

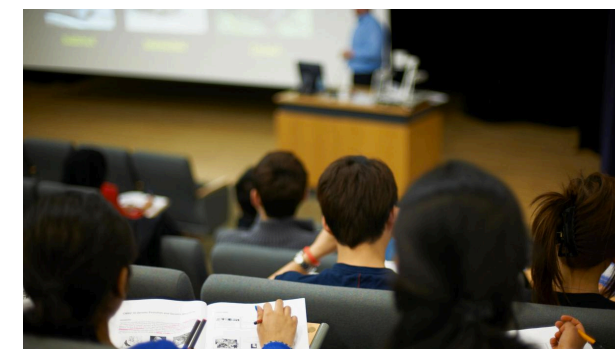
PMR 5237

Modelagem e Design de Sistemas Discretos em Redes de Petri

Aula 8: Análise de Invariantes

Prof. José Reinaldo Silva
reinaldo@usp.br

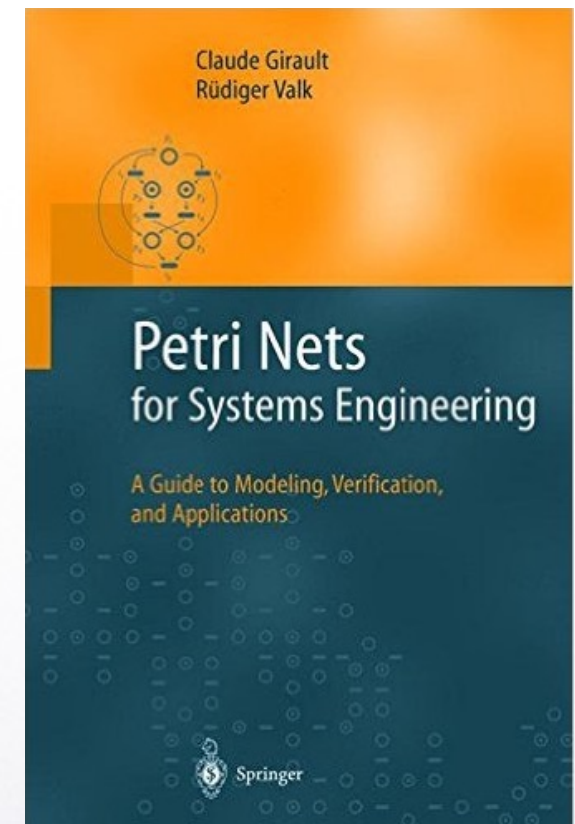
Plano de Aulas



Aula	Tema	Data
Aula7	Propriedades das redes P/T e Coloridas	9/11
Aula8	<u>Análise de Invariantes</u>	16/11
Aula9	Técnicas de modelagem	23/11
Aula10	Métodos de Design orientados a estados	30/11
Aula11	Métodos de Design orientados a eventos	07/12
Aula12	Perspectivas de pesquisa em modelagem de sistemas discretos com RdP	14/12

Sincronismo com o livro texto

1. Introduction	7
2. Essential Features of Petri Nets	9
2.1 Locality and Concurrency	10
2.2 Graphical and Algebraic Representation	12
2.3 Concurrency, Conflict, and Confusion	15
2.4 Refinement and Composition	18
2.5 Net Morphisms	23
3. Intuitive Models	29
3.1 Arc-Constant Nets	29
3.2 Place/Transition Nets	32
3.3 Coloured Nets	34
3.4 Foldings	38
4. Basic Definitions	41
4.1 Formal Definition of Place/Transition Nets	41
4.2 Formal Definition of Arc-Constant Nets	43
4.3 Formal Definition of Coloured Nets	45
5. Properties	53
5.1 Basic Properties	54
5.2 An Introduction to the Analysis	58
5.2.1 Verification Based on the Reachability Graph	60
5.2.2 Verification Based on Linear Invariants	68
6. Overview of the Book	73



Na aula passada exploramos as propriedades das redes clássicas e das redes coloridas, com o objetivo de mostrar que as principais propriedades que embasam os métodos de análise de sistemas podem ser aplicadas em ambos os tipos de rede.



PN Basic Properties

- 1) *boundedness*, characterising finiteness of the state space.
- 2) *liveness*, related to potential fireability in all reachable markings. *Deadlock-freeness* is a weaker condition in which global infinite activity (i.e. fireability) of the net system model is guaranteed, but some parts of it may not work at all.
- 3) *reversibility*, characterising recoverability of the initial marking from any reachable marking.
- 4) *mutual exclusion*, dealing with the impossibility of simultaneous *submarkings* (p-mutex) or *firing concurrency* (t-mutex).

- 1) *boundedness*, characterising finiteness of the state space.
- 2) *liveness*, related to potential fireability in all reachable markings. *Deadlock-freeness* is a weaker condition in which global infinite activity (i.e. fireability) of the net system model is guaranteed, but some parts of it may not work at all.
- 3) *reversibility*, characterising recoverability of the initial marking from any reachable marking.
- 4) *mutual exclusion*, dealing with the impossibility of simultaneous *submarkings* (p-mutex) or *firing concurrency* (t-mutex).

A Petri net (N, M_0) is said to be k -bounded, or simply bounded, if the number of tokens in each place reachable from M_0 does not exceed a finite number k , i.e., $M(p) \leq k$ for every place p and every marking $M \in R(M_0)$.

Let be a CPN net N , a generic place p in this net, and $|M(p)|$ the size of the multiset making M . The place p is said bounded iff there is a multiset n which is an upper bound of $M(p)$, that is, $\forall M \in \mathcal{R}(M_0). |M(p)| \ll = n$.

A CPN net is said to be bounded if all its places are bounded.

- 1) *boundedness*, characterising finiteness of the state space.
- 2) *liveness*, related to potential fireability in all reachable markings. *Deadlock-freeness* is a weaker condition in which global infinite activity (i.e. fireability) of the net system model is guaranteed, but some parts of it may not work at all.
- 3) *reversibility*, characterising recoverability of the initial marking from any reachable marking.
- 4) *mutual exclusion*, dealing with the impossibility of simultaneous *submarkings* (p-mutex) or *firing concurrency* (t-mutex).

Now, the *reachability problem* for Petri nets is the problem of finding if $M_n \in R(M_0)$ for a given marking M_n in a net (N, M_0) . In some applications, one may be interested in the markings of a subset of places and not care about the rest of places in a net. This leads to a *submarking reachability*

Petri Nets: Properties, Analysis and Applications

TADAO MURATA, FELLOW, IEEE

Liveness (T. Murata)

A transition t is said to be...

- 0) *dead (L0-live)* if t can never be fired in any firing sequence in $L(M_0)$.
- 1) *L1-live (potentially firable)* if t can be fired at least once in some firing sequence in $L(M_0)$.
- 2) *L2-live* if, given any positive integer k , t can be fired at least k times in some firing sequence in $L(M_0)$.
- 3) *L3-live* if t appears infinitely, often in some firing sequence in $L(M_0)$.
- 4) *L4-live or live* if t is *L1-live* for every marking M in $R(M_0)$.

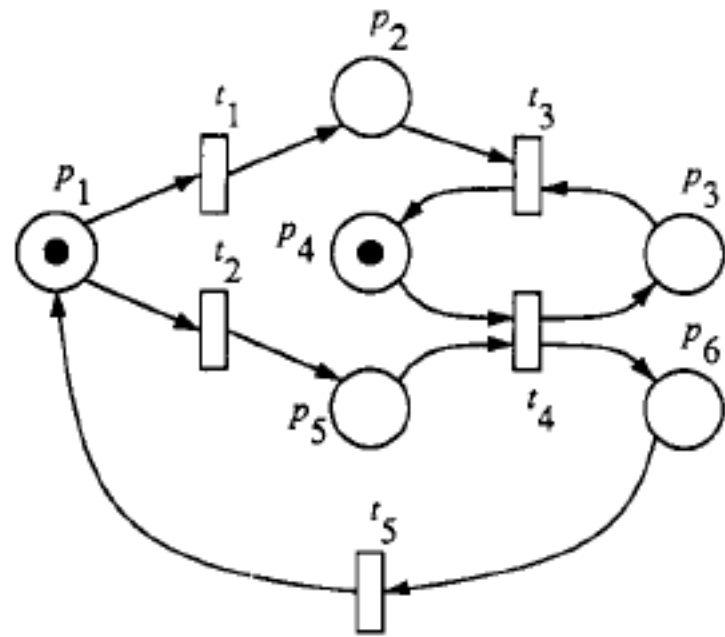
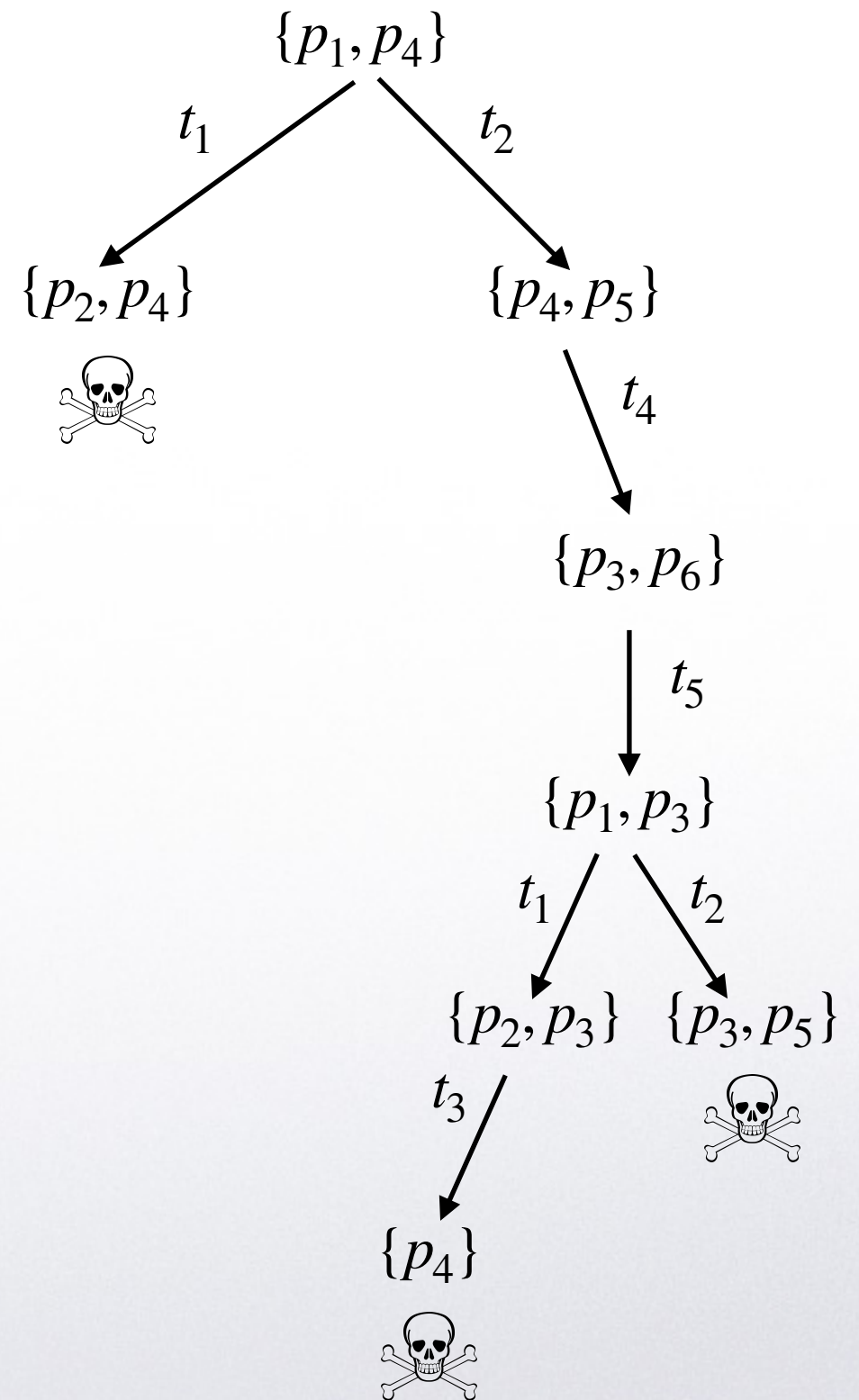


Fig. 15. A safe, nonlive Petri net. But it is strictly $L1$ -live.



A very important goal for analysis is if a given system is deadlock-free.

4) *L4-live or live if t is L1-live for every marking M in $R(M_0)$.*

Def] A transition t is potentially fireable at a given marking m if there exists a transition iff there is a firing sequence σ leading to a marking m' in which t is enabled (i.e. $m \xrightarrow{\sigma} m' \geq Pre[P, t]$).

A P/T net is deadlock-free iff all its transitions are potentially fireable.

- 1) *boundedness*, characterising finiteness of the state space.
- 2) *liveness*, related to potential fireability in all reachable markings. *Deadlock-freeness* is a weaker condition in which global infinite activity (i.e. fireability) of the net system model is guaranteed, but some parts of it may not work at all.
- 3) *reversibility*, characterising recoverability of the initial marking from any reachable marking.
- 4) *mutual exclusion*, dealing with the impossibility of simultaneous *submarkings* (p-mutex) or *firing concurrency* (t-mutex).

A CPN net N is said to be **live** iff $\forall M \exists L \mid M \in L(M_0)$.

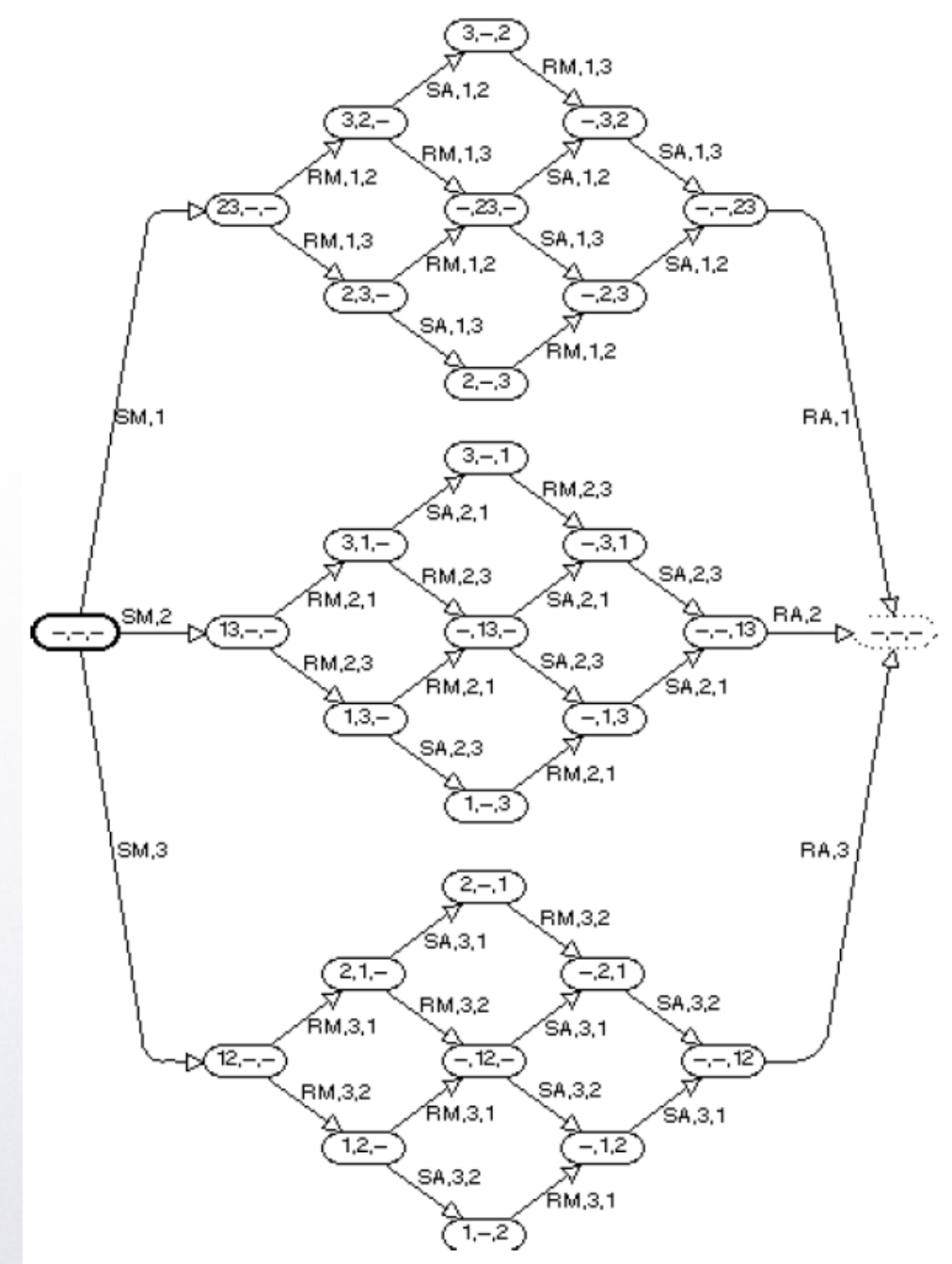
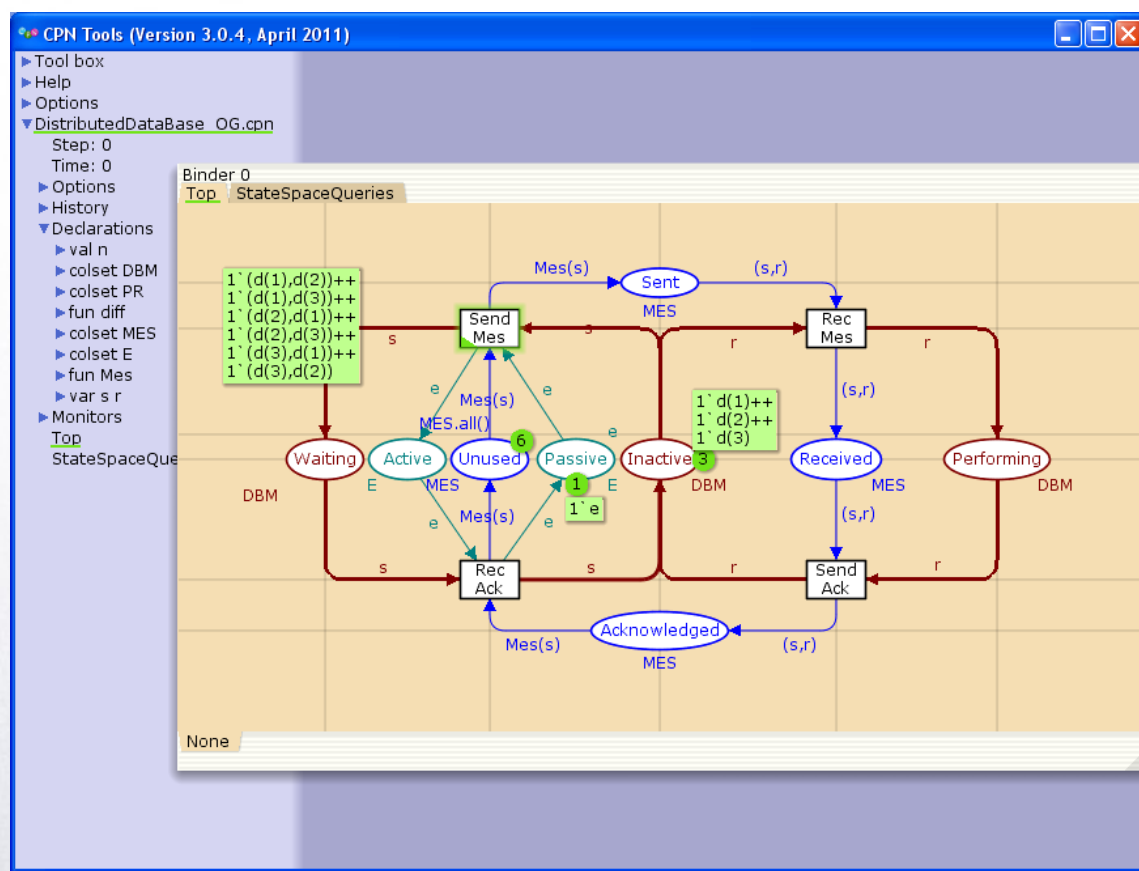
A CPN net N is said to be **strongly live** iff $\forall M . M \in SCC(M_0)$

CPN Reachability Analysis

- 1) *boundedness*, characterising finiteness of the state space.
- 2) *liveness*, related to potential fireability in all reachable markings. *Deadlock-freeness* is a weaker condition in which global infinite activity (i.e. fireability) of the net system model is guaranteed, but some parts of it may not work at all.
- 3) *reversibility*, characterising recoverability of the initial marking from any reachable marking.
- 4) *mutual exclusion*, dealing with the impossibility of simultaneous *submarkings* (p-mutex) or *firing concurrency* (t-mutex).

Reachability analysis in CPN is connected to building a “reachability graph”, which is now defined over a special directed graph called **CPN state space**.

Analysis in CPN Nets



- 1) *boundedness*, characterising finiteness of the state space.
- 2) *liveness*, related to potential fireability in all reachable markings. *Deadlock-freeness* is a weaker condition in which global infinite activity (i.e. fireability) of the net system model is guaranteed, but some parts of it may not work at all.
- 3) *reversibility*, characterising recoverability of the initial marking from any reachable marking.
- 4) *mutual exclusion*, dealing with the impossibility of simultaneous *submarkings* (p-mutex) or *firing concurrency* (t-mutex).

Def] Let be the P/T net $N = (S, T, F; W, M_0)$. A generic state defined over this net $X \subset S$ is said to be a **home state** iff X can be reached from any reachable marking.



- 1) *boundedness*, characterising finiteness of the state space.
- 2) *liveness*, related to potential fireability in all reachable markings. *Deadlock-freeness* is a weaker condition in which global infinite activity (i.e. fireability) of the net system model is guaranteed, but some parts of it may not work at all.
- 3) *reversibility*, characterising recoverability of the initial marking from any reachable marking.
- 4) *mutual exclusion*, dealing with the impossibility of simultaneous *submarkings* (p-mutex) or *firing concurrency* (t-mutex).

A Petri net (N, M_0) is said to be *reversible* if, for each marking M in $R(M_0)$, M_0 is reachable from M . Thus, in a reversible net one can always get back to the initial marking or state.

Petri Nets: Properties, Analysis and Applications

TADAO MURATA, FELLOW, IEEE

Existence of home states is undecidable

Eike Best^{1*}, J...

¹ Department
Carl von Ossietzky
eike.best@i...

² Fac
Technisch

Abstract. W
a home sta
decidable
Keywo

1 In

From the concept of "home states", which are defined as states that can be reached from whichever state the system might be in. In various applications, home states may be entered automatically after periods of inactivity, or may be forced to be reached by pushing a "reset" button. In self-stabilising systems [3], failure states can be recovered from automatically, preferably ending up in regular, non-erroneous home states. In Markov chain theory, home states are called "essential" states [2] a particularly important class being that of the "recurrent" states.

The main two decision problems concerning home states are (1) given a dynamic system S and a state q , is q a home state of S ? and (2) given a system S , does it have a home state? We call them the home state problem (HSP) and the home state existence problem (HSEP), respectively.

Frequently, dynamic systems must have a "home state", defined as states that can be reached from whichever state the system might be in.

- 1) *boundedness*, characterising finiteness of the state space.
- 2) *liveness*, related to potential fireability in all reachable markings. *Deadlock-freeness* is a weaker condition in which global infinite activity (i.e. fireability) of the net system model is guaranteed, but some parts of it may not work at all.
- 3) *reversibility*, characterising recoverability of the initial marking from any reachable marking.
- 4) *mutual exclusion*, dealing with the impossibility of simultaneous *submarkings* (p-mutex) or *firing concurrency* (t-mutex).

The proposition of algorithms and methods to detect home states in CPNs is still an open problem. However direct graphs can also be used to spot loops, which are essential (necessary condition) to reversibility.

Definition 9.17. Let M_{home} be a marking and M_{home}^* a set of markings.

1. M_{home} is a home marking if and only if

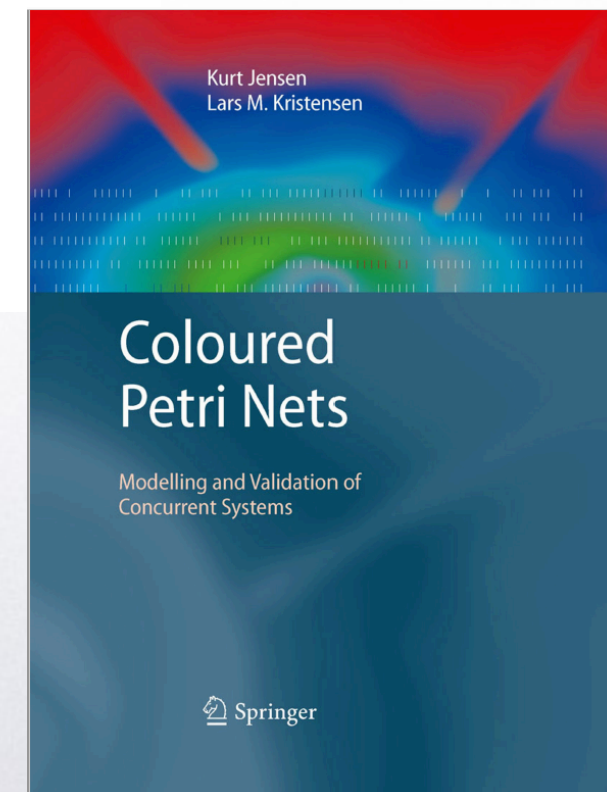
$$\forall M \in \mathcal{R}(M_0) : M_{home} \in \mathcal{R}(M)$$

2. M_{home}^* is a home space if and only if

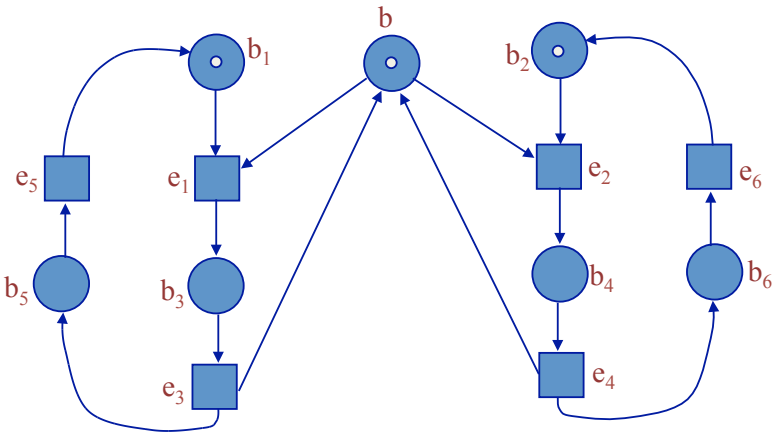
$$\forall M \in \mathcal{R}(M_0) \exists M' \in \mathcal{R}(M) : M' \in M_{home}^*$$

3. A predicate ϕ on markings is a home predicate if and only if

$$\forall M \in \mathcal{R}(M_0) \exists M' \in \mathcal{R}(M) : \phi(M')$$



Em uma situação genérica a mútua exclusão (mutex) é caracterizada por processos (transições independentes que compartilham pelo menos um lugar. O início de um processo desabilita este lugar e faz com que o outro seja desativado.



- 1) *boundedness*, characterising finiteness of the state space.
- 2) *liveness*, related to potential fireability in all reachable markings. *Deadlock-freeness* is a weaker condition in which global infinite activity (i.e. fireability) of the net system model is guaranteed, but some parts of it may not work at all.
- 3) *reversibility*, characterising recoverability of the initial marking from any reachable marking.
- 4) *mutual exclusion* dealing with the impossibility of simultaneous *submarkings* (p-mutex) or *firing concurrency* (t-mutex).



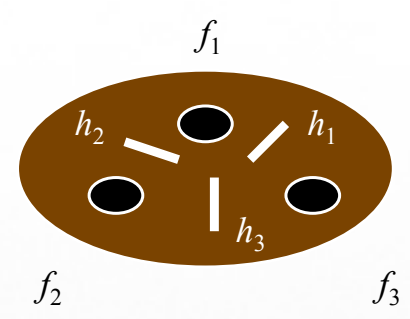
A Petri net (N, M_0) is said to be *persistent* if, for any two enabled transitions, the firing of one transition will not disable the other. A transition in a persistent net, once it is enabled, will stay enabled until it fires.

Petri Nets: Properties, Analysis and Applications

TADAO MURATA, FELLOW, IEEE

O exemplo dos filósofos

- 1) *boundedness*, characterising finiteness of the state space.
- 2) *liveness*, related to potential fireability in all reachable markings. *Deadlock-freeness* is a weaker condition in which global infinite activity (i.e. fireability) of the net system model is guaranteed, but some parts of it may not work at all.
- 3) *reversibility*, characterising recoverability of the initial marking from any reachable marking.
- 4) *mutual exclusion*, dealing with the impossibility of simultaneous *submarkings* (p-mutex) or *firing concurrency* (t-mutex).



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

$$H = \{h_1, h_2, h_3\}$$

$$U = P \cup H$$

$$l : P \rightarrow H$$

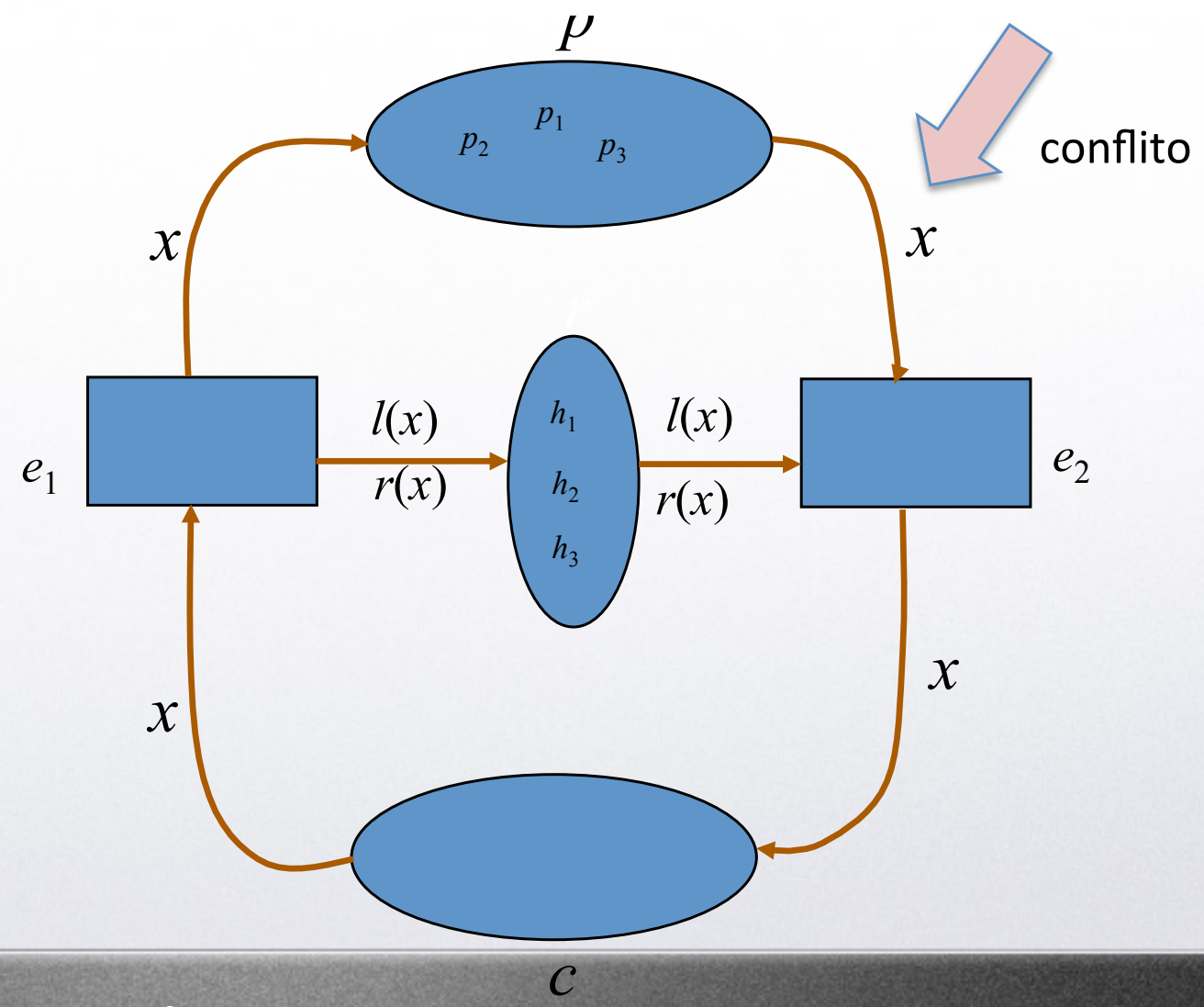
$$p_i \rightarrow h_i$$

$$r : P \rightarrow H$$

$$p_1 \rightarrow h_2$$

$$p_2 \rightarrow h_3$$

$$p_3 \rightarrow h_1$$



O exemplo dos filósofos

$P = \{p_1, p_2, p_3\}$
 $H = \{h_1, h_2, h_3\}$
 $U = P \cup H$

$l : P \rightarrow H$

$p_i \rightarrow h_i$

$r : P \rightarrow H$

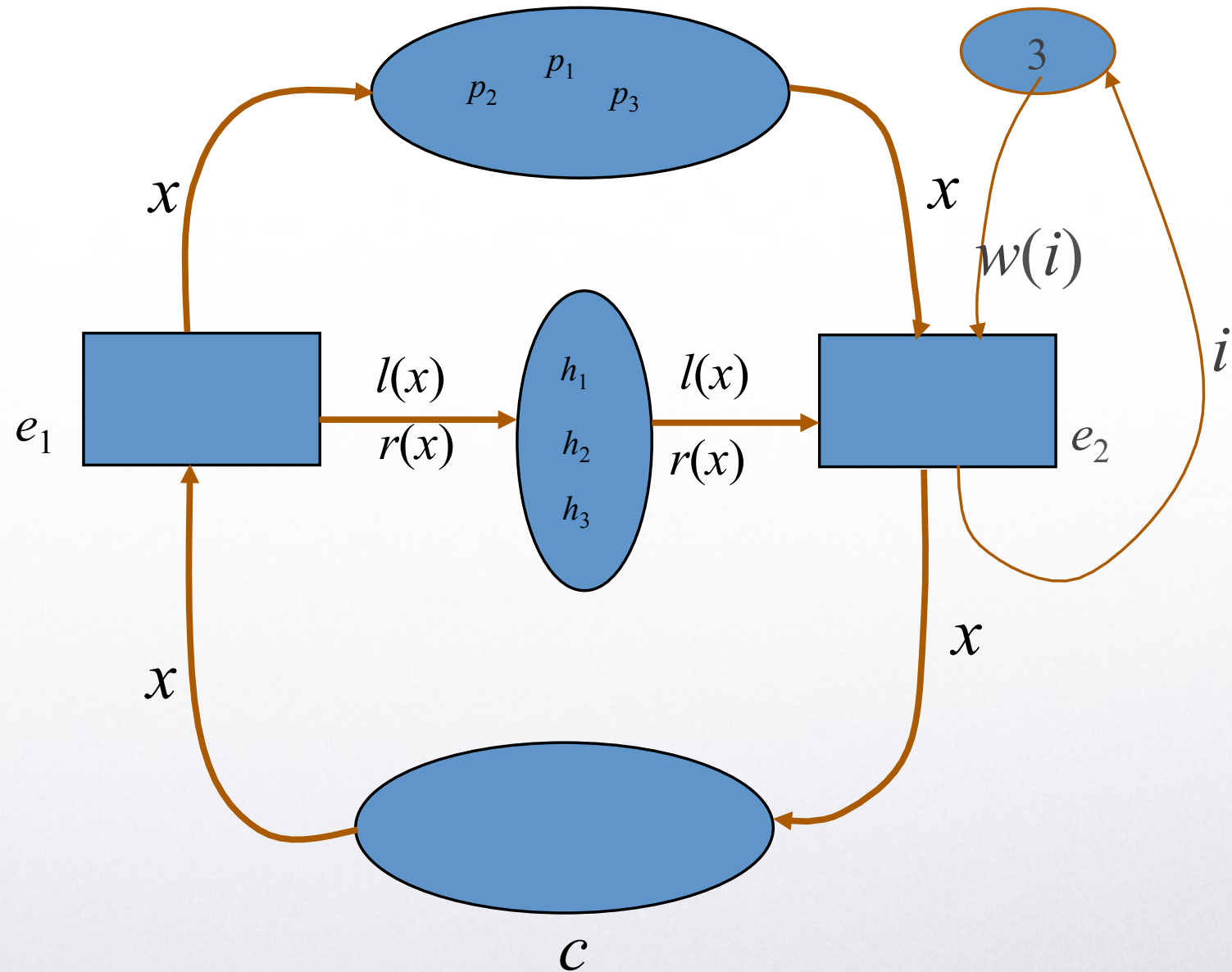
$p_1 \rightarrow h_2$

$p_2 \rightarrow h_3$

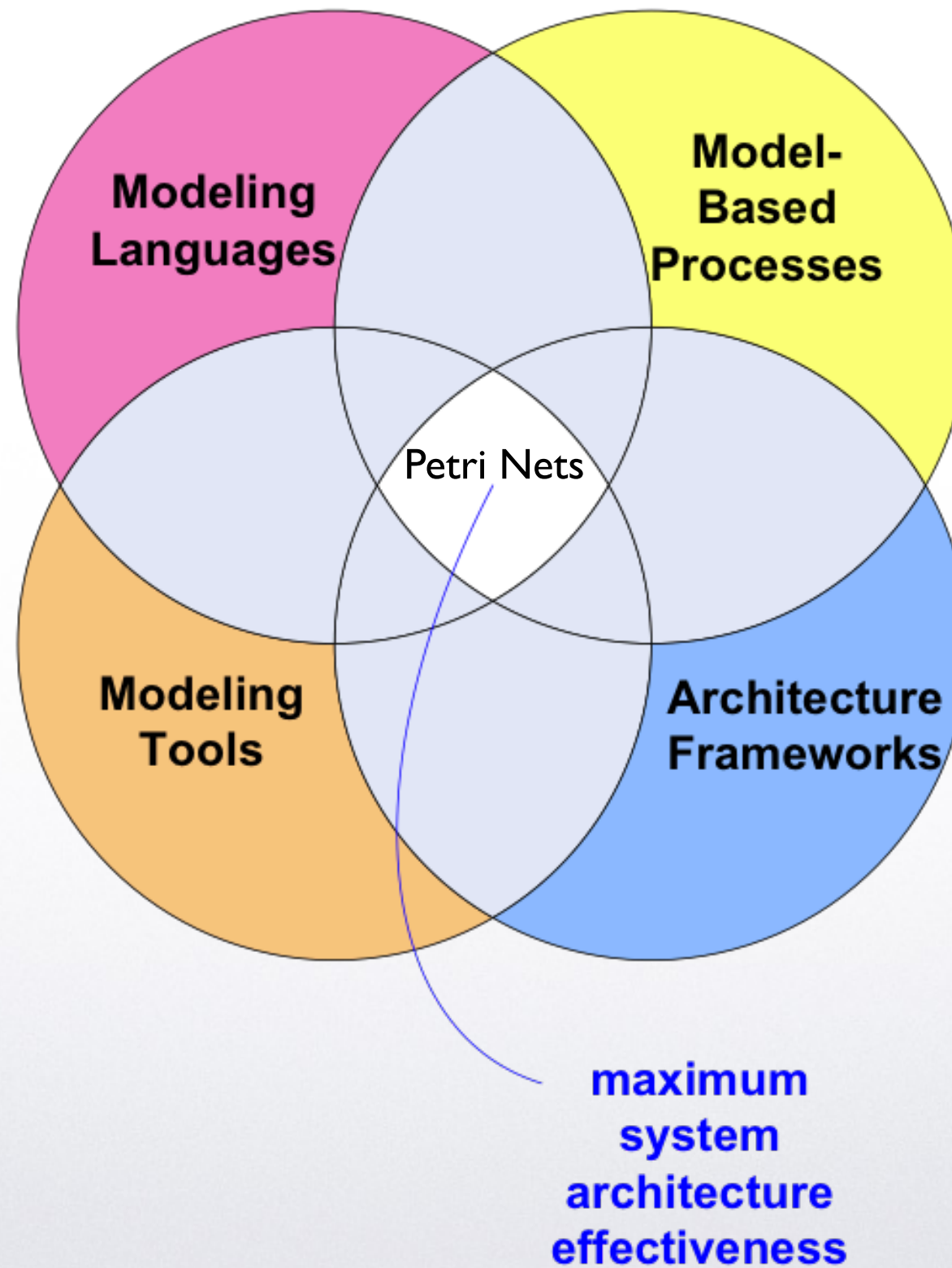
$p_3 \rightarrow h_1$

$w : P \rightarrow I$

$p_j \rightarrow j = (i) \bmod 3 + 1$

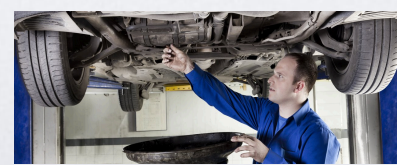
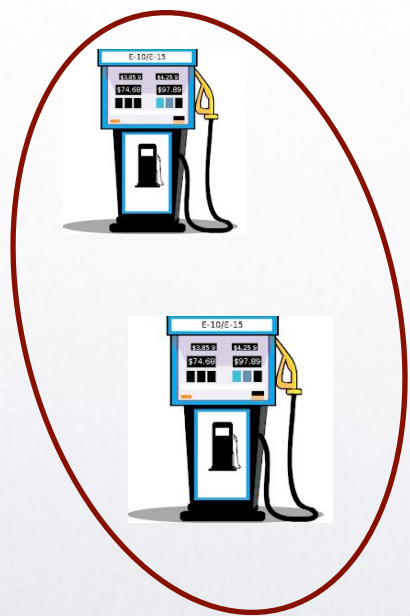


- 1) *boundedness*, characterising finiteness of the state space.
- 2) *liveness*, related to potential fireability in all reachable markings. *Deadlock-freeness* is a weaker condition in which global infinite activity (i.e. fireability) of the net system model is guaranteed, but some parts of it may not work at all.
- 3) *reversibility*, characterising recoverability of the initial marking from any reachable marking.
- 4) *mutual exclusion*, dealing with the impossibility of simultaneous *submarkings* (p-mutex) or *firing concurrency* (t-mutex).



maximum
system
architecture
effectiveness

Modelar esse sistema em redes CPN (sem um modelo prévio em redes P/T).



Invariants

Generally speaking, an **invariant** is a quantity that remains constant during the execution of a given **algorithm**. In other words, none of the allowed operations changes the value of the invariant. The invariant principle is extremely useful in analyzing the end result (or possible end results) of an algorithm, because we can discard any potential result that has a different value for the invariant as impossible to reach.

Alexander Katz, Pi Han Goh, Geoff Pilling



History

The formal study of algebraic (linear) invariants began in 1840 with the seminal paper of George Boole, "Exposition of a General Theory of Linear Transformations". The idea was to identify functions (or vectors) that do not depend on the vectorial basis of the linear transformation.

Invariant Analysis

Invariant analysis is concerned with a perception of conservation laws in the flux of changing, for instance, the repetition of states (cycles). Therefore, in state-transition (or transition systems) analysis, invariants could be associated with structural properties which could also affect systems behavior.

A *p-flow* (*t-flow*) is a vector $y : P \rightarrow \mathbb{Q}$ such that $y \cdot C = 0$ ($x : T \rightarrow \mathbb{Q}$ such that $C \cdot x = 0$), where C is the incidence matrix of the net. The set of *p-flows* (*t-flows*) is a vector space, orthogonal to the space of the rows (columns) of C .

Therefore, the flows can be generated from a *basis* of the space. Natural (i.e. non-negative integer) *p-flows* (*t-flows*) are called *p-semiflows* (*t-semiflows*): vectors $y : P \rightarrow \mathbb{N}$ such that $y \cdot C = 0$ ($x : T \rightarrow \mathbb{N}$ such that $C \cdot x = 0$).

$$M_{i+1} = M_0 + \sum (A^T \cdot \sigma_i) \Rightarrow M_{i+1} - M_0 = A^T \cdot \bar{\sigma}$$

$$M_{i+1} - M_0 = \Delta M = A^T \cdot \bar{\sigma}; \exists \beta \mid \beta \cdot \Delta M = 0$$

$$\beta \cdot \Delta M = (\beta \cdot A^T) \cdot \bar{\sigma} = 0 \Rightarrow \underbrace{(\beta \cdot A^T)}_y = 0$$

$\underbrace{\hspace{10em}}_C$

A *p-flow* (*t-flow*) is a vector $y : P \rightarrow \mathbb{Q}$ such that $y \cdot C = 0$ ($x : T \rightarrow \mathbb{Q}$ such that $C \cdot x = 0$), where C is the incidence matrix of the net. The set of *p-flows* (*t-flows*) is a vector space, orthogonal to the space of the rows (columns) of C .

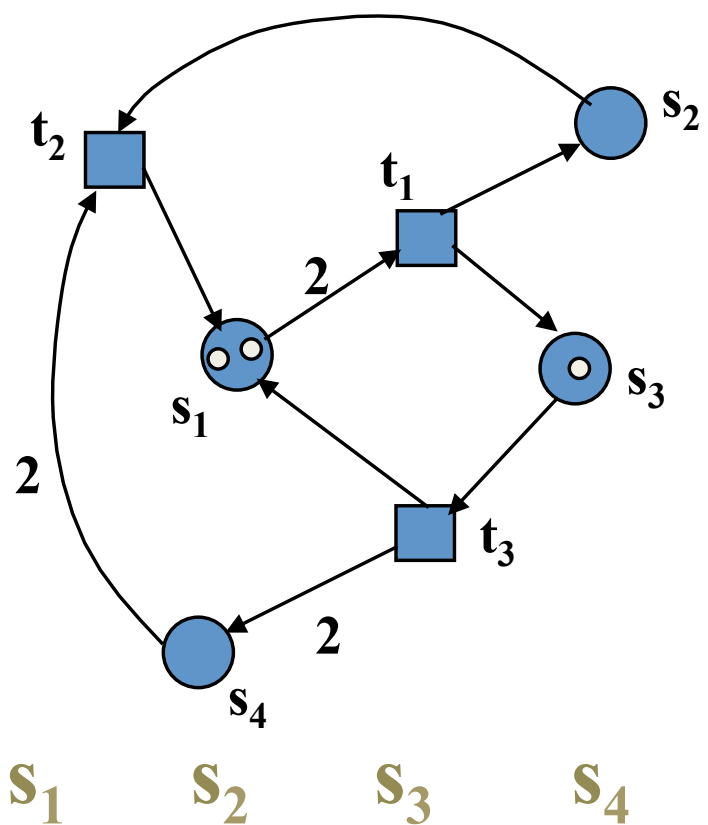
Therefore, the flows can be generated from a *basis* of the space. Natural (i.e. non-negative integer) *p-flows* (*t-flows*) are called *p-semiflows* (*t-semiflows*): vectors $y : P \rightarrow \mathbb{N}$ such that $y \cdot C = 0$ ($x : T \rightarrow \mathbb{N}$ such that $C \cdot x = 0$).

$$M_{i+1} = M_0 + \sum (A^T \cdot \sigma_i) \Rightarrow M_{i+1} - M_0 = A^T \cdot \bar{\sigma}$$

$$M_{i+1} - M_0 = \Delta M = 0 \iff \underset{C}{A^T} \cdot \underset{x}{\bar{\sigma}} = 0$$

O cálculo de invariantes nas redes clássicas implica na resolução das equações lineares $y \cdot C = 0$ ou $C \cdot x = 0$, onde a matriz de incidência não é necessariamente quadrada. Portanto os métodos diretos não podem ser aplicados.

Equação de Estado



$$\mathbf{M}_{i+1} = \mathbf{M}_i + \mathbf{A}^T \boldsymbol{\sigma}_i$$

	s_1	s_2	s_3	s_4	
$\mathbf{A} =$	$\begin{pmatrix} -2 & 1 & 1 & 0 \\ 1 & -1 & 0 & -2 \\ 1 & 0 & -1 & 2 \end{pmatrix}$				t_1
					t_2
					t_3

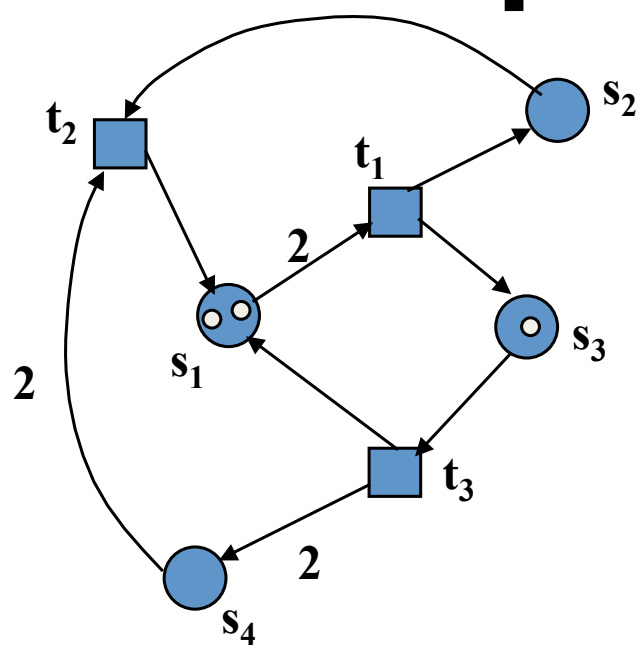
$\boldsymbol{\sigma} =$	$\begin{pmatrix} 1 \\ 0 \\ 1 \end{pmatrix}$	$\mathbf{M}_0 =$	$\begin{pmatrix} 2 \\ 0 \\ 1 \\ 0 \end{pmatrix}$
-------------------------	---	------------------	--

Determinação de B_f

$$\mathbf{A} = \begin{array}{c} \begin{array}{cc} \underline{m-r} & \underline{r} \\ \mathbf{A}_{11} & \mathbf{A}_{12} \\ \mathbf{A}_{21} & \mathbf{A}_{22} \end{array} \\ \left. \begin{array}{l} | \mathbf{r} \\ | \mathbf{n-r} \end{array} \right\} \end{array}$$

$$\mathbf{B}_f = [\mathbf{I}_{m-r} : -\mathbf{A}_{11}^T (\mathbf{A}_{12}^T)^{-1}]$$

Voltando ao exemplo



$$\mathbf{B}_f = \left[\begin{pmatrix} 1 & 0 \\ 0 & 1 \end{pmatrix} : - \begin{pmatrix} -2 & 1 \\ 1 & -1 \end{pmatrix} \begin{pmatrix} 1 & 0 \\ 0 & -1/2 \end{pmatrix} \right] =$$

$$= \begin{pmatrix} 1 & 0 & 2 & 1/2 \\ 0 & 1 & -1 & -1/2 \end{pmatrix}$$

s_1 s_2 s_3 s_4

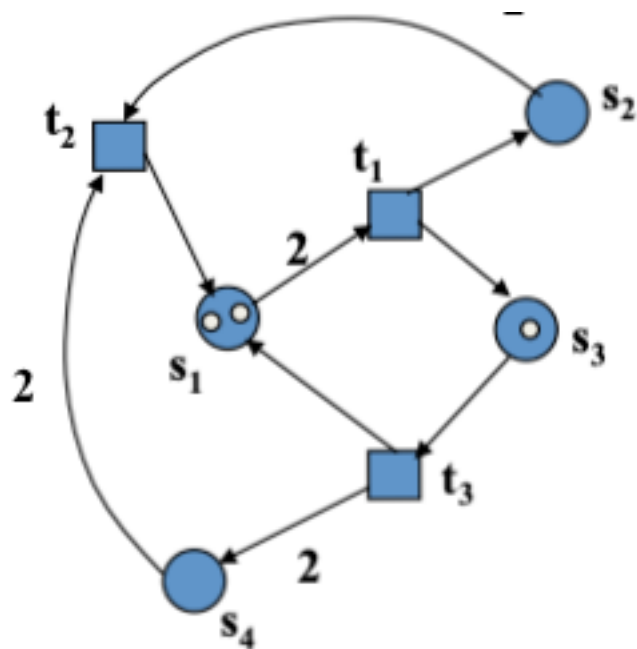
$$\mathbf{A} = \begin{pmatrix} \boxed{-2} & \boxed{1} & \boxed{1} & \boxed{0} \\ \boxed{1} & \boxed{-1} & \boxed{0} & \boxed{-2} \\ \boxed{1} & \boxed{0} & \boxed{-1} & \boxed{2} \end{pmatrix}$$

t_1
 t_2
 t_3

$$\mathbf{A} = \begin{matrix} \begin{matrix} \underline{m-r} & \underline{r} \end{matrix} \\ \begin{bmatrix} \mathbf{A}_{11} & \mathbf{A}_{12} \\ \mathbf{A}_{21} & \mathbf{A}_{22} \end{bmatrix} \begin{matrix} | \mathbf{r} \\ | \mathbf{n-r} \end{matrix} \end{matrix}$$

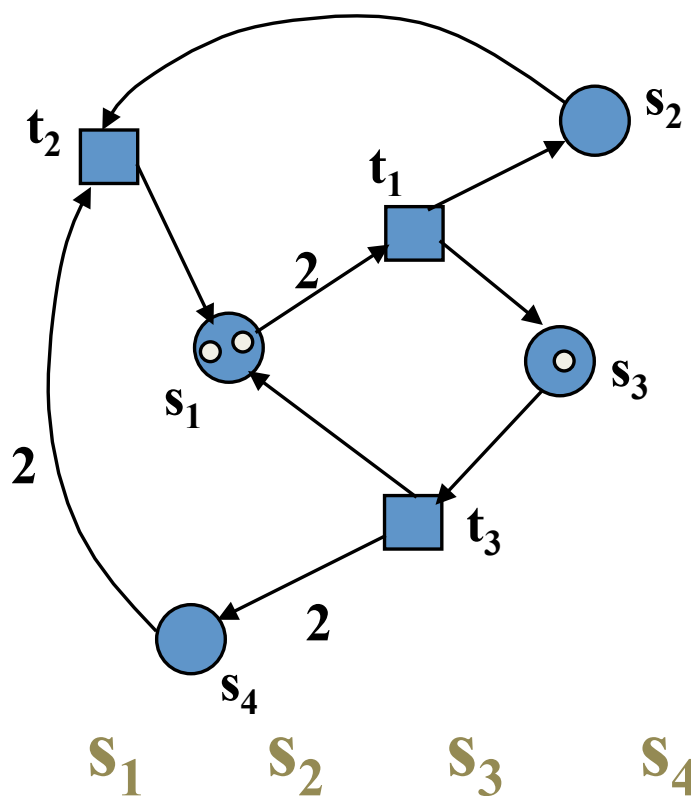
$$\mathbf{B}_f = [\mathbf{I}_{m-r} : -\mathbf{A}_{11}^T (\mathbf{A}_{12}^T)^{-1}]$$

No exemplo utilizado até aqui ...



$$\begin{pmatrix} 2 \\ 0 \\ 1 \\ 0 \end{pmatrix} \xRightarrow{t_1} \begin{pmatrix} 0 \\ 1 \\ 2 \\ 0 \end{pmatrix} \xRightarrow{t_3} \begin{pmatrix} 1 \\ 1 \\ 1 \\ 2 \end{pmatrix}$$

$$B_f \Delta M = B_f \left[\begin{pmatrix} 2 \\ 0 \\ 1 \\ 0 \end{pmatrix} - \begin{pmatrix} 0 \\ 1 \\ 2 \\ 0 \end{pmatrix} \right] = \begin{pmatrix} 1 & 0 & 2 & 1/2 \\ 0 & 1 & -1 & -1/2 \end{pmatrix} \begin{pmatrix} 2 \\ -1 \\ -1 \\ 0 \end{pmatrix} = \begin{pmatrix} 0 \\ 0 \end{pmatrix}$$

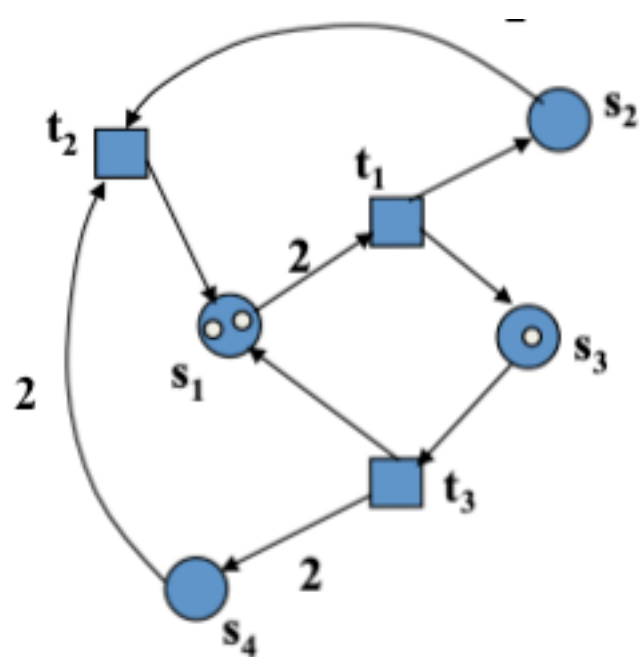


$$\begin{pmatrix} -2 & 1 & 1 \\ 1 & -1 & 0 \\ 1 & 0 & -1 \\ 0 & -2 & 2 \end{pmatrix} \begin{pmatrix} x \\ y \\ z \end{pmatrix} = \begin{pmatrix} 0 \\ 0 \\ 0 \\ 0 \end{pmatrix}$$

$x = y = z$

$$\mathbf{A} = \begin{pmatrix} -2 & 1 & 1 & 0 \\ 1 & -1 & 0 & -2 \\ 1 & 0 & -1 & 2 \end{pmatrix} \begin{matrix} t_1 \\ t_2 \\ t_3 \end{matrix}$$

Em primeira determinação, isto é, fazendo com que $x=y=z$ assumam o menor inteiro positivo, temos que $x=y=z=1$



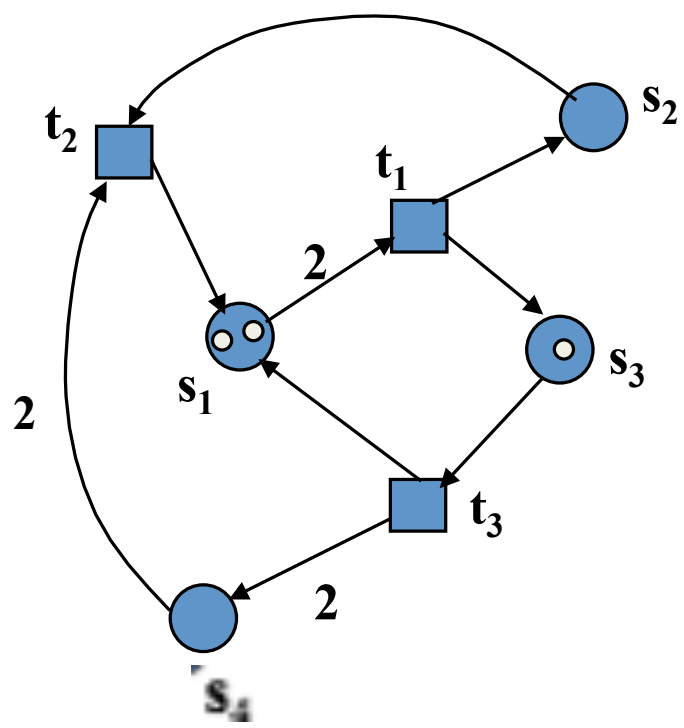
$$\begin{pmatrix} -2 & 1 & 1 \\ 1 & -1 & 0 \\ 1 & 0 & -1 \\ 0 & -2 & 2 \end{pmatrix} \begin{pmatrix} x \\ y \\ z \end{pmatrix} = \begin{pmatrix} 0 \\ 0 \\ 0 \\ 0 \end{pmatrix}$$

$$-2x + y + z = 0$$

$$x - y = 0$$

$$-2y + 2z = 0$$

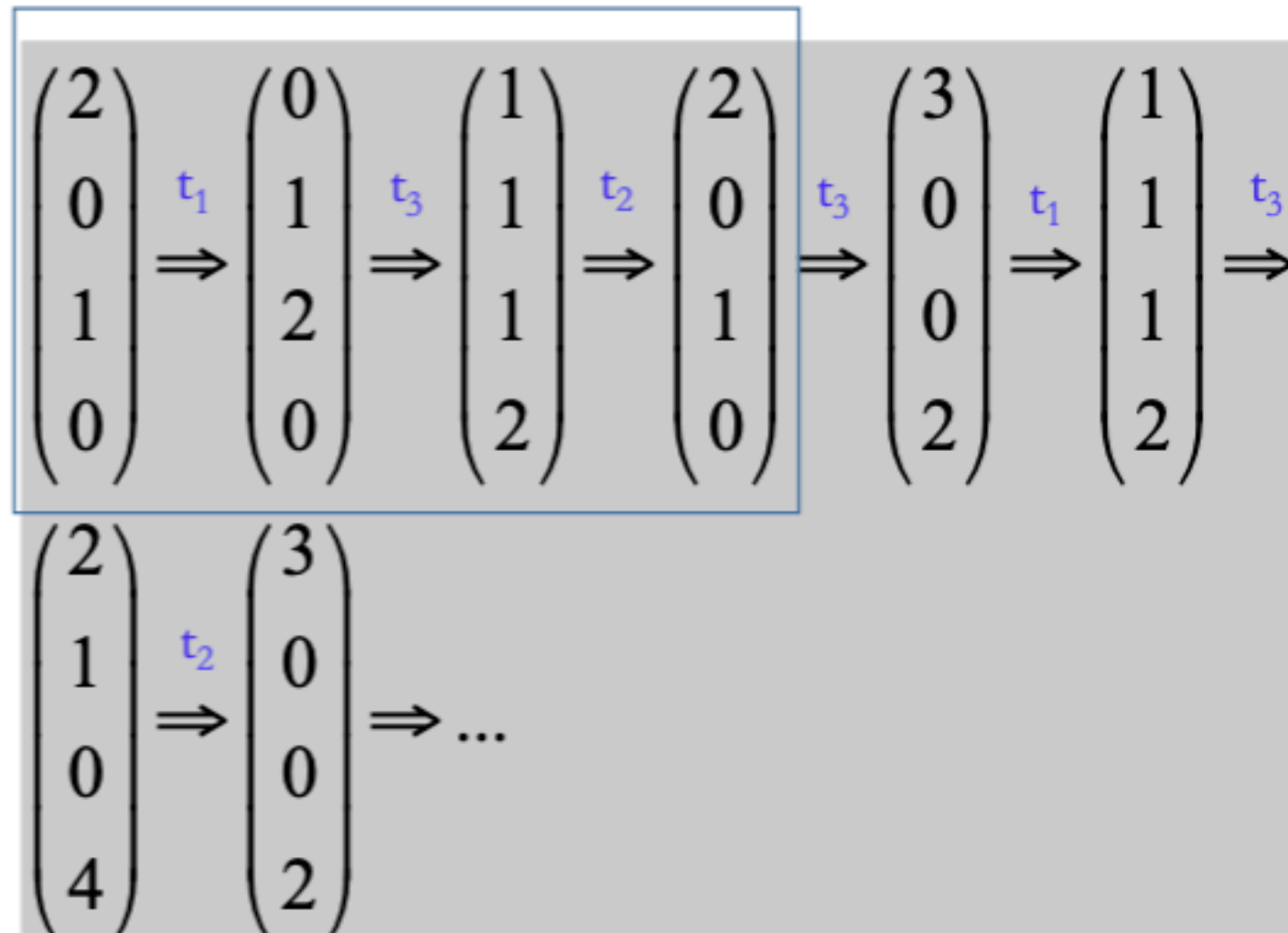
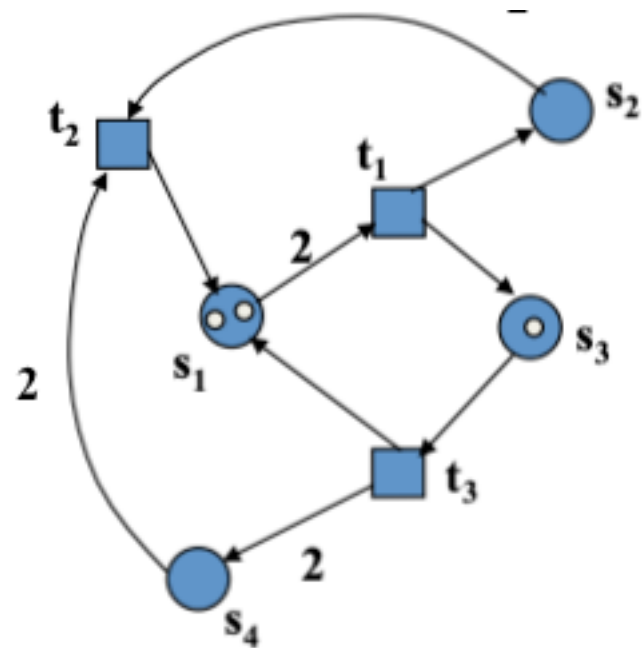
$$\therefore x = y = z$$



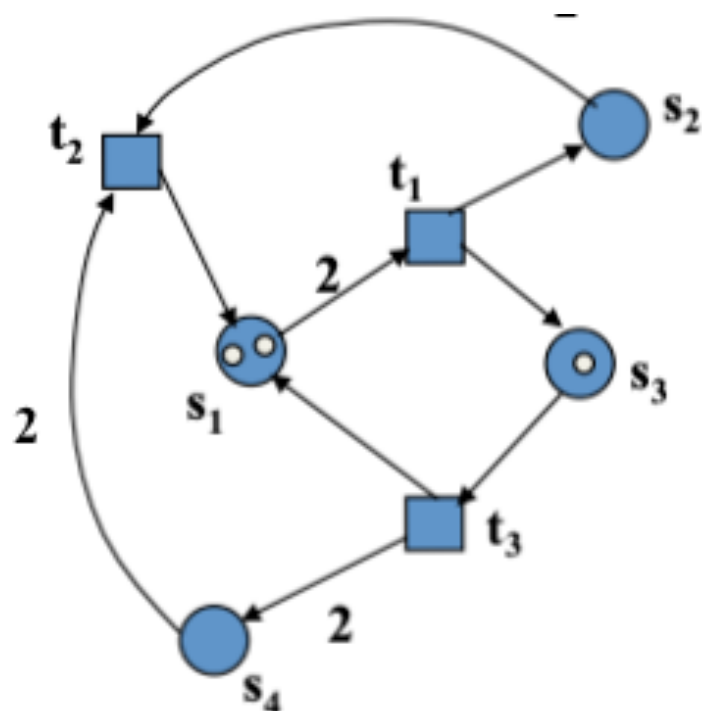
$$\begin{pmatrix} 2 \\ 0 \\ 1 \\ 0 \end{pmatrix} \xRightarrow{t_1} \begin{pmatrix} 0 \\ 1 \\ 2 \\ 0 \end{pmatrix} \xRightarrow{t_3} \begin{pmatrix} 1 \\ 1 \\ 1 \\ 2 \end{pmatrix} \xRightarrow{t_2} \begin{pmatrix} 2 \\ 0 \\ 1 \\ 0 \end{pmatrix} \xRightarrow{t_3} \begin{pmatrix} 3 \\ 0 \\ 0 \\ 2 \end{pmatrix} \xRightarrow{t_1} \begin{pmatrix} 1 \\ 1 \\ 1 \\ 2 \end{pmatrix} \xRightarrow{t_3} \dots$$

$$\begin{pmatrix} 2 \\ 1 \\ 0 \\ 4 \end{pmatrix} \xRightarrow{t_2} \begin{pmatrix} 3 \\ 0 \\ 0 \\ 2 \end{pmatrix} \Rightarrow \dots$$

Note que existem passos (string de transições independentes) que levam o sistema ao mesmo estado, isto é, denotam ciclos



O processo $t_1 t_3 t_2$ gera ciclos invariantes em estados diferentes, como mostrado anteriormente.



$$\sigma_1 = \begin{pmatrix} 1 \\ 0 \\ 0 \end{pmatrix}, \sigma_2 = \begin{pmatrix} 0 \\ 0 \\ 1 \end{pmatrix} \text{ e } \sigma_3 = \begin{pmatrix} 0 \\ 1 \\ 0 \end{pmatrix} \Rightarrow$$

$$\Rightarrow \sum_1^3 \sigma_i = \begin{pmatrix} 1 \\ 1 \\ 1 \end{pmatrix} \text{ é um T-invariante}$$

Methods to invariant calculus:

Mathematic linear programming (Simplex);

Gaus-Jordan method

Algoritmo para calcular invariantes de lugar

Tese de doutorado: Modelagem e Análise de Requisitos de Sistemas Automatizados Usando UML e Redes de Petri, Arianna Z. Olivera Salmon

Passo 1: Para obter os invariantes de lugar é preciso em primeiro lugar calcular a transposta da matriz de incidência (A^T).

Passo 2: Em seguida deve-se aplicar o método de Gauss- Jordan para transformar a matriz de incidência em uma matriz diagonal. Se o posto da matriz de incidência $r(A^T)$ for menor que o número de incógnitas (número de lugares da rede), então ir ao passo 3. Se o posto da matriz de incidência for igual ao número de incógnitas, então terminar, o sistema não tem invariantes de lugar.

Passo 3: Obter a solução básica do sistema representado pela matriz de incidência.

Algoritmo para calcular invariantes de transição

Tese de doutorado: Modelagem e Análise de Requisitos de Sistemas Automatizados Usando UML e Redes de Petri, Arianna Z. Olivera Salmon

Passo 1: Calcular a matriz de incidência do sistema modelado (A).

Passo 2: Aplicar o método de Gauss- Jordan para transformar a matriz de incidência em uma matriz diagonal. Se o posto da matriz de incidência $r(A)$ for menor que o número de incógnitas (número de lugares da rede), então ir ao passo 3. Se e o posto da matriz de incidência for igual ao número de incógnitas, então terminar, o sistema não tem invariantes de transição.

Passo 3: Obter a solução básica do sistema representado pela matriz de incidência.

Proposition 5.2.5. *A p -(t -)semiflow is minimal iff it is canonical and its support does not strictly contain the support of any other p -(t -)semiflow. Moreover, the set of minimal p -(t -)semiflows of a net is finite and unique.*

From the above result, the number of minimal p-(t-)semiflows is less than or equal to the number of incomparable vectors of dimension k ($k = |P|$ or $k = |T|$): Number of minimal p-(t-)semiflows $\leq \binom{k}{\lceil k/2 \rceil}$, where $\binom{\star}{\star}$ denotes binomial coefficient and $\lceil \star \rceil$ denotes rounding up to an integer. In practice the number of minimal p-(t-)semiflows is much smaller than the previously stated upper bound.

Algorithm 5.4 (Computation of the minimal p-semiflows)

Input - The incidence matrix C . A fixed but arbitrary order in P is supposed.

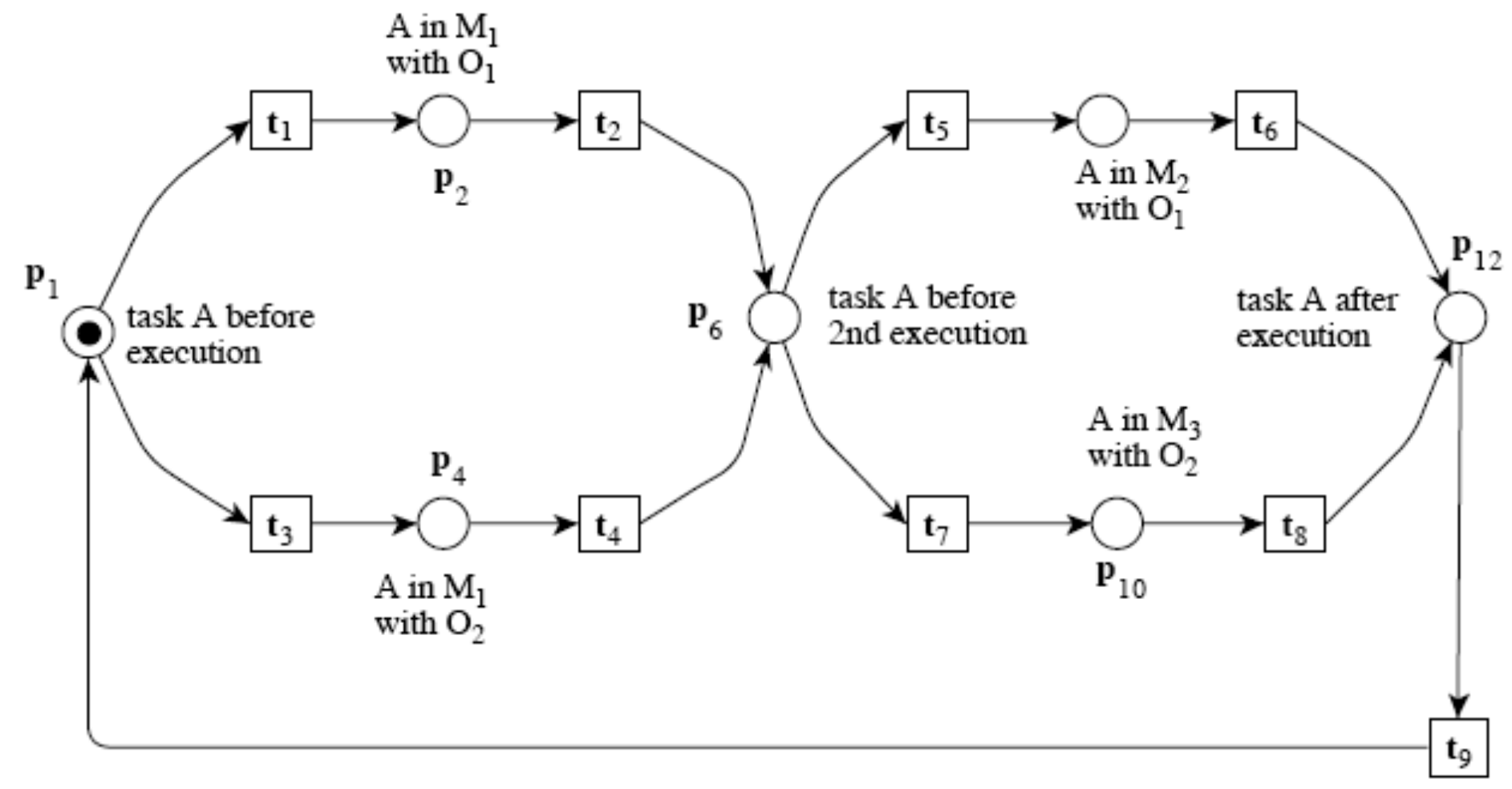
Output - The p-semiflow matrix Ψ , where each row is a minimal p-semiflow.

1. $A = C$; $\Psi = I_n$; { I_n is an identity matrix of dimension n }
 2. for $i = 1$ to m do { $m = |T|$ }
 - 2.1 Add to the matrix $[\Psi|A]$ all rows which are natural linear combinations of pairs of rows of $[\Psi|A]$ and which annul the i -th column of A
 - 2.2 Eliminate from $[\Psi|A]$ the rows in which the i -th column of A is non-null.
 3. Transform the rows of Ψ into canonical p-semiflows and remove all non-minimal p-semiflows from Ψ using the characterisation of proposition 5.2.5.
-

Procedimento de modelagem e análise

Requisitos do problema: *Seja um sistema de manufatura simples, composto de 3 máquinas DNC: M1, M2 e M3. Estas máquinas podem executar duas operações diferentes, O1 e O2, de modo que O1 pode ser executado nas máquinas M1 e M2 mas não simultaneamente. Similarmente, O2 pode ser executado em M1 e M3 mas não simultaneamente.*

Uma forma de tratar o problema é em primeiro lugar modelar a sequência de operações, sem levar em conta nenhuma restrição e nenhum recurso.



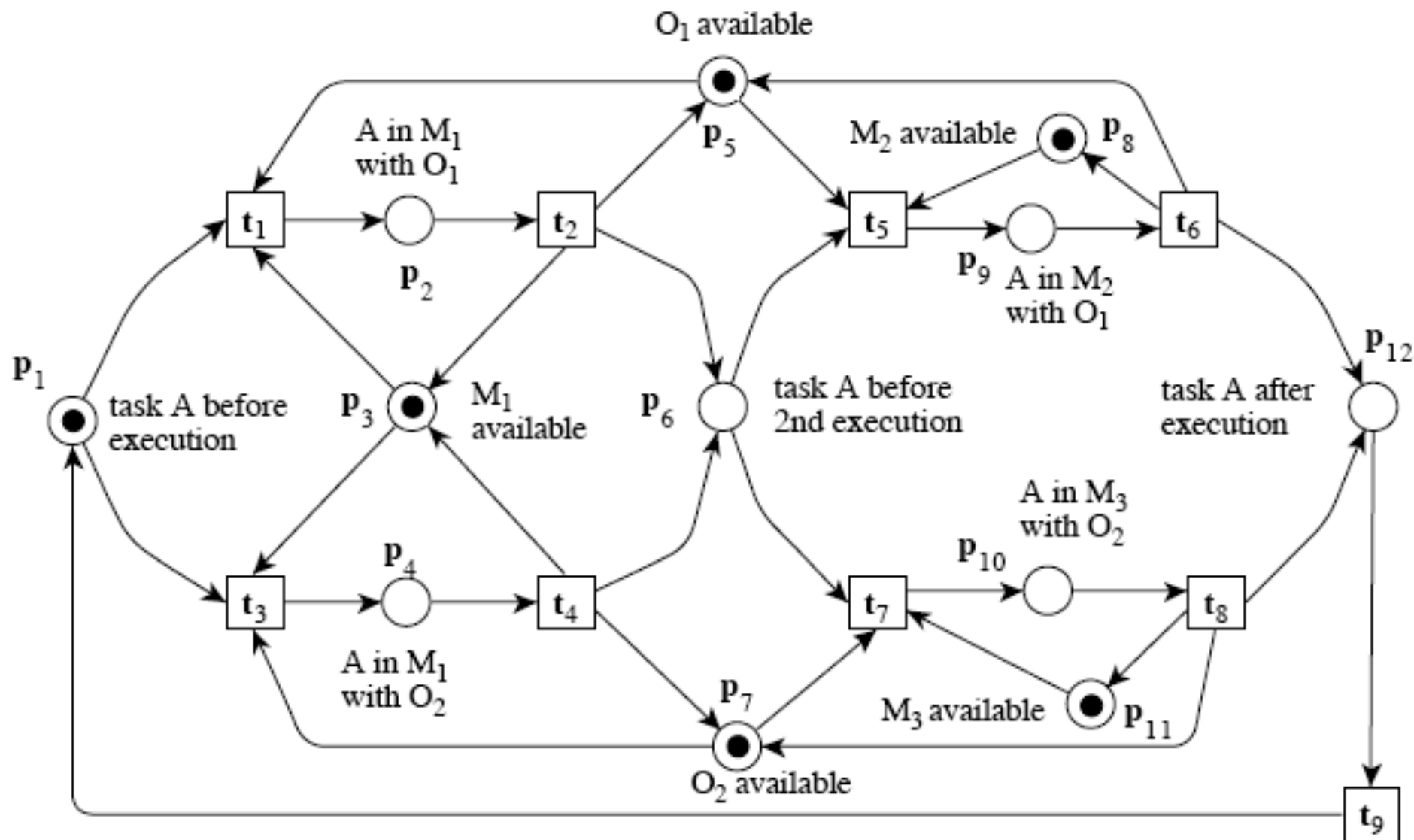
Girauld, C. and Valk, R.; Petri Nets for Systems Engineering, Springer-Verlag, 2003

Análise

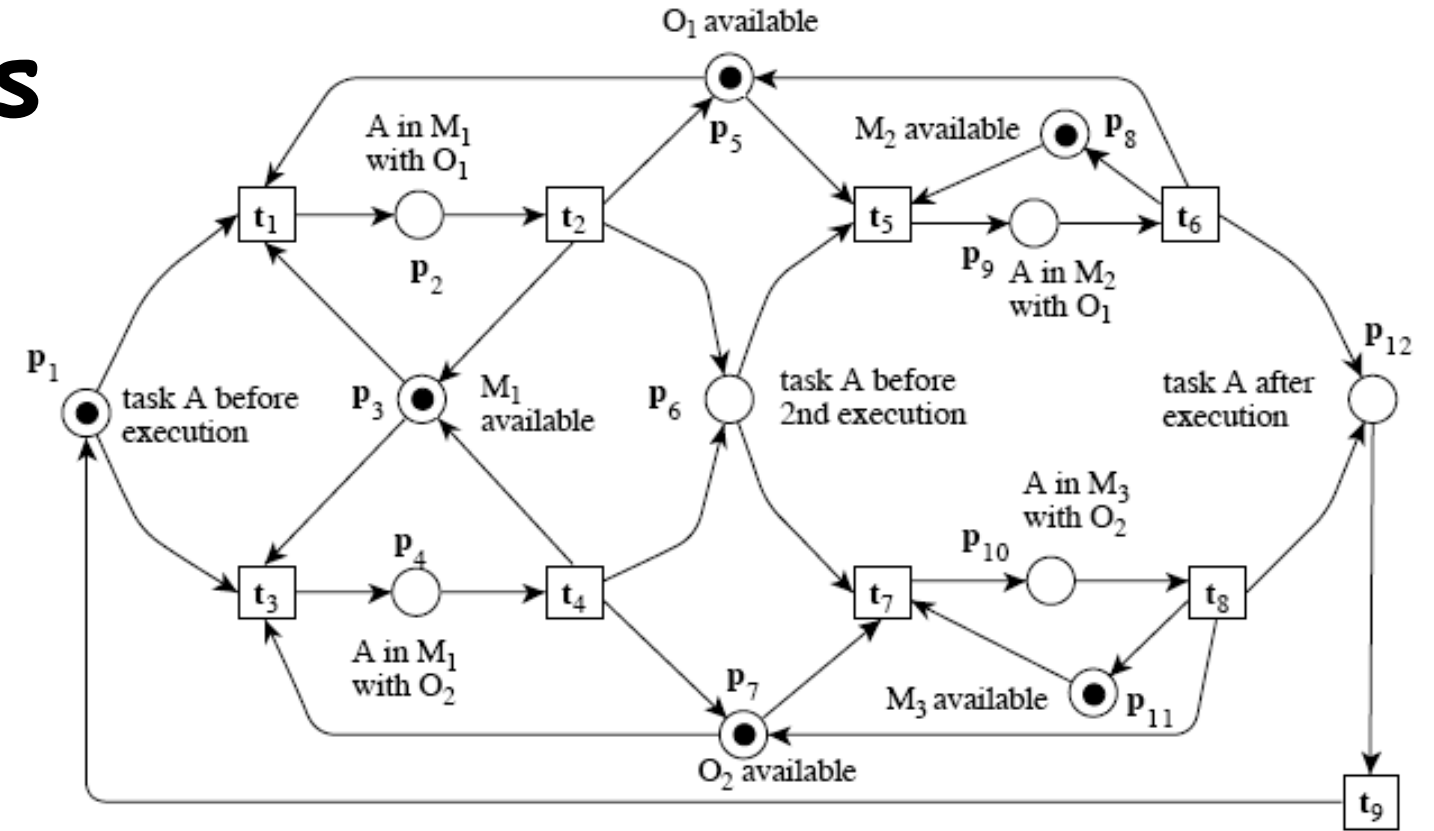
O sistema é cíclico, e permite a combinação das operações em qualquer ordem (e portanto o sistema estaria preparado para implementar qualquer receita de peça). O sistema é conservativo, de modo que cada peça seria representada por uma marca que deve estar em algum dos lugares já especificados.

Na especificação de requisitos, os recursos são representados pela disponibilidade das máquinas e pela sua capacidade de executar cada uma das operações. Uma vez feita a parte sequencial da rede devemos agora introduzir estas restrições, que devem alterar a rede ou a sucessão de estados desta.

Introduzindo os recursos, segundo a especificação de requisitos já apresentada, temos:



Análise de Invariantes

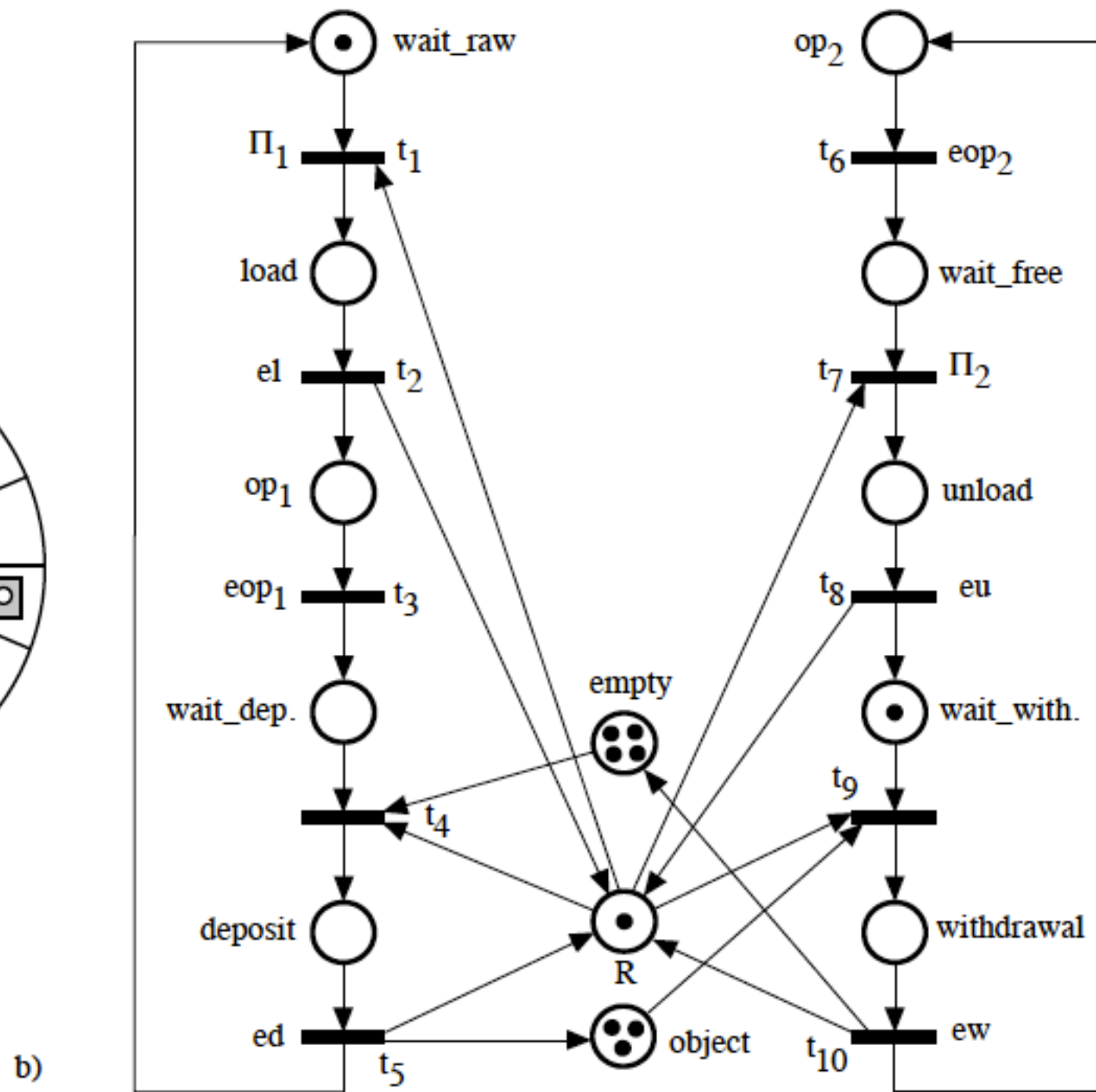
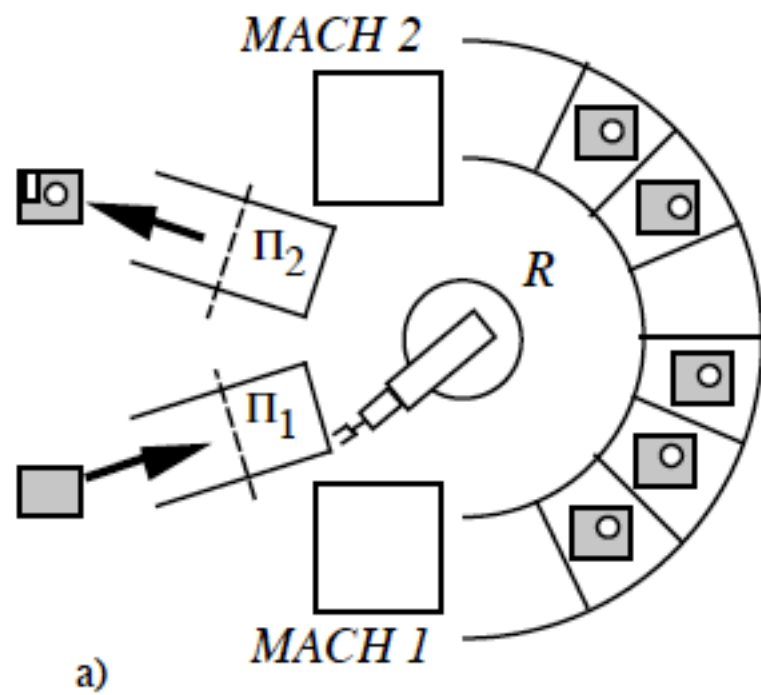


Análise de invariantes

Os invariantes podem ser analisados e servir como forma de verificação para o atendimento dos requisistos

- i) $m[p_2] + m[p_5] + m[p_9] = 1;$
- ii) $m[p_4] + m[p_7] + m[p_{10}] = 1;$
- iii) $m[p_2] + m[p_3] + m[p_4] = 1;$
- iv) $m[p_1] + m[p_2] + m[p_4] + m[p_6] + m[p_9] + m[p_{10}] + m[p_{12}] = c$

Example:



$$\begin{aligned}
 m[\text{wait_raw}] + m[\text{load}] + m[\text{op}_1] + m[\text{wait_dep.}] + m[\text{deposit}] &= 1 \\
 m[\text{op}_2] + m[\text{wait_free}] + m[\text{unload}] + m[\text{wait_with.}] + m[\text{withdrawal}] &= 1 \\
 m[\text{empty}] + m[\text{deposit}] + m[\text{object}] + m[\text{withdrawal}] &= 7 \\
 m[R] + m[\text{load}] + m[\text{unload}] + m[\text{deposit}] + m[\text{withdrawal}] &= 1
 \end{aligned}$$


Bounds: $\forall p_i \in P \setminus \{\text{empty}, \text{object}\} . (m[p_i] \leq 1 \quad \wedge \quad m[\text{empty}] \leq 7 \quad \wedge \quad m[\text{object}] \leq 7)$.

The places in each of the following sets are in marking mutual exclusion:

- a) $\{\text{wait_raw}, \text{load}, \text{op}_1, \text{wait_dep.}, \text{deposit}\}$
- b) $\{\text{op}_2, \text{wait_free}, \text{unload}, \text{wait_with.}, \text{withdrawal}\}$
- c) $\{R, \text{load}, \text{unload}, \text{deposit}, \text{withdrawal}\}$



A Formal Approach to Requirements Engineering of Automated Systems: Facing the Challenge for New Automated Systems

Arianna Z. Olivera Salmon¹ · Pedro M. G. del Foyo² · Jose Reinaldo Silva³ 

Received: 24 August 2020 / Revised: 23 March 2021 / Accepted: 23 April 2021 / Published online: 9 May 2021
 © Brazilian Society for Automatics–SBA 2021

Abstract

It is a consensus that intelligent manufacturing plants should be automated, especially in what concerns automated processes to Industry 4.0. This new manufacturing approach is based on multifunctional distributed systems that, for its turn, depend on a sound design requirements phase. Consequently, the requirement specifications became even more significant in the design of intelligent manufacturing systems, where the possibility to analyze requirements previously could save time and effort. Frequently, relevant requirements assumptions change during the engineering process due to emergent and volatile requirements—called “scope creep.” A revision of the requirements approach should be demanding to minimize this problem. The proposal presented in this work anticipates the formal representation of automated systems requirements to allow the proper analysis and validation. Also, since conventional production lines have been replaced by a product-service (production) systems (PSS), there is pressure to review the requirements phase, mainly to automated systems. To increase the accuracy, we propose a requirements cycle composed of modeling, analysis, and (formal) verification where formalization is anticipated by capturing visual modeled requirements. Alternative approaches to the functional method are also explored, considering the Goal-oriented Requirements Engineering approach, the best fit to PSS. A formal process to treat requirements in a functional approach, represented by UML diagrams, is transferred to Petri Nets and submitted to formal analysis. Traditional translation from UML to Petri Nets is replaced by an object-oriented match to Petri Nets extensions covered by the standard ISO/IEC 15.909. A comparison between this approach and the goal-oriented raises a discussion about the requirements efficiency dilemma, where functionality could be pointed as the main reason to scope creep. Therefore, a strictly functional approach is proposed with a new transference algorithm, and the results are used to support the discussion about using goal-oriented alternatives. A case study of the chemical industry illustrates the practical use of the proposal.

Keywords Intelligent manufacturing · Requirements engineering · Manufacturing modeling and design · Petri Nets

1 Introduction

The demand for complex and sophisticated automated systems has been growing since the end of the last century pushed by digital convergence, which leads to intensive use of

information to manage the control of multifunctional systems (Silva and Nof, 2015). Such demand requires more accuracy in the systems design process to fit users’ and stakeholders’ expectations in a short—but still accurate—cycle. The requirements phase is pointed as the focus in that discussion (Sanghera, 2019; Johanson et al., 2016).

In 2017, the NASA Formal Methods conference (NASA, 2017) pointed to a demand for sound design processes, including formal requirements analysis. That raised a discussion about which suitable language presentations (or ontology) would better express requirements. Practical methods used by the software industry are based on semi-formal languages or even in natural languages. While such methods can speed up elicitation, translation to formal presentations becomes very slow or entirely impossible. Semi-formal presentations as UML rely on visual diagrams, which were

✉ Jose Reinaldo Silva
 rcinaldo@usp.br

Arianna Z. Olivera Salmon
 azolivera@gmail.com

¹ TrueChange Co., Rua Dona Maria Cesar, 170, Recife, PE CEP 50030-140, Brazil

² Mechanical Engineering Department, UFPE, Av. Acadêmico Hélio Ramos s/n, Recife, PE, Brazil

³ Mechatronics Engineering Department, Universidade de São Paulo, São Paulo, SP, Brazil

Formal methods have been successfully used for the development of safety-critical systems; however, the need for skilled knowledge when writing formal models and the reasoning about them represents a major barrier in the adoption of formal methodologies for the development of non-critical applications. A key aspect in the verification of formal models and in the development of reliable systems is the identification of invariants. However, finding correct and meaningful invariants for a model represents a significant challenge. We have used automated theory formation (ATF) techniques to automatically discover invariants of Event-B models. In particular, we use Colton's HR system to explore the domain of Event-B and suggest potential invariants.

The invariant method

We first *construct* a set of place invariants.

Then we check whether they are *fulfilled*.

- This is done by showing that each occurring binding element *respects* the invariants.
- The *removed* set of tokens must be identical to the *added* set of tokens – when the weights are taken into account.

Finally, we use the place invariants to *prove* behavioural properties of the CP-net.

- This is done by a *mathematical proof*.



Kurt Jensen
Univ. of Ahrus, Demark

Automating the invariant analysis

Automatic calculation of all place invariants:

- This is possible, but it is a very *complex* task.
- Moreover, it is difficult to represent the results on a *useful form*, i.e., a form which can be used by the system designer.

Interactive calculation of place invariants:

- The *user* proposes some of the weights.
- The *tool* calculates the *remaining weights* – if possible.

Interactive calculation of place invariants is *much easier* than a fully automatic calculation.

The invariant method in CPN

- The user needs some ingenuity to *construct* invariants. This can be supported by *computer tools* – interactive process.
- The user also needs some ingenuity to *use* invariants. This can also be supported by *computer tools* – interactive process.
- Invariants can be used to verify a system – without fixing the *system parameters* (such as the number of sites in the data base system).

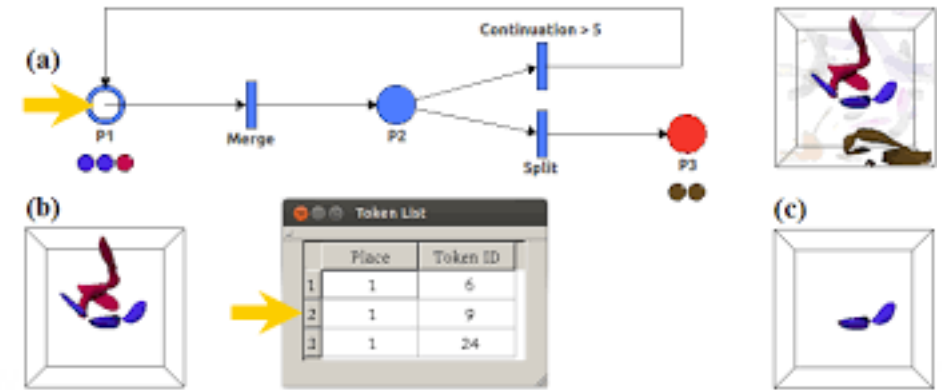
Sobre o artigo final:



**ARTIGO
ACADÊMICO**

Teremos que focar no artigo final porque estamos a 30 dias do final do curso. Portanto na próxima aula teremos mais um milestone, onde agora vocês devem apresentar além do título, abstract, palavras-chave, uma introdução e uma seção de "background", além de bibliografia.

Na aula que vem trataremos do problema da análise de invariáveis nas redes coloridas e voltaremos à discussão sobre o uso da análise de propriedades na modelagem e análise de sistemas automatizados.



seja qual for a opção para análise de invariantes (em redes clássicas ou coloridas) o nosso objetivo é nos certificar de que o ambiente de modelagem segue o formalismo, e, se for o caso usá-lo para determinar os invariantes.

