L01- Basic concepts of Petri Nets

José-Manuel Colom
May 14-25, 2012, Barcelona

Outline

I. Basic concepts of Petri Nets
II. Modelling features and abstraction
III. Structure and behaviour
IV. The Analysis Problem
V. The Synthesis Problem
I. Basic Concepts of Petri Nets

- Petri nets were developed in the early 1960s by C.A. Petri in his Ph.D. dissertation

- They are useful for modeling concurrent, distributed, asynchronous behaviour in a discrete system

- There are many families of Petri Nets for managing
  - Continuous/Hybrid systems
  - Timed systems
  - High-level information or tasks of the system
  - Variable Parameter Systems

---

A **Petri Net** is a 4-tuple

\[ N = (P, T, F, W) \]

where

- \( P = \{p_1, p_2, \ldots, p_m\} \) is a finite set of **places**
- \( T = \{t_1, t_2, \ldots, t_n\} \) is a finite set of **transitions**
- \( F \subseteq (P \times T) \cup (T \times P) \) is a set of **arcs**
- \( W: F \to \{1,2,3,\ldots\} \) is a **weighting function**
- \( P \cap T = \emptyset \) and \( P \cup T \neq \emptyset \)
I. Basic Concepts of Petri Nets

A Net System is a pair

\[(N, m),\]

where

\[N = (P, T, F, W),\]

is a Petri Net

\[m: P \rightarrow \{1, 2, 3, \ldots\}\]

is the initial marking
I. Basic Concepts of Petri Nets

Firing rule or occurrence rule of a transition
Given a net system \((N, m)\) and a transition \(t\) of \(N\)
- \(t\) is enabled at \(m\) iff for all \(p\) in \(\bullet t\), \(m[p] \geq W(p,t)\)
- \(t\) can occur at \(m\) iff \(t\) is enabled at \(m\)
- the occurrence of \(t\) leads to a new marking \(m'\) defined as:

\[
∀ p ∈ \bullet t, m'[p] = m[p] - W(p,t) \\
∀ p ∈ t\bullet, m'[p] = m[p] + W(t,p) \\
∀ p ∈ P\setminus(\bullet t \cup t\bullet), m'[p] = m[p]
\]

Incidence matrix of the Petri Net: \(C = \text{Post} - \text{Pre}\) (self-loops!!)
\[
∀ p \in P, t \in T: \text{If } (p,t) \in F \text{ then } \text{Pre}[p,t] = W(p,t) \text{ else } \text{Pre}[p,t] = 0 \\
∀ p \in P, t \in T: \text{If } (t,p) \in F \text{ then } \text{Post}[p,t] = W(t,p) \text{ else } \text{Post}[p,t] = 0
\]
I. Basic Concepts of Petri Nets

- Net subclasses are defined at the syntactical level, i.e. imposing constraints to the interplaying of conflicts and synchronizations
  - Modelling capabilities are reduced
  - Analysis is easier

- Some syntactical subclasses (ordinary nets)
  - **State machine**: for all $t \in T$: $|\cdot t| = |t\cdot| = 1$;
  - **Marked graphs**: for all $p \in P$: $|\cdot p| = |p\cdot| = 1$;
  - **Free Choice**:
    - for every $p, q \in P$: $p\cdot \cap q\cdot = \emptyset$ or $p\cdot = q\cdot$;
    - for every $t, u \in T$: $\cdot t \cap \cdot u = \emptyset$ or $\cdot t = \cdot u$;
  - **Asymmetric Choice, or simple**:
    - for every $p, q \in P$: $p\cdot \cap q\cdot = \emptyset$ or $p\cdot \supseteq q\cdot$ or $q\cdot \supseteq p\cdot$;

II. Modelling Features...

- Modelling expressivity
  - Sequences
  - Conflicts (decisions, iterations)
  - Concurrency and synchronizations
II. Modelling Features...

1. Rendezvous, RV

2. Semaphore, S

3. Symmetrical RV using semaphores S1 and S2

4. Fork-Join

5. Guard

6. Shared resource R

7. Subroutine
II. … Methodology

...Refinement
II. ... Abstraction

Let us consider a manufacturing cell ...

<table>
<thead>
<tr>
<th></th>
<th>op1</th>
<th>op2</th>
<th>op3</th>
<th>op4</th>
<th>op5</th>
<th>Capacity</th>
</tr>
</thead>
<tbody>
<tr>
<td>M1</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>3 parts</td>
</tr>
<tr>
<td>M2</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>2 parts</td>
</tr>
<tr>
<td>M3</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>2 parts</td>
</tr>
</tbody>
</table>

... where two types of parts must be produced

Type 1
op1 → op2

Type 2
op3 → op4

G1
begin (M1,op1) (M3,op1) (M2,op2) → end

G2
begin (M2,op3) (M1,op4) → end
II. ... Abstraction

A net of the S³PR subclass

Systems of Simple Sequential Processes with Resources

II. ... Methodology

- Steps to build the model:
  - Strongly connected state machines – **Sequential tasks**
    - one for each type of part – **Process Plans**
    - two kind of places – **Idle State + Processing/Transport States**
    - a structural termination property: all circuits contain the idle place – Production sequences
II. … Methodology

- a set of resource places
  - each resource place – **Type of resources**
  - initial marking of a resource place – **Number of identical copies of a type of resource; or Capacity of a given resource where a part is**
- only one copy of only one resource type per part in a state

![Petri Net Diagram]

- an **admissible** initial marking
- composition of production plans with resources via **fusion** of resource places
What’s Structure Theory of Petri Nets?
III. Structure and Behaviour

What’s Structure Theory of Petri Nets?

A ▲

C

Start

C1

C1/A/Stop

Start

C1/A-C/R

C1/C/Stop

C1/C-A/L
What’s Structure Theory of Petri Nets?

III. Structure and Behaviour
III. Structure and Behaviour

What’s Structure Theory of Petri Nets?

Synchronised Transitions

C1/A/Stop; C2/B/Stop

C1/A-C/R; C2/B-D/R

C1/C/Stop; C2/B-D/R

C1/C-A/L; C2/D-B/L

C1/A/Stop; C2/D-B/L

C1/C-A/L; C2/B/Stop

C1/C-A/L; C2/D-B/L

C1/A-Stop; C2/B/Stop

C1/A-C/R; C2/B-D/R

C1/C/Stop; C2/B-D/R

C1/C-A/L; C2/D-B/L

C1/A/Stop; C2/D-B/L

C1/C-A/L; C2/B/Stop
III. Structure and Behaviour

What’s Structure Theory of Petri Nets?
III. Structure and Behaviour

What’s Structure Theory of Petri Nets?
III. Structure and Behaviour

What’s Structure Theory of Petri Nets?

IV. The Analysis Problem

6n+2 vs. $2^{n+1}$
IV. The Analysis Problem

Model of the system

Specification of the system

Properties

Behavioural properties (for $m_0$)

- **Boundedness**: finiteness of the state space, i.e. the marking of all places is bounded
  \[ \forall p \in P \; \exists k \in \mathbb{N} \; \text{ such that } \; m(p) \leq k \]
- **Safeness** = 1-boundedness (binary marking)
- **Mutual Exclusion**: two or more places cannot be marked simultaneously (problem of shared resources)
- **Deadlock**: situation where there is no transition enabled
- **Liveness**: infinite potential activity of all transitions
  \[ \forall t \in T, \forall m \text{ reachable}, \; \exists m', \; m [\sigma > m'] \; \text{ such that } \; m'[t >] \]
- **Home state**: a marking that can be recovered from every reachable marking
- **Reversibility**: recovering of the initial marking
  \[ \forall m \text{ reachable}, \; \exists \sigma \; \text{ such that } \; m [\sigma > m_0] \]
Basic properties

- Liveness, boundedness, and reversibility are just three different good behavioral properties

- They are three independent properties

Structural properties:

The structure essentially enforces the property

- N is **structurally bounded** if for all \( m_0 \), \( <N, m_0> \) is bounded

- N is **structurally live** if there exists a \( m_0 \) for which \( <N, m_0> \) is live
IV. The Analysis Problem

- Techniques for the **analysis** of net systems
  - **Enumeration** (Reachability/Coverability graphs)
  - **Transformation**: Reduction
  - **Structural**: bridge between Behaviour and Structure

- **Structural analysis**
  - Based on the separation: Structure - State
    \[ S = <N, mo> \]
  - Basic approaches
    1) **Reduction**: Looking for “simpler” net systems preserving the properties under study
    2) **Global structural approaches** (or structural techniques)

---

- **Global structural approaches:**
  - The structure, \( N \), leads to two perspectives
    - GRAPH based perspective: \( N = (P, T, F, W) \)
    - LINEAR based perspective: \( N = (P, T, \text{Pre, Post}) \)
      - Linear Algebra
      - Convex Geometry/Linear Programming
  - **Problems with the basic structural approaches**
    - Reduction may be not complete for a given kit of reduction rules and a given net system
    - Global structural analysis can lead only to necessary or sufficient conditions (semidecision techniques)
Complementariness of analysis techniques

- **Enumeration:**
  - Powerful, if we consider bounded net systems
  - High computational complexity
  - Conclusions are for a given $m_0$

- **Transformation:**
  - Computational efficiency (on standard kit)
  - Incompleteness in general systems (completeness for some subclasses)

- **Structural:**
  - Computational efficiency
  - Semidecision in general nets (decision for net subclasses)

The trade-off of analysis methods:

*Computational Efficiency vs Decision power*

- Therefore: The practical analysis of general net systems may need *cooperative use of several analysis techniques*

- Nevertheless, necessary and sufficient conditions in the characterization of some properties are available for net subclasses (i.e. simpler nets)

- Net/Systems subclasses: They are defined constraining the interleaving of synchronizations and conflicts

*For “simpler” systems, the analysis problem become “easier”*
V. The Synthesis Problem

The synthesis problem ~ *Synthesis of controllers*

- Liveness enforcing
- Forbidden state problem
- ...

**Example** - What to do when deadlocks can appear?

- the ostrich algorithm
- deadlock detection and recovery
- deadlock avoidance
- **deadlock prevention**

---

The synthesis problem in a RAS:

- To make **live** the original net system,
- by means of **deadlock prevention control policies**,
- based on the **non-liveness characterizations**
- generating **additional restrictions** to the way resources are allocated, and
- implemented by means of **Petri net objects** (virtual resources) or by means of the interpretation of the net (transition guards)
V. The Synthesis Problem

The synthesis problem:

The dream

The realities of the life

Good markings
Very bad markings
Bad markings
V. The Synthesis Problem
V. The Synthesis Problem

Permissiveness Criterion