

Petri Nets

Peter Marwedel
TU Dortmund,
Informatik 12

2009/11/09



Models of computation considered in this course

Communication/ local computations	Shared memory	Message passing	
		Synchronous	Asynchronous
Undefined components		Plain text, use cases (Message) sequence charts	
Communicating finite state machines	StateCharts		SDL
Data flow	(Not useful)		Kahn networks, SDF
Petri nets		C/E nets, P/T nets, ...	
Discrete event (DE) model	VHDL, Verilog, SystemC, ...	Only experimental systems, e.g. distributed DE in Ptolemy	
Von Neumann model	C, C++, Java	CSP, ADA	C, C++, Java with libraries

Introduction

Introduced in 1962 by Carl Adam Petri in his PhD thesis.
Focus on modeling causal dependencies;
no global synchronization assumed (message passing only).

Key elements:

- **Conditions**

Either met or no met.

- **Events**

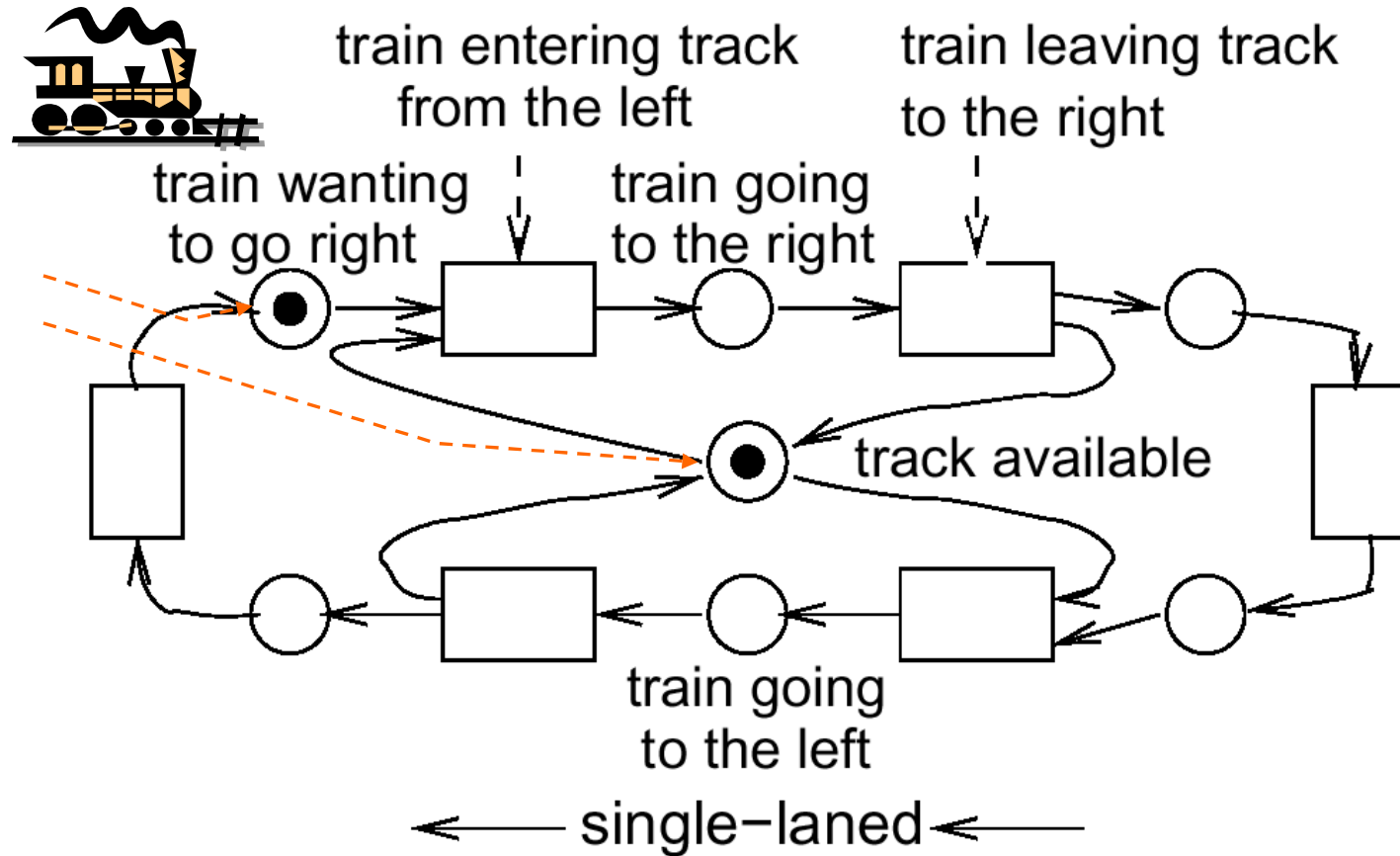
May take place if certain conditions are met.

- **Flow relation**

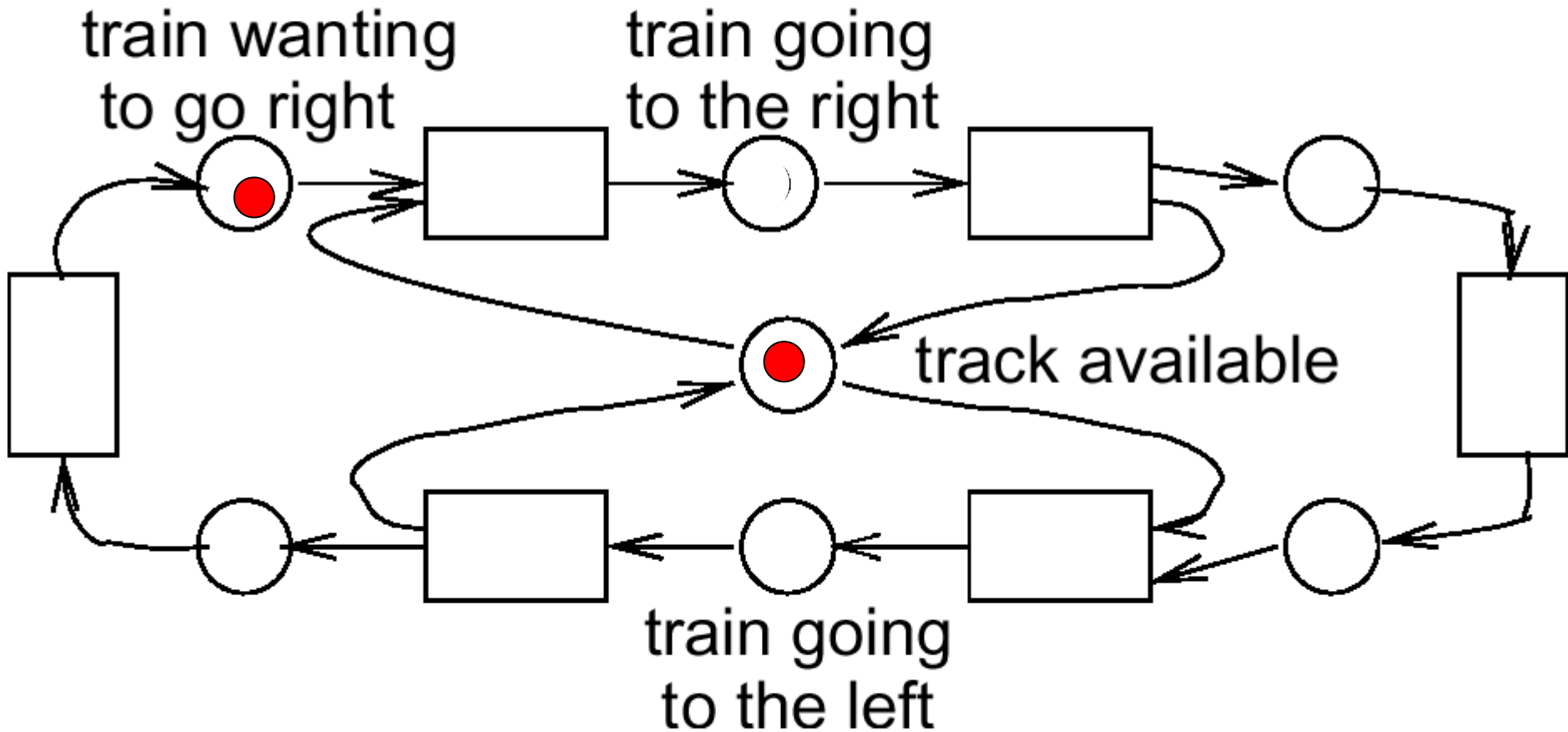
Relates conditions and events.

Conditions, events and the flow relation form
a **bipartite graph** (graph with two kinds of nodes).

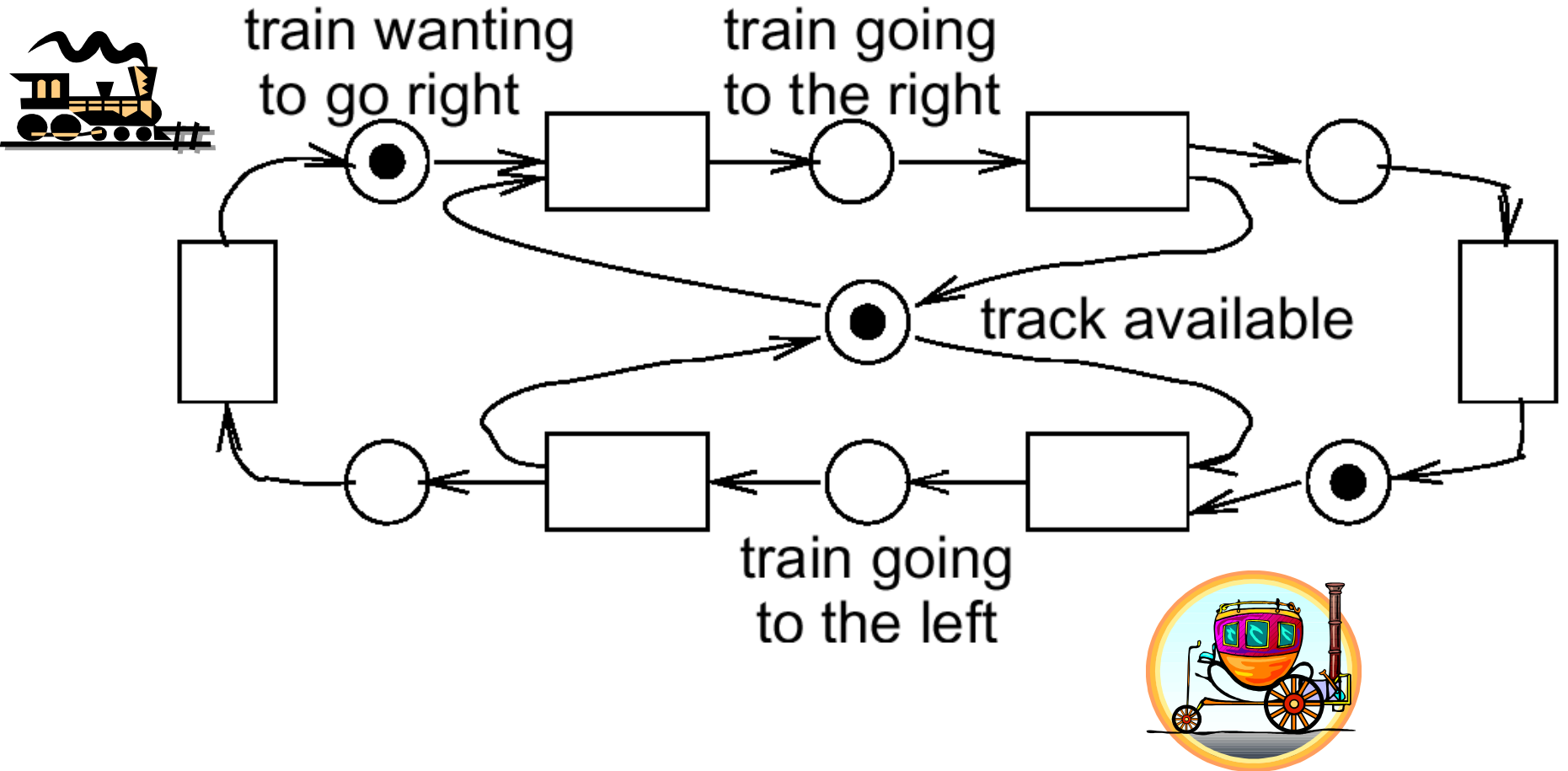
Example: Synchronization at single track rail segment



Playing the “token game“



Conflict for resource "track"



More complex example (1)

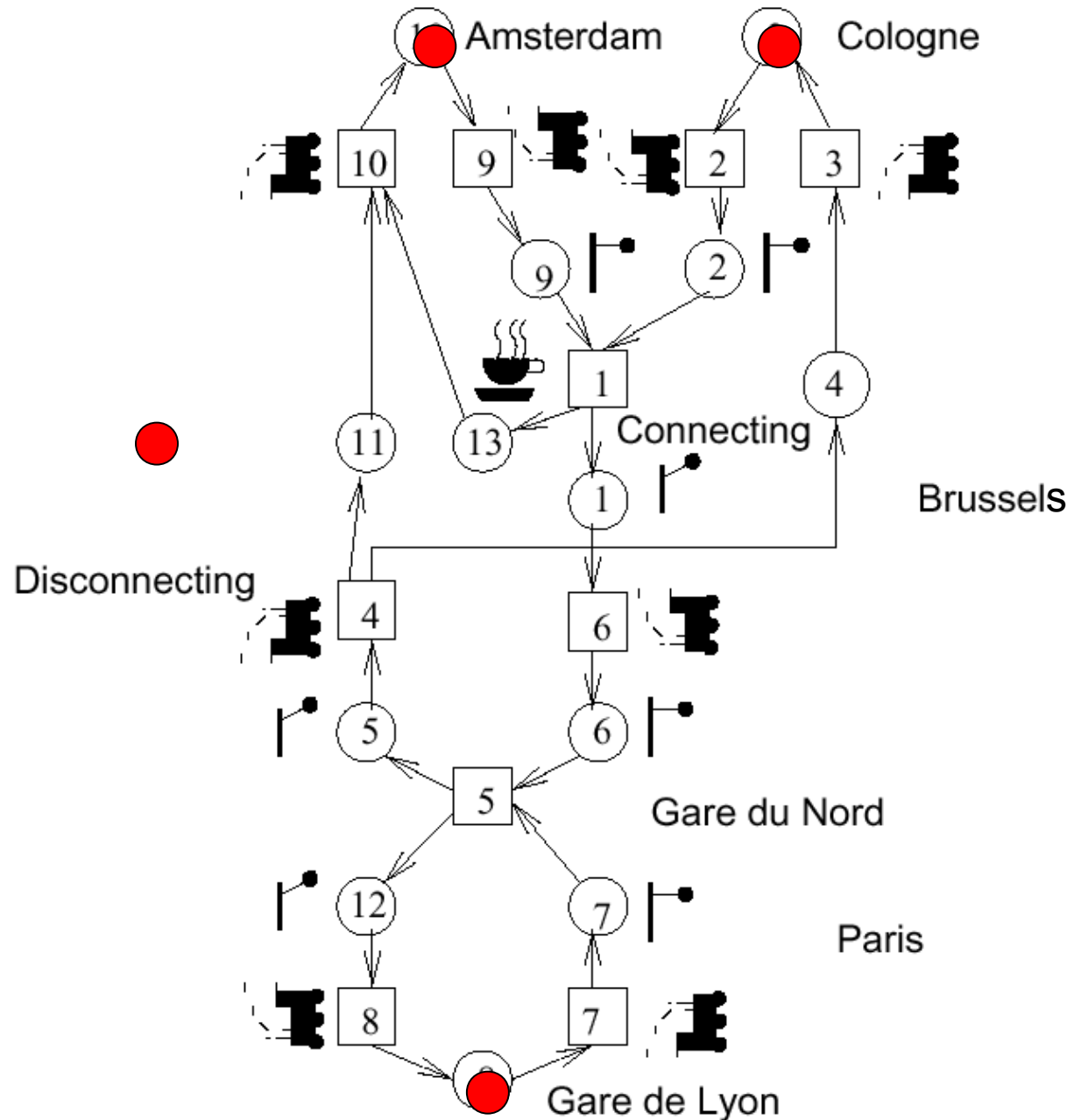
Thalys trains between
Cologne, Amsterdam,
Brussels and Paris.



[<http://www.thalys.com/be/en>]

More complex example (2)

Slightly simplified:
Synchronization
at Brussels and
Paris,
using stations
“Gare du Nord”
and “Gare de
Lyon” at Paris



Condition/event nets

Def.: $N=(C,E,F)$ is called a **net**, iff the following holds

1. C and E are disjoint sets
2. $F \subseteq (C \times E) \cup (E \times C)$; is binary relation, ("**flow relation**")

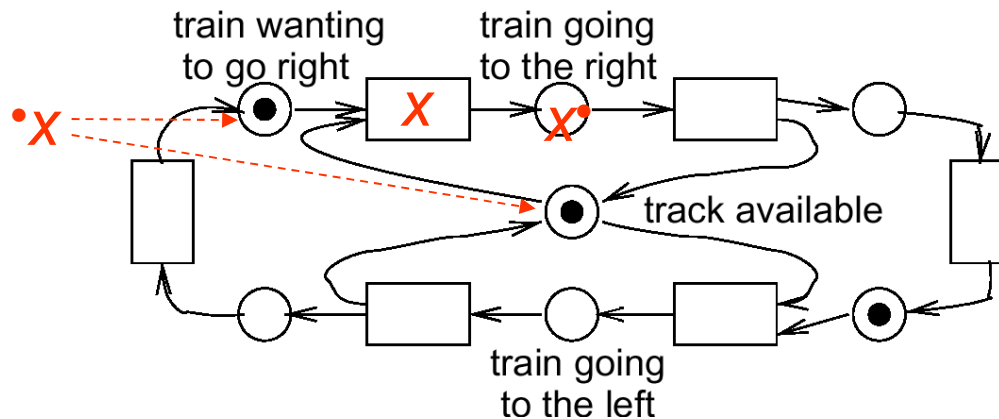
Pre- and post-sets

Def.: Let N be a net and let $x \in (C \cup E)$.

$\bullet x := \{y \mid y F x\}$ is called the **pre-set** of x ,
(or **preconditions** if $x \in E$)

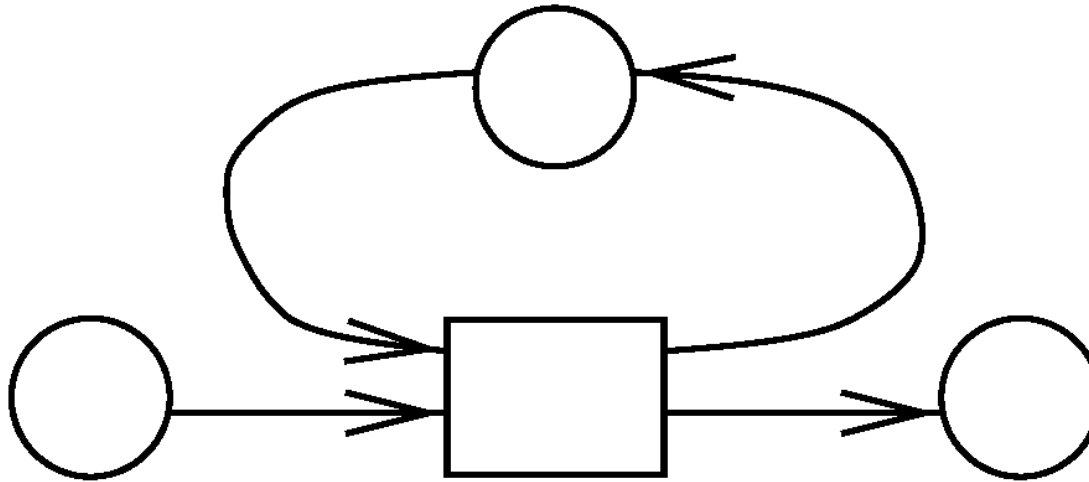
$x^\bullet := \{y \mid x F y\}$ is called the set of **post-set** of x ,
(or **postconditions** if $x \in E$)

Example:



Loops and pure nets

Def.: Let $(c,e) \in C \times E$. (c,e) is called a **loop** iff $cFe \wedge eFc$.

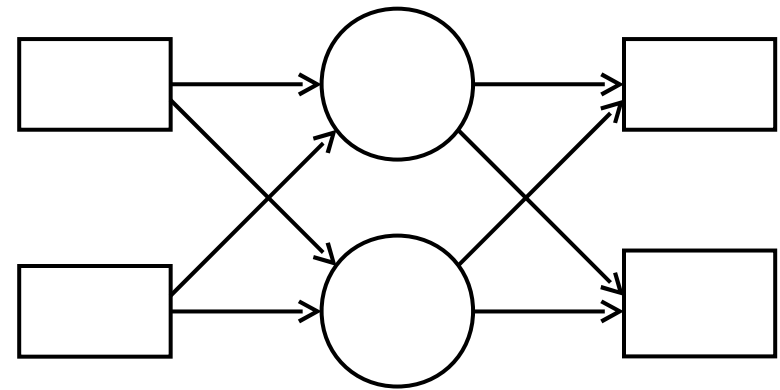
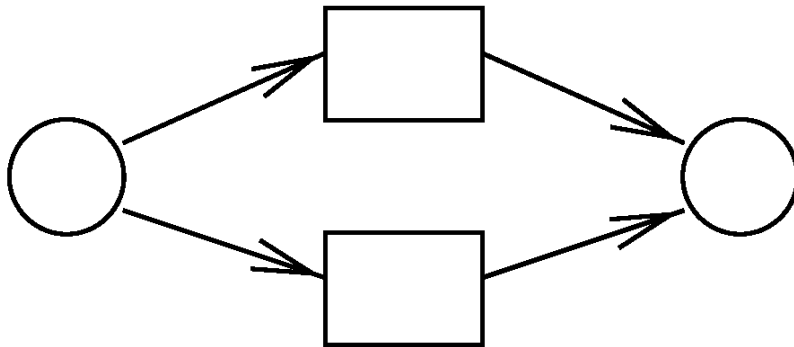


Def.: Net $N=(C,E,F)$ is called **pure**, if F does not contain any loops.

Simple nets

Def.: A net is called **simple** if no two nodes $n1$ and $n2$ have the same pre-set and post-set.

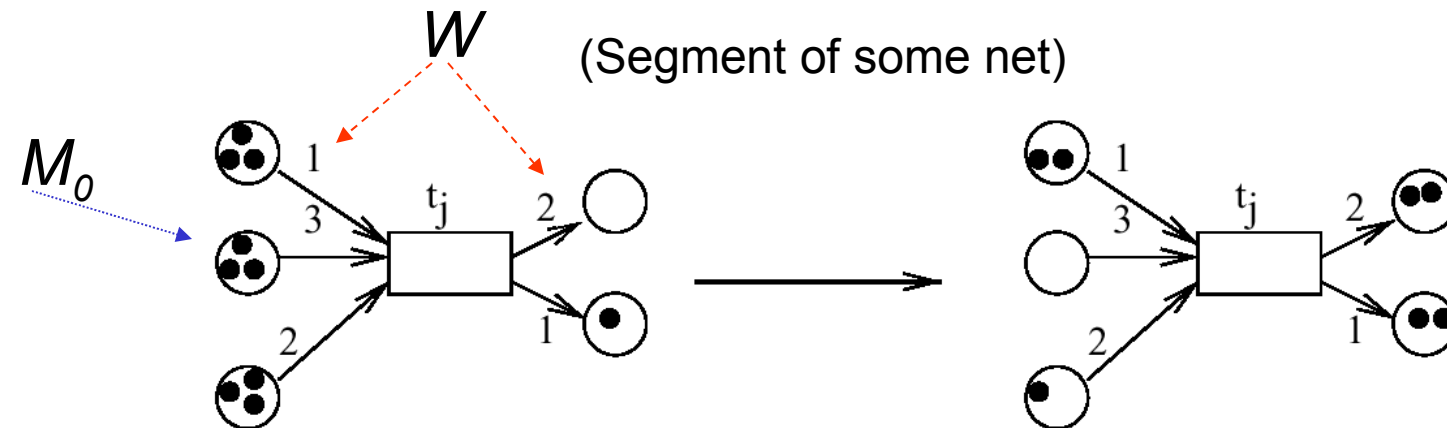
Example (not simple nets):



Def.: Simple nets with no isolated elements meeting some additional restrictions are called **condition/event nets (C/E nets)**.

Place/transition nets

- Def.:** (P, T, F, K, W, M_0) is called a **place/transition net** iff
1. $N=(P, T, F)$ is a **net** with places $p \in P$ and transitions $t \in T$
 2. $K: P \rightarrow (\mathbb{N}_0 \cup \{\omega\}) \setminus \{0\}$ denotes the **capacity** of places
(ω symbolizes infinite capacity)
 3. $W: F \rightarrow (\mathbb{N}_0 \setminus \{0\})$ denotes the **weight** of graph edges
 4. $M_0: P \rightarrow \mathbb{N}_0 \cup \{\omega\}$ represents the **initial marking** of places

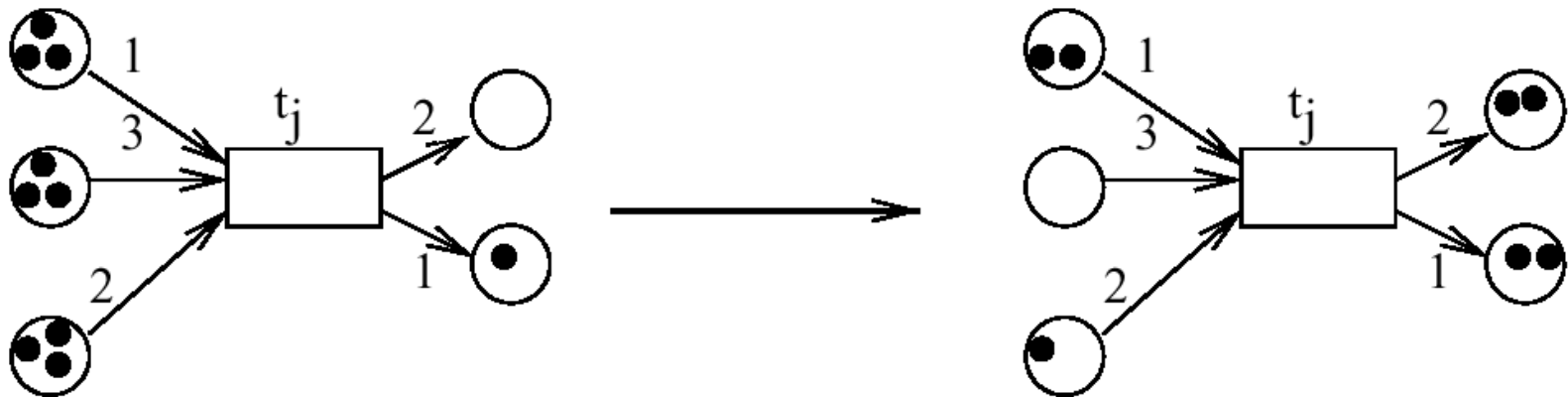


defaults:
 $K = \omega$
 $W = 1$

Computing changes of markings

“Firing” transitions t generate new markings on each of the places p according to the following rules:

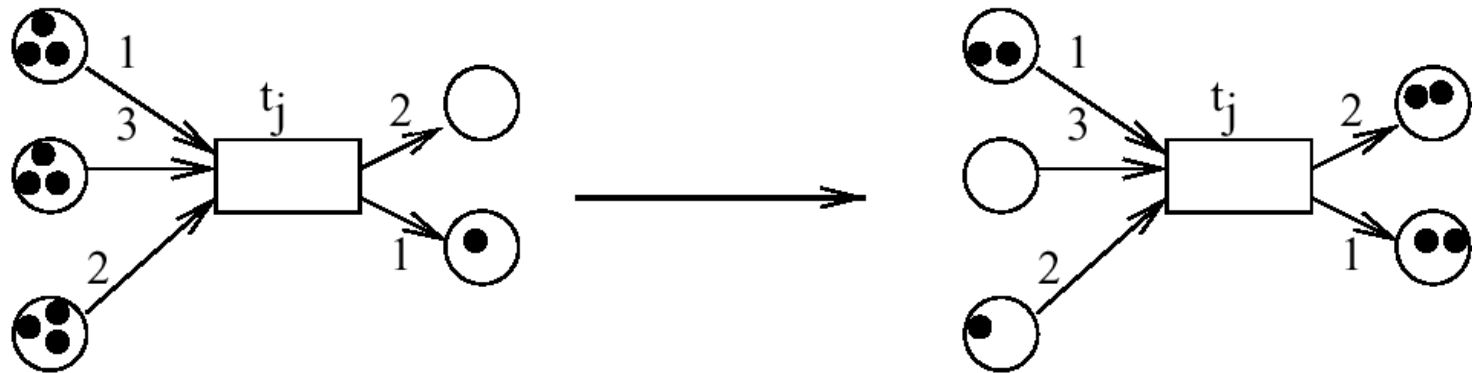
$$M'(p) = \begin{cases} M(p) - W(p, t), & \text{if } p \in \bullet t \setminus t^\bullet \\ M(p) + W(t, p), & \text{if } p \in t^\bullet \setminus \bullet t \\ M(p) - W(p, t) + W(t, p), & \text{if } p \in \bullet t \cap t^\bullet \\ M(p) & \text{otherwise} \end{cases}$$



Activated transitions

Transition t is “activated“ iff

$$(\forall p \in \bullet t : M(p) \geq W(p,t)) \wedge (\forall p \in t^\bullet : M(p) + W(t,p) \leq K(p))$$



Activated transitions can “take place“ or “fire“,
but don't have to.

We never talk about “time“ in the context of Petri nets.

The order in which activated transitions fire, is not fixed
(it is non-deterministic).

Shorthand for changes of markings

Slide 12:

$$M'(p) = \begin{cases} M(p) - W(p, t), & \text{if } p \in \bullet t \setminus t^\bullet \\ M(p) + W(t, p), & \text{if } p \in t^\bullet \setminus \bullet t \\ M(p) - W(p, t) + W(t, p), & \text{if } p \in \bullet t \cap t^\bullet \\ M(p) & \text{otherwise} \end{cases}$$

Let

$$\underline{t}(p) = \begin{cases} -W(p, t) & \text{if } p \in \bullet t \setminus t^\bullet \\ +W(t, p) & \text{if } p \in t^\bullet \setminus \bullet t \\ -W(p, t) + W(t, p) & \text{if } p \in \bullet t \cap t^\bullet \\ 0 & \text{otherwise} \end{cases}$$

$$\Rightarrow \quad \forall p \in P: M'(p) = M(p) + \underline{t}(p)$$

$$\Rightarrow \quad M' = M + \underline{t} \quad \text{+ : vector add}$$

Matrix \underline{N} describing all changes of markings

$$\underline{t}(p) = \begin{cases} -W(p,t) & \text{if } p \in {}^{\bullet}t \setminus t^{\bullet} \\ +W(t,p) & \text{if } p \in t^{\bullet} \setminus {}^{\bullet}t \\ -W(p,t) + W(t,p) & \text{if } p \in t^{\bullet} \cap {}^{\bullet}t \\ 0 & \end{cases}$$

Def.: Matrix \underline{N} of net N is a mapping

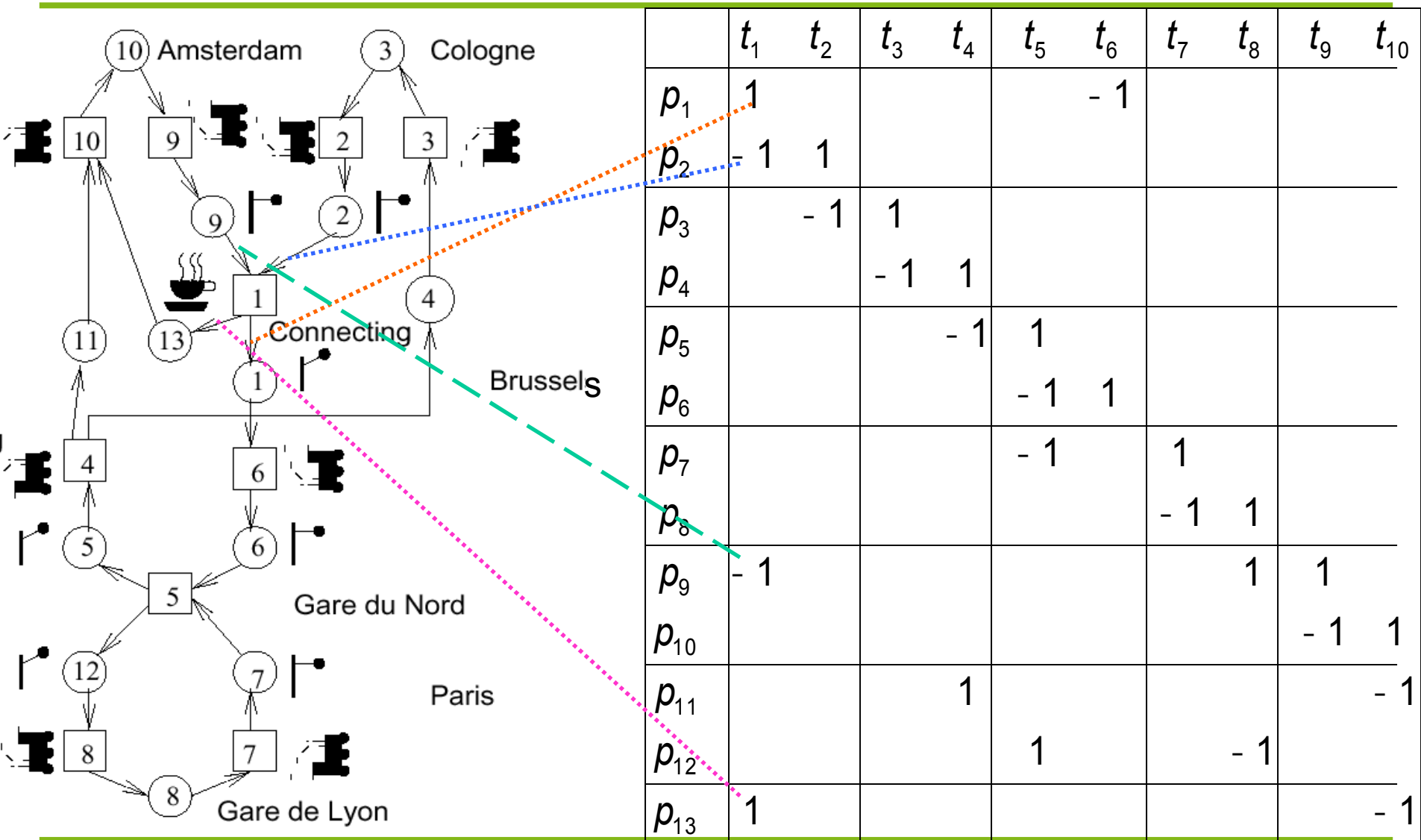
$$\underline{N}: P \times T \rightarrow \mathbb{Z} \text{ (integers)}$$

such that $\forall t \in T: \underline{N}(p,t) = \underline{t}(p)$

Component in column t and row p indicates the change of the marking of place p if transition t takes place.

For pure nets, (\underline{N}, M_0) is a complete representation of a net.

Example: $N =$



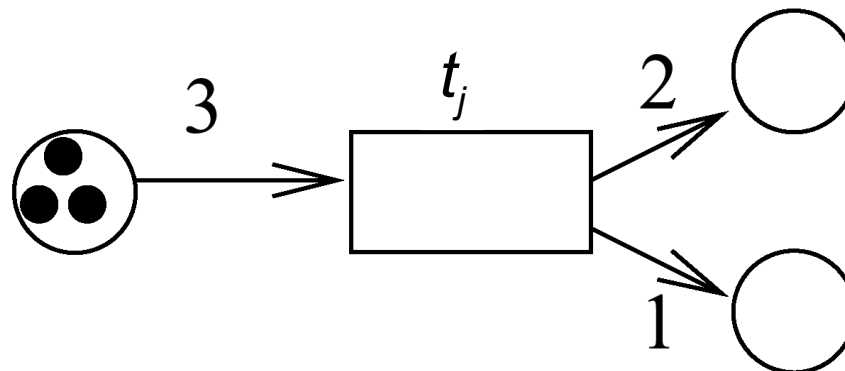
Place - invariants

Standardized technique for proving properties of system models

For any transition $t_j \in T$ we are looking for sets $R \subseteq P$ of places for which the accumulated marking is constant:

$$\sum_{p \in R} t_j(p) = 0$$

Example:



Characteristic Vector

$$\sum_{p \in R} t_{-j}(p) = 0$$

Let:
$$\underline{c}_R(p) = \begin{cases} 1 & \text{if } p \in R \\ 0 & \text{if } p \notin R \end{cases}$$

$$\Rightarrow 0 = \sum_{p \in R} t_{-j}(p) = \sum_{p \in P} t_{-j}(p) \underline{c}_R(p) = t_{-j} \cdot \underline{c}_R$$

Scalar product

Condition for place invariants

$$\sum_{p \in R} \underline{t}_j(p) = \sum_{p \in P} \underline{t}_j(p) \underline{c}_R(p) = \underline{t}_j \cdot \underline{c}_R = 0$$

Accumulated marking constant for **all** transitions if

$$\begin{array}{rcl} \underline{t}_1 \cdot \underline{c}_R & = & 0 \\ & \dots & \dots \\ \underline{t}_n \cdot \underline{c}_R & = & 0 \end{array}$$

Equivalent to $\underline{N}^T \underline{c}_R = \mathbf{0}$ where \underline{N}^T is the transposed of \underline{N}

More detailed view of computations

$$\begin{pmatrix} t_1(p_1) \dots t_1(p_n) \\ t_2(p_1) \dots t_2(p_n) \\ \dots \\ t_m(p_1) \dots t_m(p_n) \end{pmatrix} \begin{pmatrix} c_R(p_1) \\ c_R(p_2) \\ \dots \\ c_R(p_n) \end{pmatrix} = \begin{pmatrix} 0 \\ 0 \\ 0 \\ 0 \end{pmatrix}$$

System of linear equations.

Solution vectors must consist of zeros and ones.

Equations with multiple unknowns that must be integers called **diophantic** (👉 Greek mathematician Diophantos, ~300 B.C.).

Diophantic linear equation system more complex to solve than standard system of linear equations (actually NP-hard))

Different techniques for solving equation system (manual, ..)

Application to Thalys example

$$\underline{N}^T \underline{c}_R = \mathbf{0}, \text{ with } \underline{N}^T =$$

	s_1	s_2	s_3	s_4	s_5	s_6	s_7	s_8	s_9	s_{10}	s_{11}	s_{12}	s_{13}
t_1	1	-1							-1				1
t_2		1	-1										
t_3			1	-1									
t_4				1	-1						1		
t_5					1	-1	-1					1	
t_6	-1					1							
t_7							1	-1					
t_8								1				-1	
t_9									1	-1			
t_{10}										1	-1		-1

Solutions? Educated guessing:

$\sum_{\text{rows}} = 0 \rightarrow 1$ linear dependency among rows \rightarrow rank = $10 - 1 = 9$

Dimension of solution space = $13 - \text{rank} = 4$

4 components of (6, 11, 12, 13) of \underline{c}_R are independent

\rightarrow set one of these to 1

and others to 0 to obtain a basis for the solution space

1st basis

Set one of components (6, 11, 12, 13) to 1, others to 0.

→ **1st basis** b_1 :

$$b_1(s_6)=1, b_1(s_{11})=0,$$

$$b_1(s_{12})=0, b_1(s_{13})=0$$

	s_1	s_2	s_3	s_4	s_5	s_6	s_7	s_8	s_9	s_{10}	s_{11}	s_{12}	s_{13}
t_1	1	-1							-1				1
t_2		1	-1										
t_3			1	-1									
t_4				1	-1						1		
t_5					1	-1	-1					1	
t_6	-1					1							
t_7							1	-1					
t_8								1				-1	
t_9									1	-1			
t_{10}										1	-1		-1

$$\bullet t_{10}(s_{10}) b_1(s_{10}) + t_{10}(s_{11}) b_1(s_{11}) + t_{10}(s_{13}) b_1(s_{13}) = 0$$

$$\rightarrow b_1(s_{10}) = 0$$

$$\bullet t_9(s_9) b_1(s_9) + t_9(s_{10}) b_1(s_{10}) = 0$$

$$\rightarrow b_1(s_9) = 0$$

$$\bullet b_1 \equiv (1, 1, 1, 1, 1, 1, 0, 0, 0, 0, 0, 0, 0, 0)$$

All components $\in \{0, 1\}$

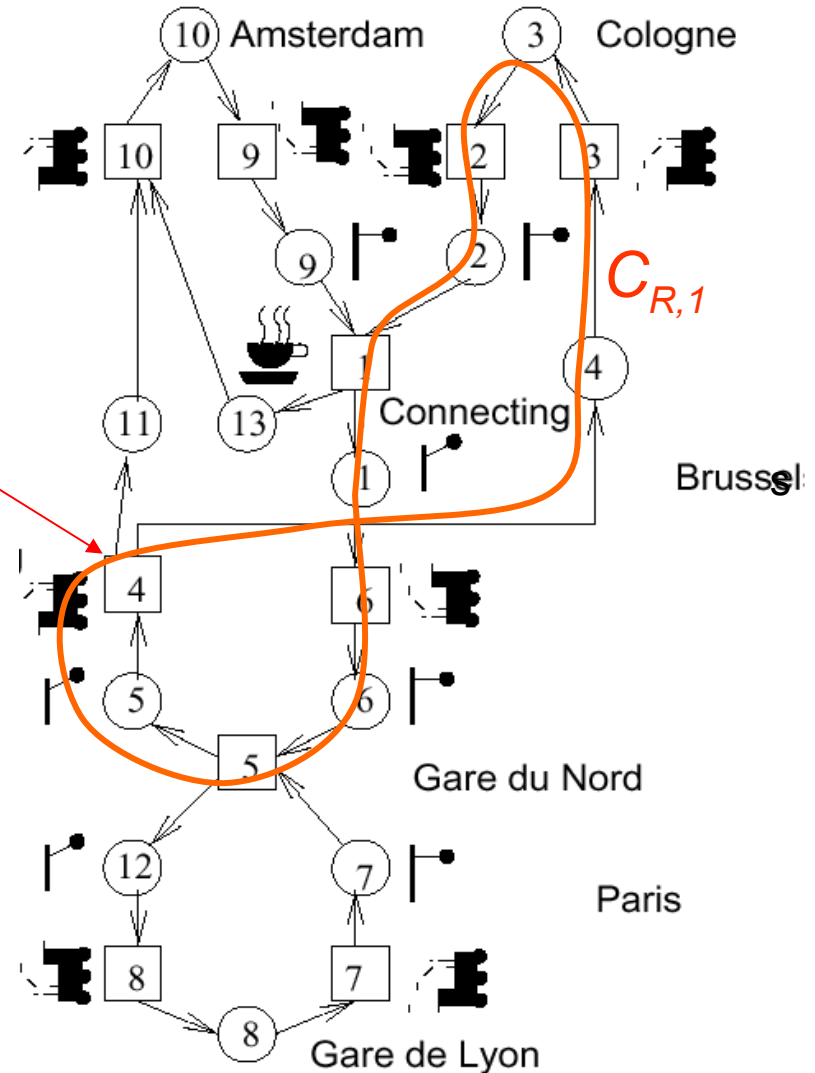
$$\rightarrow \mathbf{c}_{R1} = b_1$$

Interpretation of the 1st invariant

$$C_{R,1} = (11111100000000)$$

Characteristic vector describes places for Cologne train.

We proved that: the number of trains along the path remains constant.



2nd basis

Set one of components
(6, 11, 12, 13)
to 1, others to 0.

→ 2nd basis b_2 :

$$b_2(s_6)=0, b_2(s_{11})=1,$$

$$b_2(s_{12})=0, b_2(s_{13})=0$$

	s_1	s_2	s_3	s_4	s_5	s_6	s_7	s_8	s_9	s_{10}	s_{11}	s_{12}	s_{13}
t_1	1	-1							-1				1
t_2		1	-1										
t_3			1	-1									
t_4				1	-1						1		
t_5					1	-1	-1					1	
t_6	-1					1							
t_7							1	-1					
t_8								1				-1	
t_9									1	-1			
t_{10}										1	-1		-1

$$\bullet t_{10}(s_{10}) b_2(s_{10}) + t_{10}(s_{11}) b_2(s_{11}) + t_{10}(s_{13}) b_2(s_{13}) = 0$$

$$\rightarrow b_2(s_{10}) = 1$$

$$\bullet t_9(s_9) b_2(s_9) + t_9(s_{10}) b_2(s_{10}) = 0$$

$$\rightarrow b_2(s_9) = 1$$

$$\bullet b_2 = (0, -1, -1, -1, 0, 0, 0, 0, 1, 1, 1, 0, 0)$$

$$b_1 = (1, 1, 1, 1, 1, 1, 0, 0, 0, 0, 0, 0, 0)$$

b_2 not a characteristic
vector, but $\mathbf{c}_{R,2} = b_1 + b_2$ is

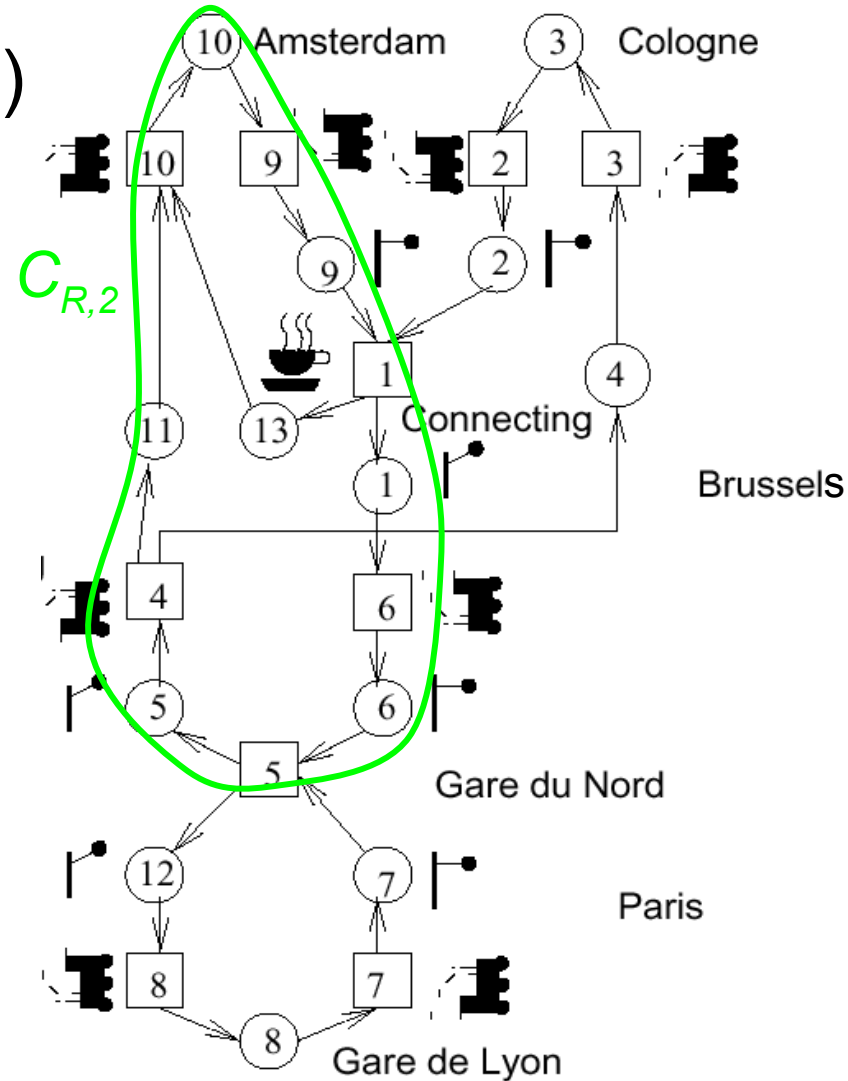
$$\rightarrow \mathbf{c}_{R,2} =$$

$$(1, 0, 0, 0, 1, 1, 0, 0, 1, 1, 1, 0, 0)$$

Interpretation of the 2nd invariant

$$C_{R,2} = (1,0,0,0,1,1,0,0,1,1,1,0,0)$$

We proved that:
None of the Amsterdam trains
gets lost (nice to know 😊).



Setting $b_3(s_{12})$ to 1 and $b_4(s_{13})$ to 1 leads to an additional 2 invariants

$$C_{R,1} = (11111100000000)$$

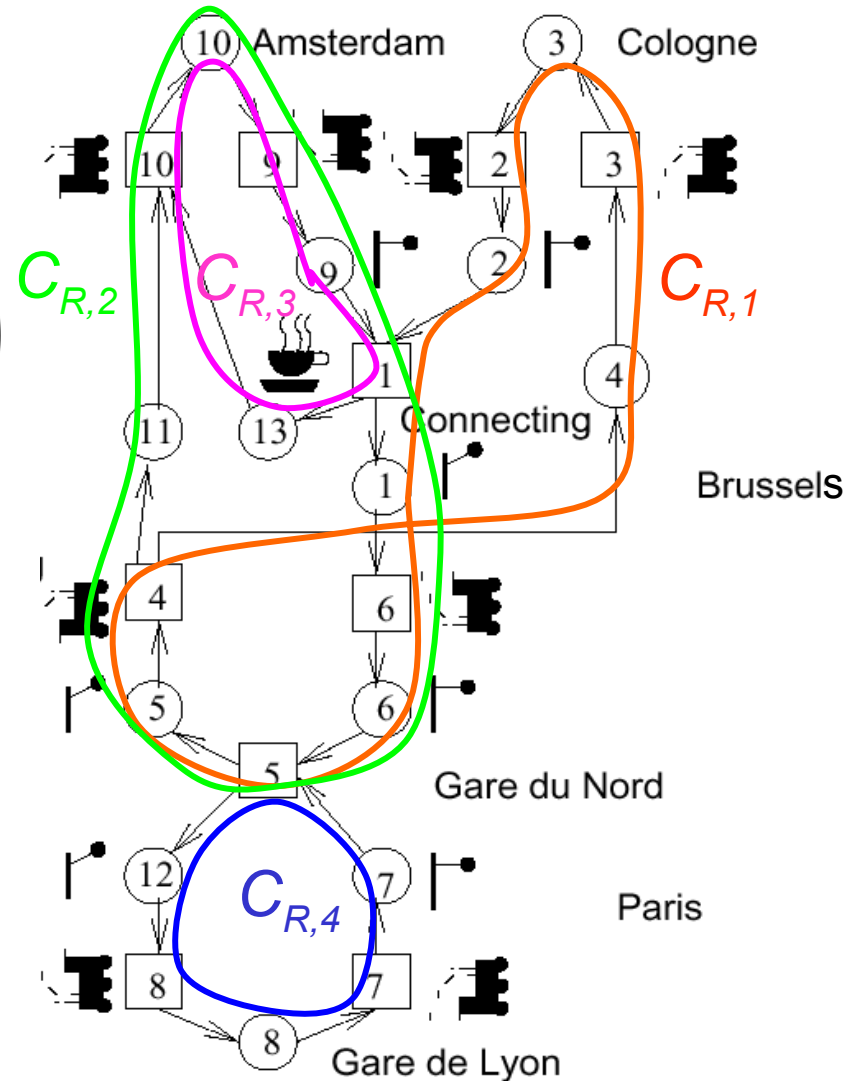
$$C_{R,2} = (10001100111000)$$

$$C_{R,3} = (00000000110001)$$

$$C_{R,4} = (00000011000010)$$

We proved that:

- the number of trains serving Amsterdam, Cologne and Paris remains constant.
- the number of train drivers remains constant.



Applications

- Modeling of resources;
- modeling of mutual exclusion;
- modeling of synchronization.

Predicate/transition nets

Goal: compact representation of complex systems.

Key changes:

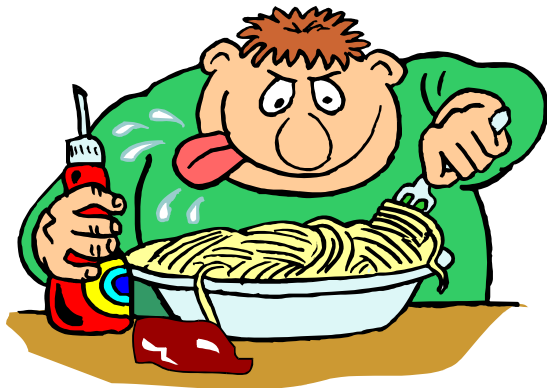
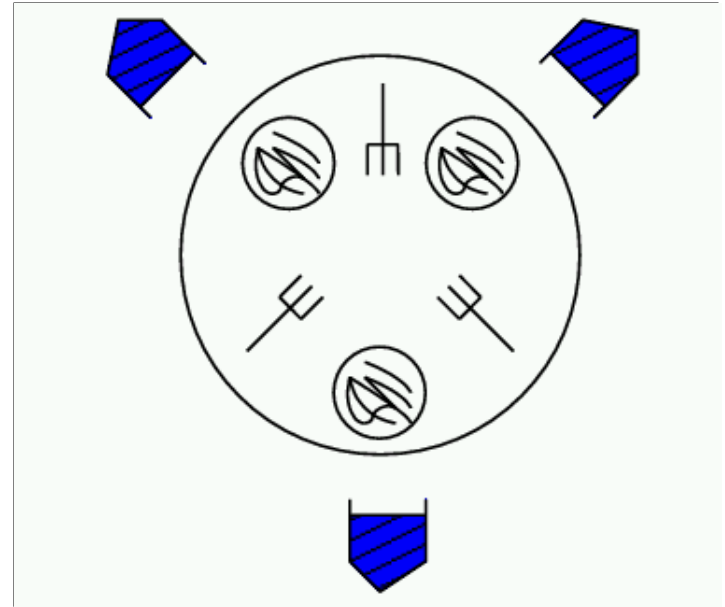
- Tokens are becoming individuals;
- Transitions enabled if functions at incoming edges true;
- Individuals generated by firing transitions defined through functions

Changes can be explained by folding and unfolding C/E nets,

☞ semantics can be defined by C/E nets.

Example: Dining philosophers problem

$n > 1$ philosophers sitting at a round table;
 n forks,
 n plates with spaghetti;
philosophers either thinking
or eating spaghetti
(using left and right fork).



2 forks
needed!

How to model conflict for forks?
How to guarantee avoiding
starvation?

Condition/event net model of the dining philosophers problem

Let $x \in \{1..3\}$

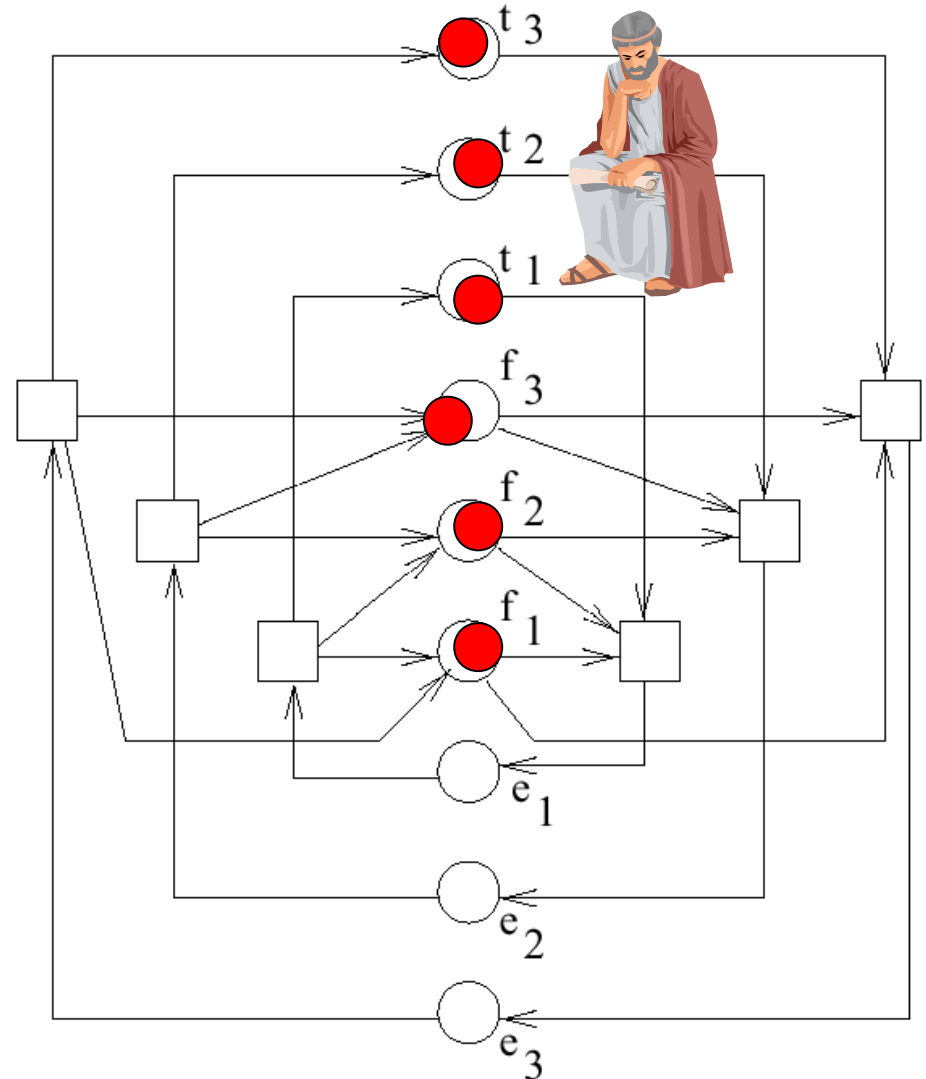
t_x : x is thinking

e_x : x is eating

f_x : fork x is available

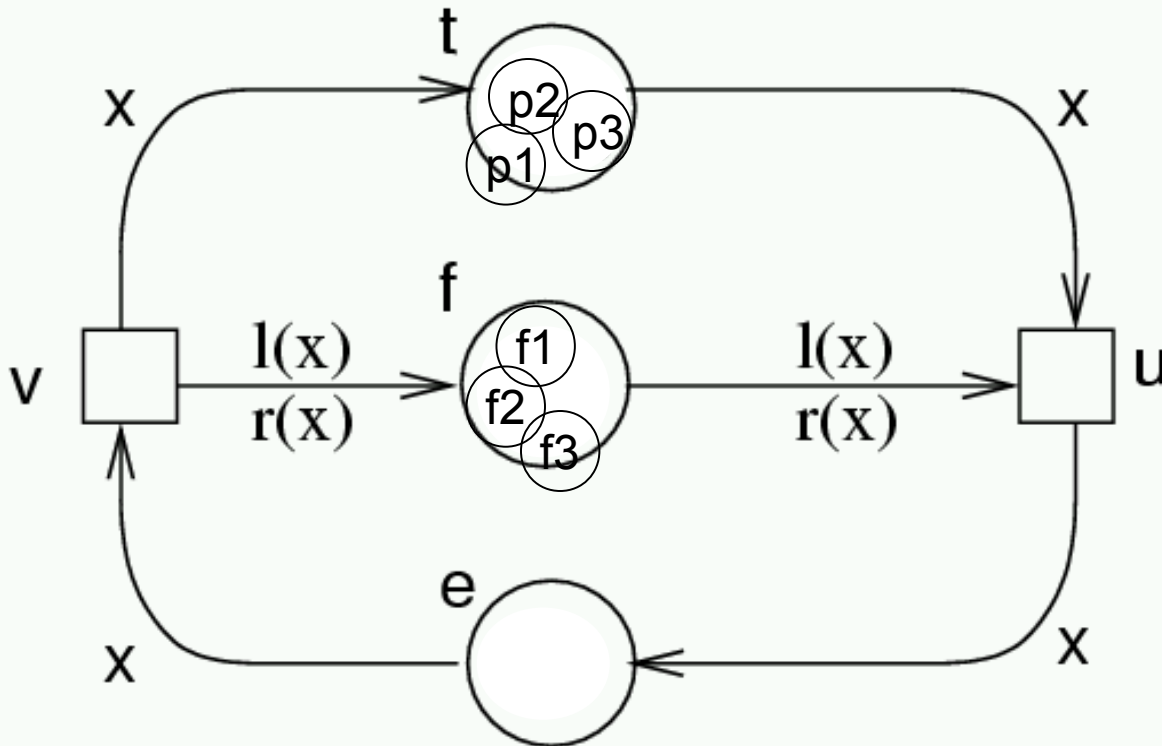
Model quite clumsy.

Difficult to extend to
more philosophers.



Predicate/transition model of the dining philosophers problem (1)

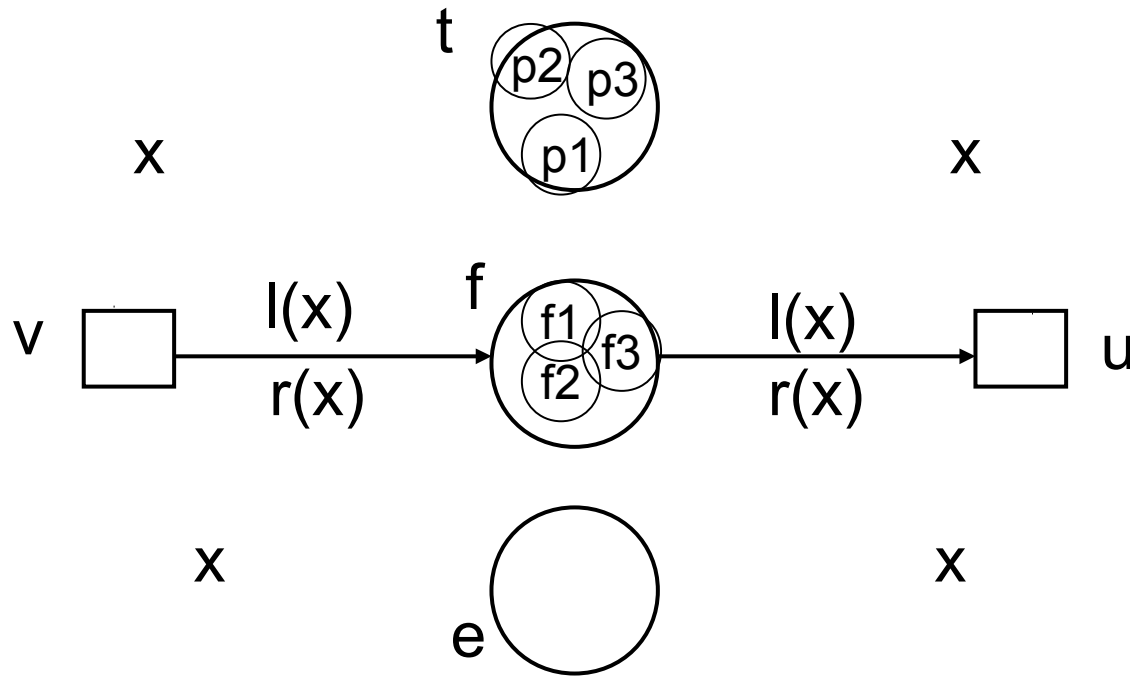
Let x be one of the philosophers,
let $l(x)$ be the left spoon of x ,
let $r(x)$ be the right spoon of x .



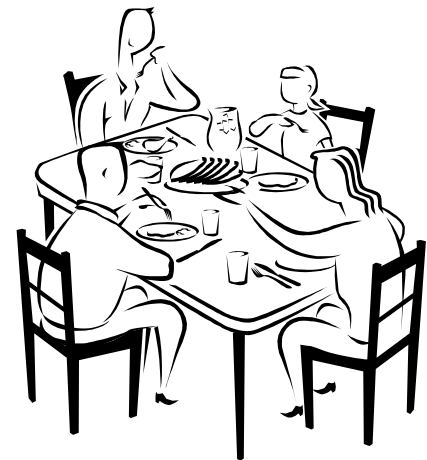
Tokens:
individuals.

Semantics can be
defined by
replacing net by
equivalent
condition/event
net.

Predicate/transition model of the dining philosophers problem (2)



Model can be extended to arbitrary numbers of people.



Evaluation

Pros:

- Appropriate for distributed applications,
- Well-known theory for formally proving properties,
- Initially a quite bizarre topic, but now accepted due to increasing number of distributed applications.

Cons (for the nets presented) :

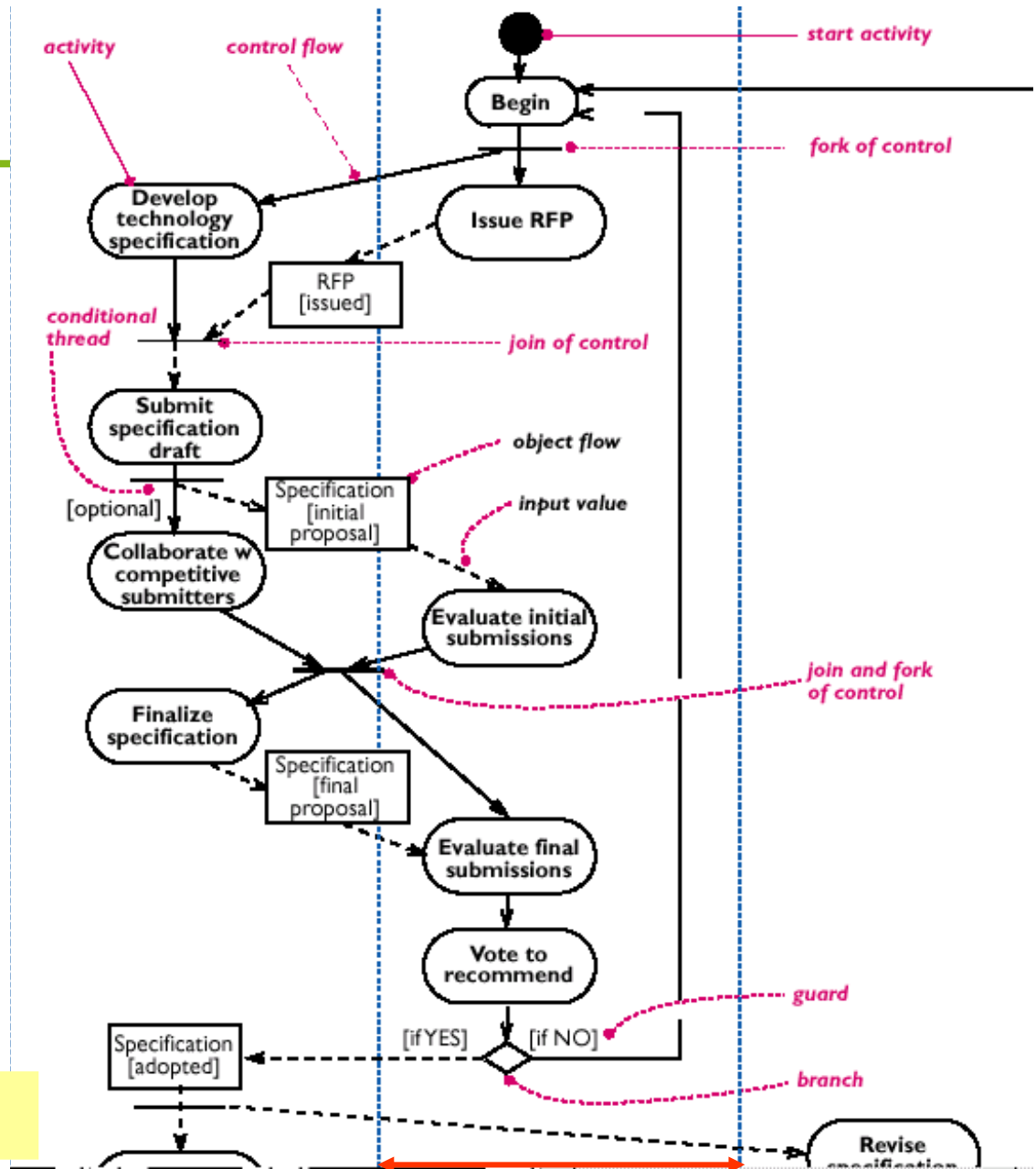
- problems with modeling timing,
- no programming elements,
- no hierarchy.

Extensions:

- Enormous amounts of efforts on removing limitations.

Computational Graphs @ UML: Activity diagram

Extended Petri nets. Include decisions (like in flow charts). Graphical notation similar to SDL.



© Cris Kobryn: UML 2001: A Standardization Odyssey, CACM, October, 1999

Summary

Petri nets: focus on causal relationships

Condition/event nets

- Single token per place

Place/transition nets

- Multiple tokens per place

Predicate/transition nets

- Tokens become individuals
- Dining philosophers used as an example

Extensions required to get around limitations