Time Petri Nets

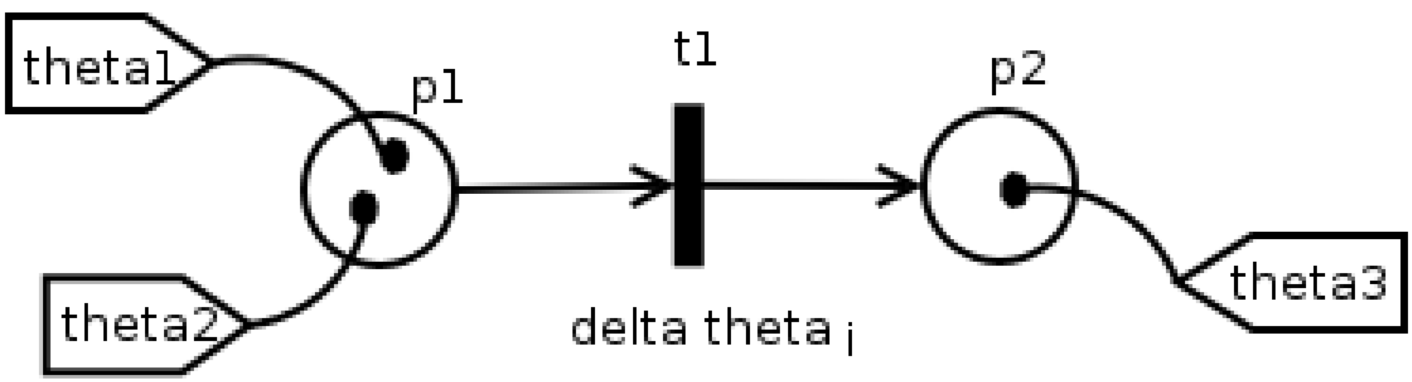
Miriam Zia
School of Computer Science
McGill University

Timing Specifications

- Why is time introduced in Petri nets?
  - To model interaction between activities taking into account their start and end times.

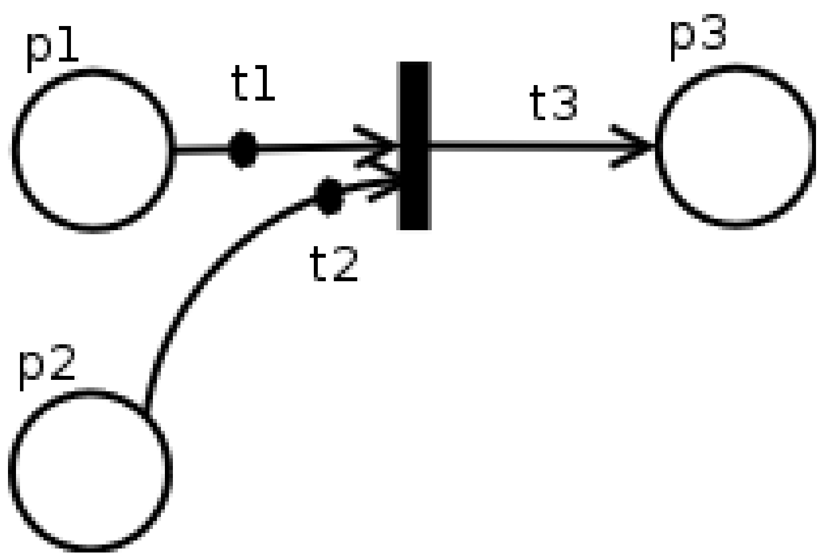
Time Associated with Tokens

- Each token is associated with a time-stamp **θ** that indicates when the token is available to fire a transition.



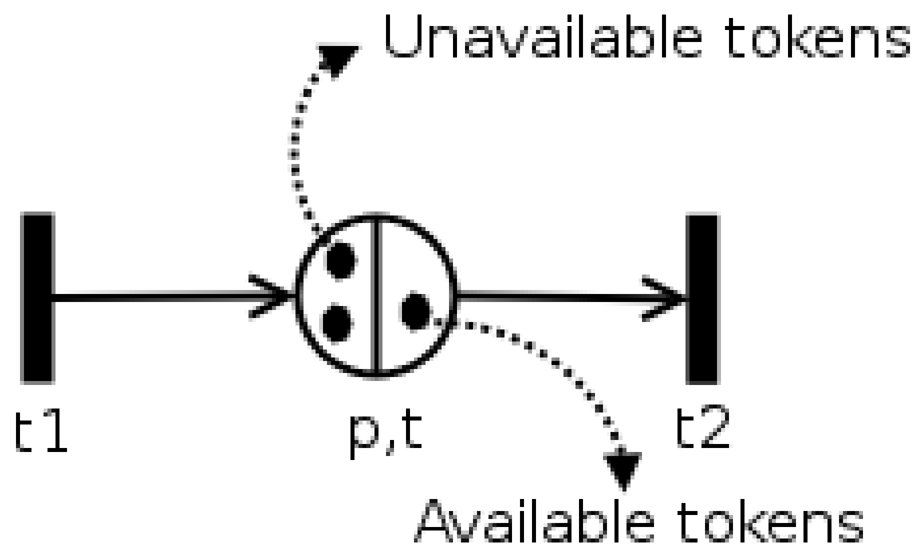
Time Associated with Arcs

- Each arc is associated with a traveling delay **t**.
- Tokens are available for firing only when they reach the transition.



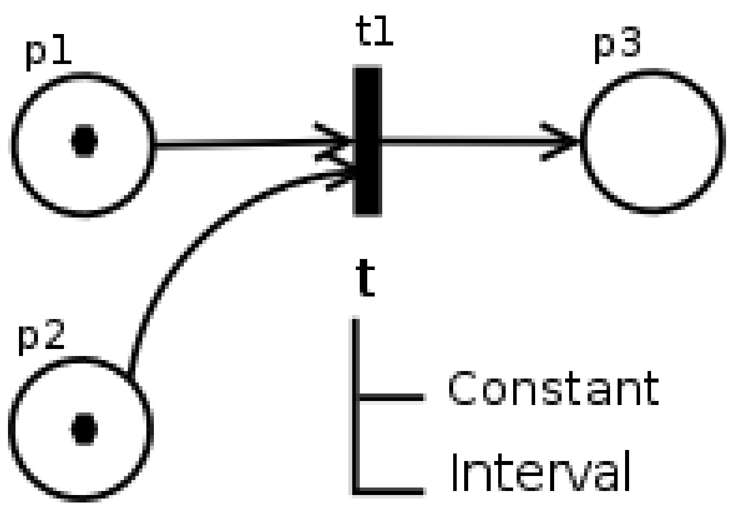
Time Associated with Places

- Timed Place Petri Nets (TPPN)
  - Each place **p** is associated with a delay attribute, say **t**.
  - Tokens generated in **p** only become available to fire a transition after the delay **t** has elapsed.



Time Associated with Transitions

- Timed Transition Petri Net (TTPN)
  - Each transition represents an activity.
    - Transition Enabling: start of activity.
    - Transition Firing: end of activity.
  - Two basic PN-based models were developed for handling time.



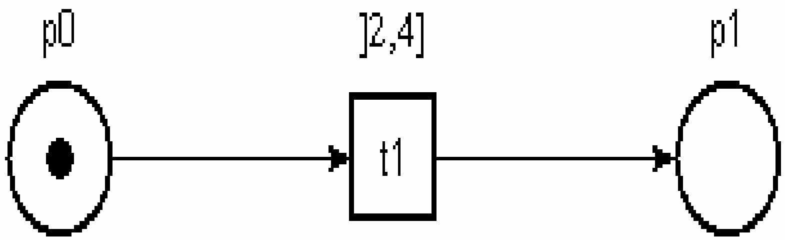
Ramchandani's Tim<u>ed</u> PN [Ram74]

- A firing duration **t** is associated with each transition of a PN.
- Firing rule:
  - Transitions are fired as soon as they are enabled.
  - Transitions take time **t** to fire.
- Used mainly for performance evaluation.

Merlin's Time PN [Mer74] (1/2)

- More general than Timed PN.
- TPN used to investigate recoverability problems in computer systems and in communications protocols.
- Two real numbers **a**,**b** are associated with each transition of a PN, with $0 \le a \le b \le \infty$.
  - **a**: time that must elapse between the ENABLING and the FIRING of a transition.
  - **b**: maximum time during which transition can be enabled without being fired.

Merlin's Time PN [Mer74] (2/2)



Times a and b for transition t1 are relative to the moment at which transition t1 is enabled.

- Assume t1 has been enabled at time *r*:
  - t1 cannot fire before time *r*+a.
  - t1 must fire before or at time *r*+b.

An Enumerative Approach for Analyzing Time Petri Nets (1/2)

- Research conducted at the LAAS of CNRS, Toulouse, France.
- Motivation: Specifying and proving correctness of time-dependent systems.
- Research:
  - Propose for TPN a technique for modeling the behaviour and analyzing the properties of timed systems.
    - Similar to the reachability analysis for PN.
  - Develop a software tool for analyzing TPN.
    - TIme petri Net Analyzer (TINA)

An Enumerative Approach for Analyzing Time Petri Nets (2/2)

- Two main papers:
  1. "An Enumerative Approach for Analyzing Time Petri Nets" (1983) [BM83].
  2. "Modeling and Verification of Time Dependent Systems Using Time Petri Nets" (1991) [BD91].

Outline of Paper Presentation

1. Time Petri nets.
   - States in a TPN.
   - Enabledness and firability condition of a set of transitions.
   - Firing rule between states.
   - Behaviour of TPN.
2. Method for analyzing TPN.
   - State classes.
   - Firing rule between state classes.
   - Reachability tree.
3. Some properties of Time Petri Nets.
4. TINA : TIme petri Net Analyzer.

Time Petri Net is a Tuple (1/2)

$TPN = \langle P,T,B,F,M_0,SIM \rangle$

P: finite nonempty set of places;

T: finite nonempty set of transitions ti can be viewed as an ordered set {t1, t2, …, t_i, …, };

B: backward incidence function

B: T x P → N (where N is the set of nonnegative integers);

F: forward incidence function

F: T x P → N;

M_0: initial marking function

M_0: P → N;

Time Petri Net is a Tuple (2/2)

TPN = ‹P,T,B,F,M0,SIM›

SIM: static interval mapping

SIM: T → Q* x (Q* U ∞) (where N is the set of positive rational numbers)

- A static interval is associated with transitions:
  $SIM(t_i) = (\alpha_i^s, \beta_i^s)$
- α_i^s, β_i^s are rationals such that:
  $0 \leq \alpha_i^s \leq \beta_i^s \leq \infty$
- (α_i^s, β_i^s) is called the static firing interval of transition t_i.
- Left bound α_i^s is the static Earliest Firing Time (static EFT) for t_i.
- Right bound β_i^s is the static Latest Firing Time (static LFT) for t_i.

A Couple of Comments

- Times α_i^s and β_i^s are relative to the moment at which t_i is enabled.
- If a pair (α_i^s, β_i^s) is not defined for t_i, it has the pair $(0, \infty)$ – classic PN transition
- In [BM91]: TPNs considered are such that none of their transitions may become enabled more than once "simultaneously" by any marking M:

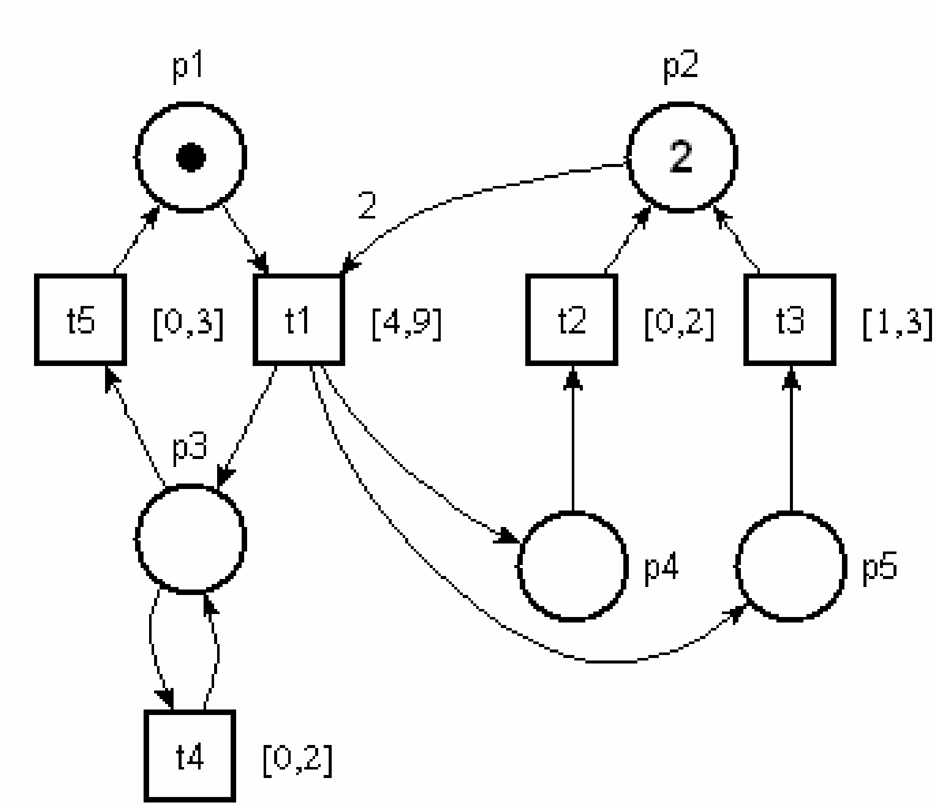
  for any enable transition t_i $(\exists p)(M(p) < 2 \cdot B(t_i,p))$

  - there is at least 1 place which prevents t_i from being firable twice.

States in a TPN Are a Pair (1/2)

- S = (M,I) consisting of:
  - A marking M.
  - A Firing Interval vector I
    - Associates with each transition enabled by M the time interval in which the transition is allowed to fire.

States in a TPN Are a Pair (2/2)



- S0 = (M0,I0), with
  M0: p1(1),p2(2)
  I0: {(4,9)}
- S1 = (M1,I1), with
  M1: p3(1),p4(1),p5(1)
  I1: {(0,2),(1,3),(0,2),(0,3)}
- S2 = (M2,I2), with
  M2: p2(1),p3(1),p5(1)
  I2:{(1,3),(0,2),(0,3)}
  if transition t2 fires

Enabledness Condition of a Set of Transitions

- Transition t_i becomes enabled at time *r* in state S = (M,I) in the usual PN sense:

  $M(p) \geq B(t_i,p)$ for all p in the incident set $I(t_i)$

Firability Condition of a Set of Transitions

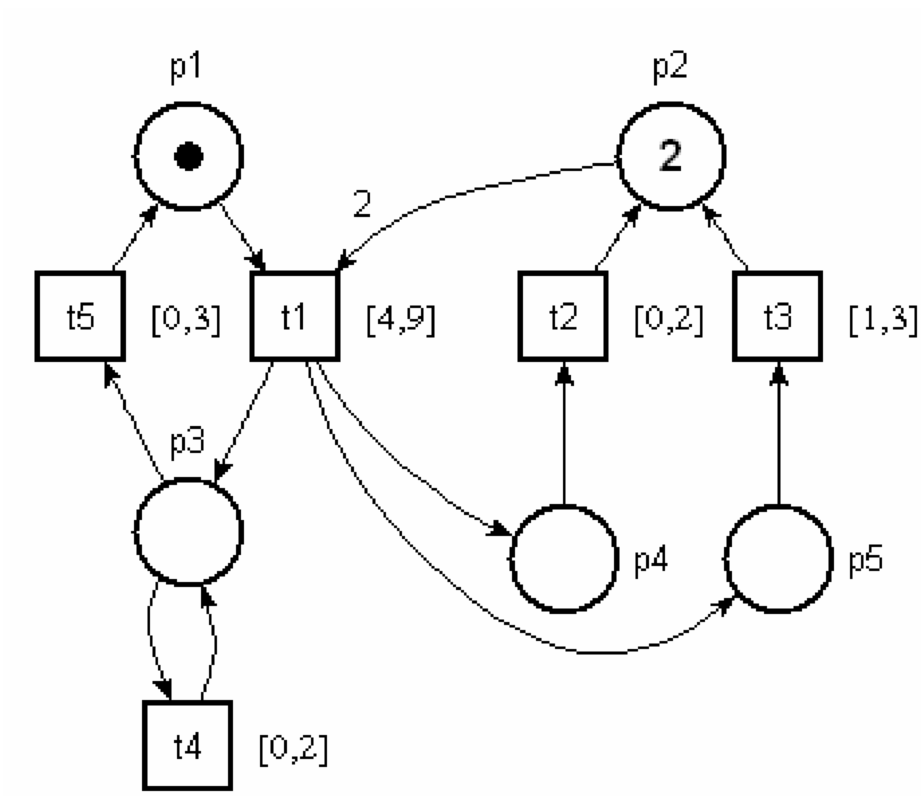
- Formally expressed by 2 conditions:
  - Condition 1: t_i is enabled by marking M at time *r (absolute enabling time)*.
  - Condition 2: the relative firing time θ (relative to *r*) is not smaller than the EFT of t_i and not greater than the smallest of the LFTs of all the transitions enabled by M:

$$\text{EFT of } t_i \leq \theta \leq \min\{\text{LFT of } t_k\}$$ (where k ranges over the set transitions enabled by M).

Firing Rule Between States (1/2)

- State S’ = (M’,I’) can be reached by firing t_i at relative time θ from state S=(M,I).
- S’ is computed in 2 steps:
  - M’ is computed, for all places p, as:
    (for all p)M’(p) = M(p) – B(t_i,p) + F(t_i,p)
  - I’ is computed in 3 steps:
    - Remove from I those intervals disabled when t_i is fired.
    - Shift by θ towards the origin of times all intervals of I that remained enabled; time is always nonnegative: I’ = (max(0,EFT*k* - θ), LFT*k* - θ)
    - Introduce in I’ the static intervals of the new transitions enabled.

Firing Rule Between States (2/2)



- S0 = (M0,I0), with
  M0: p1(1),p2(2)
  I0: {(4,9)}
- t1 fires at θ_1
  S1 = (M1,I1), with
  M1: p3(1),p4(1),p5(1)
  I1: {(0,2),(1,3),(0,2),(0,3)}
- If t2 fires at θ_2
  S2 = (M2,I2), with
  M2: p2(1),p3(1),p5(1)
  I2:{(max(0,1 - θ_2),3 - θ_2),
  (0, 2 - θ_2),
  (0, 3 - θ_2)}

Behaviour of a TPN (1/2)

- "transition t_i is firable from state S at time θ and its firing leads to state S' "

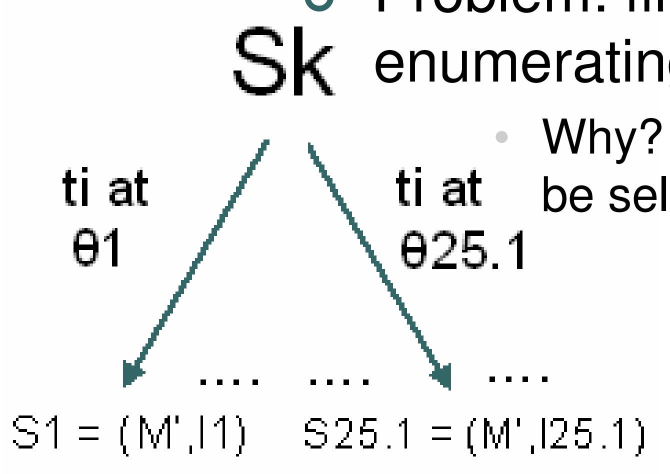
$$S \xrightarrow{(t_i,\theta_i)} S'$$

- A firing schedule will be a sequence of pairs (transition t, relative time θ):
  - (ti,θ1)·(t2,θ2) ·……· (tn,θn)
  - This schedule is feasible from a state S iff there exist states S1, S2, …, Sn such that:

$$S \xrightarrow{(t_1,\theta_1)} S1 \xrightarrow{(t_2,\theta_2)} S2 \dashrightarrow Sn\text{-}1 \xrightarrow{(t_n,\theta_n)} Sn$$

Behaviour of a TPN (2/2)

- The firing rule permits one to compute states and a reachability relation among them.
- The set of states that are reachable from the initial state, through a firing sequence ω, characterize the behaviour of the TPN.
  - Much like with reachable markings in PN.
- Problem: firing sequences can be defined but enumerating this set of states is not possible.
  - Why? Because there are infinite time values which can be selected to fire a transition from a given marking.



State Classes of a TPN (1/2)

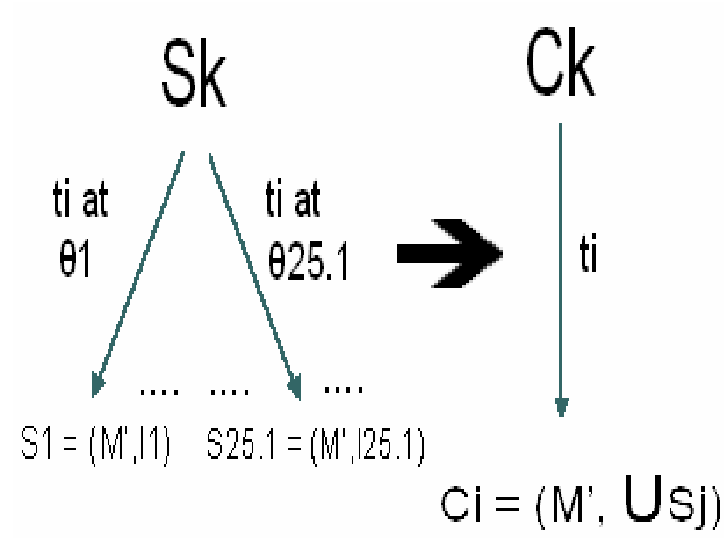
- Recap:

  A state is a set of all possible firing intervals, defined as the product set of the firing intervals of the transitions enabled by M.

- Now we consider the following:

  The set of all states reached from the initial state by firing all feasible firing values corresponding to the same firing sequence ω.

  This set will be called the state class associated with the firing sequence ω.



State Classes of a TPN (2/2)

- Class C = (M,D), associated with a firing sequence ω from the initial state, consisting of
  - A marking M of the class: all states in the class have the same marking.
  - A firing domain D of the class
    - Finitely represents the infinite number of firing domains of states possible from a marking M by firing schedules with firing sequence ω.
    - D may be expressed as the solution set of some system of linear inequalities:

      $D = \{t \mid A \cdot \underline{t} \geq \underline{b}\}$

      where A a matrix, $\underline{b}$ is a vector of constants, and variable ti corresponds to the i^{th} transition enabled by M.

      Note: t is an ordered set, and t(i) will refer to the i^{th} enabled transition.

Enabledness of Transitions from Classes

- Assuming t(i) is the i^{th} transition enabled by marking M, t(i) becomes enabled if:

  M(p) ≥ B(t(i), p) for all p in the incident set I(t(i))

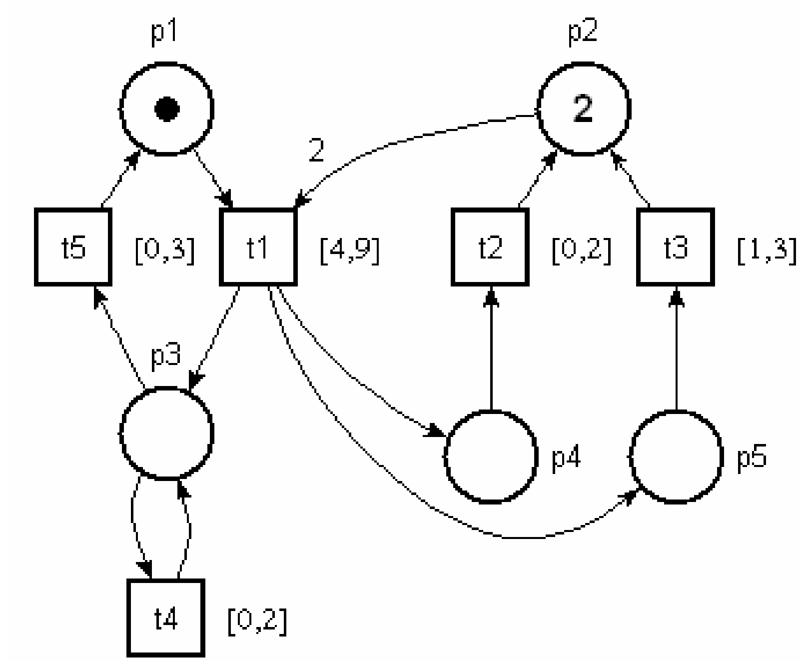
Firability of Transitions from Classes

- Transition t(i) is firable from class C = (M,D) iff:
  - Condition 1: t(i) is enabled by marking M.
  - Condition 2: the firing interval related to transition t(i) must satisfy the following augmented system of inequalities:

$$A \cdot t \geq b$$

$$t(i) \leq t(j) \text{ for all } j,\ j \neq i$$ (where t(j) also denotes the firing interval related to the j^{th} component of vector t)

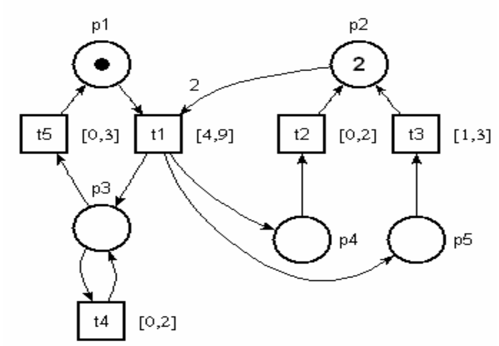
State Classes of a TPN (1/3)



- C0 = (M0,D0), with
  M0: p1(1),p2(2)
  D0: Solution set of
  $4 \leq \theta_1 \leq 9$
- t1 fires at θ_1
  C1 = (M1,D1), with
  M1: p3(1),p4(1),p5(1)
  D1: Solution set of
  $0 \leq \theta_2 \leq 2$
  $1 \leq \theta_3 \leq 3$
  $0 \leq \theta_4 \leq 2$
  $0 \leq \theta_5 \leq 3$
- Simple case: When firing t1, no transition already enabled remained enabled after the firing.

State Classes of a TPN (2/2)

- A complex case occurs when some transitions remain enabled.
- t2 can fire from time $\theta=0$ to $\theta=\theta_{max}$, e.g.: t2 can fire at any θ_2 in the interval $0 \le \theta_2 \le 2$
- Firing t2 is possible if the following system has a solution:



$$0 \le \theta_2 \le 2 \qquad (1)$$
$$1 \le \theta_3 \le 3 \qquad (2)$$
$$0 \le \theta_4 \le 2 \qquad (3)$$
$$0 \le \theta_5 \le 3 \qquad (4)$$
$$\theta_2 \le \theta_3 \qquad (5)$$
$$\theta_2 \le \theta_4 \qquad (6)$$
$$\theta_2 \le \theta_5 \qquad (7)$$

- Computation of all possible firing times for transitions can be handled by an adequate change of variables:
  - → θ_{2F} denotes the relative time at which t2 is fired.
- After the firing of t2, transitions t3, t4, t5 remain enabled while a time θ_{2F} has elapsed. Their new time values θ'_3 , θ'_4, θ'_5 can be defined by $\theta_i = \theta'_i + \theta_{2F}$
- Firing t2 is possible if the following system has a solution:

$$1 \le \theta'_3 + \theta_{2F} \le 3 \qquad (8)$$
$$0 \le \theta'_4 + \theta_{2F} \le 2 \qquad (9)$$
$$0 \le \theta'_5 + \theta_{2F} \le 3 \qquad (10)$$

State Classes of a TPN (2/2)

or

$1 - \theta_{2F} \leq \theta'_3 \leq 3 - \theta_{2F}$ (11)
$0 - \theta_{2F} \leq \theta'_4 \leq 2 - \theta_{2F}$ (12)
$0 - \theta_{2F} \leq \theta'_5 \leq 3 - \theta_{2F}$ (13)

with

$0 \leq \theta_{2F} \leq 2$ (14)

- (8), (9) and (10) can be rewritten:

  $1 - \theta'_3 \leq \theta_{2F} \leq 3 - \theta'_3$ (15)
  $0 - \theta'_4 \leq \theta_{2F} \leq 2 - \theta'_4$ (16)
  $0 - \theta'_5 \leq \theta_{2F} \leq 3 - \theta'_5$ (17)

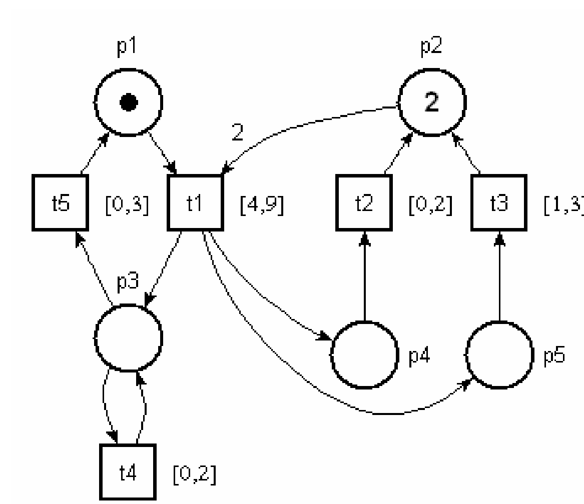
- Eliminating θ_{2F} gives:

  $0 \leq \theta'_3 \leq 3$ from (11) and (14)
  $0 \leq \theta'_4 \leq 2$ from (12) and (14)
  $0 \leq \theta'_5 \leq 3$ from (13) and (14)
  $\theta'_3 - \theta'_4 \leq 3$ from (15), (16) and (17)
  $\theta'_3 - \theta'_5 \leq 3$ from (15) , (16) and (17)
  $\theta'_4 - \theta'_3 \leq 1$ from (15) , (16) and (17)
  $\theta'_4 - \theta'_5 \leq 2$ from (15) , (16) and (17)
  $\theta'_5 - \theta'_3 \leq 2$ from (15) , (16) and (17)
  $\theta'_5 - \theta'_4 \leq 3$ from (15) , (16) and (17)

- The state class reached after firing t2 is:

  C2 = (M2,D2), with:

  M2: p2(1),p3(1),p5(1) and D2: solution set to the inequalities defined above.



Firing Rule Between State Classes

- Class C' = (M',D') can be reached by firing t(f) from class C = (M,D).
- C' is computed in 2 steps:
  - M' is computed, for all places p, as:
    (for all p)M'(p) = M(p) – B(t_i,p) + F(t_i,p)
  - D' is computed in 3 steps:
    - Add to the system A · t ≥ b the firability condition for t(f), leading to the augmented system:
      A · t ≥ b ; t(f) ≤ t(j) for all j, j ≠ f
      Make the change of variable: t(j) = t(f) + t''(j) and eliminate from the system the variable t(f).
    - Remove from the system obtained above all variables corresponding to transitions disabled when t(f) is fired.
    - Augment the system with new variables associated with each new transition enabled. These variables belong to their static firing interval.

Formal Definition of D

- The firing domains D of state classes for a T-Safe TPN can be expressed as solution sets of systems of inequalities of the following form:

$$\alpha_i \leq t(i) \leq \beta_i \text{ for all } i$$

$$t(j) - t(k) \leq \gamma_{jk} \text{ for all } j,k\ k \neq j$$

Reachability Tree (1/2)

- Using the firing rule, a tree of classes can be built.
  - The root is the initial class C, and there is an arc labelled ti from C to C' if ti is firable from class C, and if its firing leads to C'.
  - Each class will have a finite number of successors, at most one for each transition enabled by the marking of the class.
  - Any sequence of transitions firable in the TPN will be a path in this tree.

Reachability Tree (2/2)

- A finite graph will be associated to the TPN when the tree of classes will have a bounded number of *distinct* nodes.
  - The graph is obtained by grouping equal classes of the tree into the same class.
    - Two classes are defined to be equal if their markings are equal and their firing domains are equal.
    - A method to achieve this is to define the domains into some canonical form, and then compare these forms.
  - This will be called the reachability graph of the TPN.

Some Properties of TPN (1/2)

- The set of markings a TPN can reach from its initial marking M0 is denoted R(M0).
- The ***reachability*** problem is whether or not a given marking belongs to R(M0).
- The ***boundedness*** problem is whether or not all markings in R(M0) are bounded:
  - For all markings in R(M0) and for all places in P: $M(p) \leq k$, for some k in *N*

Some Properties of TPN (2/2)

- A TPN is said ***T-bounded*** if there exists a natural number k s.t. none of its transitions may be enabled more than k times simultaneously by any reachable marking.
  - for all ti in T there exists p in P such that: $M(p) < (k+1) \cdot B(ti,p)$
  - When k = 1, the TPN is said to be ***T-safe***.
- The reachability and boundedness problems for TPNs are undecidable.

So, What Do We Have Here?

- An approach for analyzing TPNs:
  - Permits one to check the properties of systems in the presence of timing specifications.

Possible Extensions

- No necessary or sufficient condition can be stated for the boundedness property
  - Must develop strong conditions!
- More specific and semantic checks could be developed
  - We could stop enumeration early on if the behaviour is not as expected.
- Develop alternative analysis techniques.

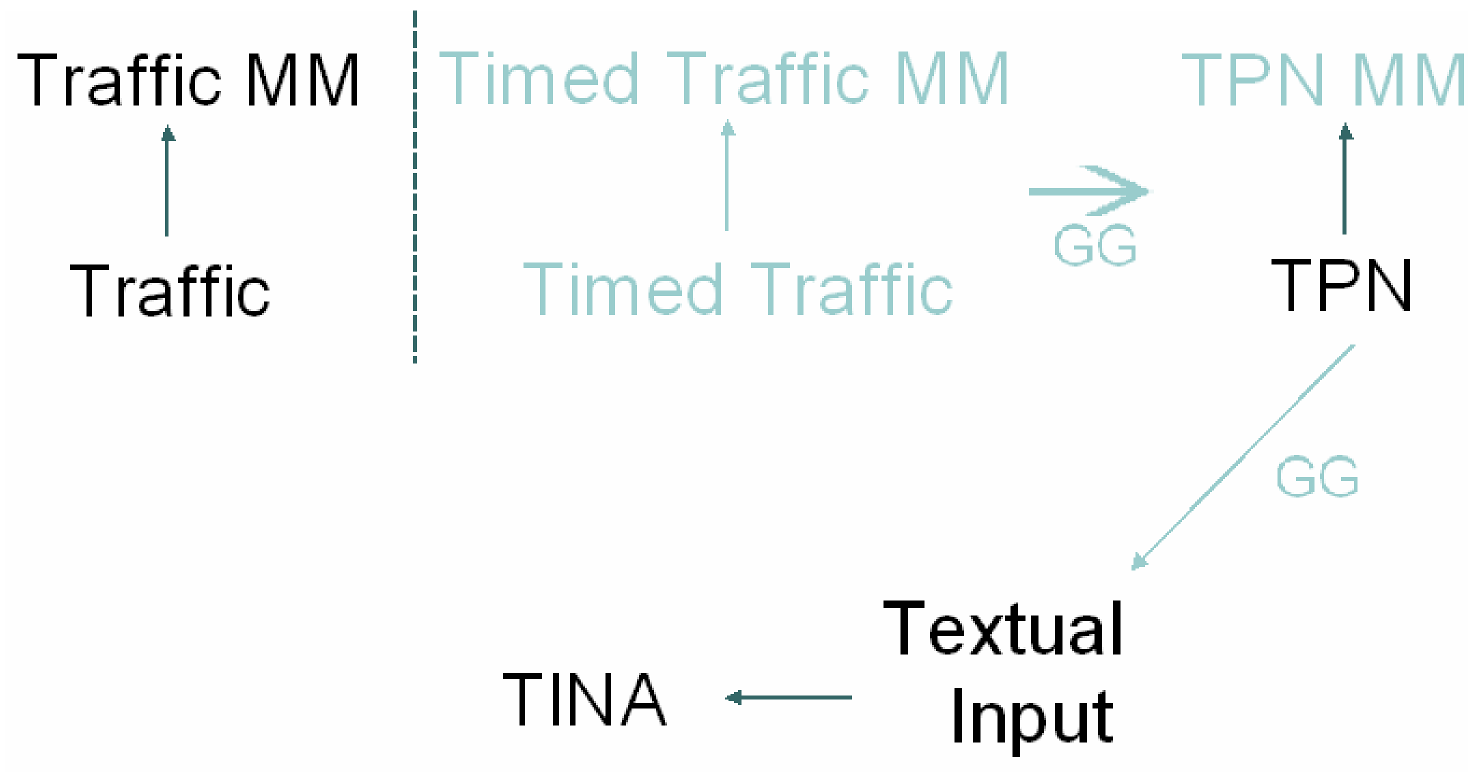
TINA

- Experimental toolbox for editing and analyzing PNs and TPNs.
  - tina:
    - Builds various state space abstractions for PN and TPN: reachability and coverability graphs (Karp & Miller technique), and efficiently checks the boundedness property.
    - Builds a linear state class graph of a TPN (Berthomieu & Menasche technique).
    - Takes as input descriptions of PN/TPN in textual or graphical form.
  - struct:
    - computes generator sets for semi-flows and flows.
    - Determines the invariance and consistence properties.
  - nd (NetDraw):
    - PN, TPN and Automata editor.
    - Allows one to create TPN in graphical or textual form.
    - Interfaced with the above tools.

TINA is not a Model-Checker

- It can't be used to check satisfaction of a concrete property (except reachability properties): no design verification performed.
- It can be used as a front-end for a model-checker.
  - It provides a reduced state space on which the properties can be checked more efficiently than on the original state space.

What Do I Intend to do with TPN?



References (1)

[BD91] Bernard Berthomieu and Michel Diaz, “Modeling and Verification of Time Dependent Systems Using Time Petri Nets”, IEEE Transactions on Software Engineering, 17(3), 1991.

link: http://ieeexplore.ieee.org/xpl/tocresult.jsp?isNumber=2506&puNumber=32

[BM83] Bernard Berthomieu and Miguel Menasche, “An Enumerative Approach for Analyzing Time Petri Nets”, IFIP Congress 1983, Paris, 1983.

link: http://www.laas.fr/tina/papers.php

[BRV04] B. Berthomieu, P.-O. Ribet and F. Vernadat, “The tool TINA -- Construction of Abstract State Spaces for Petri Nets and Time Petri Nets”, International Journal of Production Research, Vol. 42, No 4, July 2004.

link: www.laas.fr/~poribet/PUBLICATIONS/ribet_2004_ijpr.ps

References (2)

[Jan01] Mookyung Jang, "Introduction to Timed Petri Net", Data & Knowledge Engineering Lab, Postech I.E., June 2001.
link: http://home.postech.ac.kr/~mkjang/Resources/TimePetriIntro.pdf

[Mer74] P. Merlin, "A Study of the Recoverability of Computer Systems", Ph.D. Thesis, Department of Computer Science, University of California, Irvine, 1974.

[Ram74] C. Ramchandani, "Analysis of Asynchronous Concurrent Systems by Timed Petri Nets", Massachusetts Institute of Technology, Project MAC, technical Report 120, February 1974.