(p') &\geq 1 \end{aligned}$$

then the places p and p' are in *mutual exclusion*

Theorem – sufficient condition

Given a net system $\mathcal{S} = \langle N, \boldsymbol{m}_0 \rangle$, if the following feasibility problem **does not** admit any solution $\boldsymbol{m} \in \mathbb{N}^m, \boldsymbol{\sigma} \in \mathbb{N}^n$

$$\boldsymbol{m} = \boldsymbol{m}_0 + \boldsymbol{C} \cdot \boldsymbol{\sigma}$$
$$\boldsymbol{m} \geq \mathbf{0}$$
$$\boldsymbol{\sigma} \geq \mathbf{0}$$
$$\bigvee_{p \in {}^\bullet t} \boldsymbol{m}(p) < \mathbf{Pre}(p, t), \quad \forall\, t \in T$$

then $\mathcal{S}$ is *deadlock free*

Software tools

- Many tools
- Maybe even more than for automata (due to the infinite number of PNs *subspecies*)
- A database is maintained at **Petri Nets World website**

  `http://www.informatik.uni-hamburg.de/TGI/PetriNets/index.php`
- Many tools have been developed for *model-based verification and validation*
- Some tools
  - TINA (LAAS/CNR)
  - Petri Net Toolbox (University “Gh. Asachi" of Iasi)
  - SNAKES (Python toolkit for coloured Petri nets)
  - . . .

Software tools

- If the algebraic approach is used, tools to solve ILP problems are needed
  - Solvers: CPLEX, XPRESS, GLPK (free, `http://glpkmex.sourceforge.net/`),...
  - YALMIP: a useful Matlab parser for optimization problems `https://yalmip.github.io/`

Summarizing. . .

Petri nets vs automata – Pros. . .

- More powerful expressive power – not just regular languages
- More compact and modular representation
- Twofold representation that permits to use linear programming techniques

. . .and cons

- Lack of necessary and sufficient conditions to due practical semi-decidability of the reachability problem
- Many proposed algorithms are NP-hard . . .which nowadays does not necessarily represent a problem

References


M. P. Cabasino, A. Giua, C. Seatzu
Introduction to Petri nets
in *Control of Discrete-Event Systems*
Springer, 2013


M. P. Cabasino, A. Giua, C. Seatzu
Structural analysis of Petri nets
in *Control of Discrete-Event Systems*
Springer, 2013

Petri nets and their twofold representation to model DES

From observability to privacy and security in discrete event systems

Prof. Gianmaria DE TOMMASI
Email: detommas@unina.it

December 2020

DIETI UNIVERSITA' DEGLI STUDI DI NAPOLI FEDERICO II
DIPARTIMENTO DI INGEGNERIA ELETTRICA E DELLE TECNOLOGIE DELL'INFORMAZIONE