Scenario duration characterization of t-timed Petri nets using linear logic

Brigitte Pradin-Chézalviel
Robert Valette
Luis Allan Künzle
  – e-mail: robert@laas.fr
  – http://www.laas.fr/~robert

Introduction (1)

The objective of this communication is:

• Characterize partial order relations in a scenario
  – resulting from the structure of the Petri net
  – resulting from the initial and the final markings (reachability)

• Derive an algebraic formulae of the duration of the scenario
  – how long to go from the initial marking to the final one
  – a duration is attached to each transition
  – within the framework of t-timed Petri net (firing duration which may be imprecise or fuzzy), also valid for t-time (enabling duration)
Introduction (2)

Relationships between Petri nets and linear logic

• **Equivalence between reachability and provability**
  – some sequent is provable in intuitionist linear logic sequent calculus
  – there exists a firing sequence leading from a marking M to M'

• **More than one way for representing PN in linear logic**
  – Gelhot's approaches is based on proper axioms for representing transitions
  – markings are not explicitly represented
  – we represent transitions by formulas with a linear implication
  – markings (and Pre(t) and Post(t)) are monomial in \( \otimes \)

Introduction (3)

Two approaches for exploiting this equivalence

• **Gelhot's approach: rewriting a proof tree**
  – first derive a proof in linear logic without any restriction
  – second rewrite the proof tree (rewriting rules) in order to increase concurrency
  – the rewriting process is similar to "cut elimination" in sequent calculus (each cut rule utilization corresponds to a serialization of two subproofs)
  – an algebraic expression with ";" and "/" (may introduce unnecessary partial order r.)

• **A canonical "cut free" proof tree**
  – possible because no proper axioms (just linear logic)
  – exhibit logical partial order constraints among transition firings
  – no explicit concurrency relations but algebraic expression for duration
Scenario (1)

• **What is a scenario? A linear logic sequent**
  – a Petri net (its structure) (structural partial order and concurrency relations)
  – an initial marking and a final marking (dynamical concurrency)
  – a multi set of transitions to be fired

• **Conflicts between tokens**
  – select one token in a place to fire a transition

• **Conflict between transitions**
  – select one transition to consume a token

→ **A sequent defines a disjunctive set of scenarios**
  – add new partial order relations (solve the conflicts) to completely specify a scenario

---

Scenario (2)

**Partial order relations**
– they depend on Petri net structure and initial marking
– they have to be defined between transition firings, not statically between transitions

\[
\begin{align*}
\text{t}_1, \text{duration } d_1 & : A \otimes C, oB \otimes C \\
\text{scenario } s_1 & : A \otimes A \otimes C, t_1, t_1 \mid - B \otimes B \otimes C \\
\text{duration } 2d_1 \\
\text{scenario } s_2 & : A \otimes A \otimes C \otimes C, t_1, t_1 \mid - B \otimes B \otimes C \otimes C \\
\text{duration } d_1
\end{align*}
\]
Scenario (3)

Conflict between tokens
– choosing one token or the other one in C results in an other duration
– a scheduling decision is required

\[
\text{scenario } s : \ A \odot B, t_1, t_2, t_3 \vdash C \odot D
\]

\[
\text{duration } \left( \max(d_1, d_2 + d_3) \right) \text{ or } \left( \max(d_2, d_1 + d_3) \right)
\]

(t_2 before t_3) or (t_1 before t_3) to define a scenario and a duration

Scenario (4)

Conflict between transitions
– fire first t_1 or t_2, a scheduling decision

\[
\text{scenario } s : \ A \odot B \odot R, t_1, t_2, t_3, t_4 \vdash E \odot F \odot R
\]

\[
\text{duration } \left( d_1 + \max(d_3, d_2 + d_4) \right) \text{ or } \left( d_2 + \max(d_4, d_1 + d_3) \right)
\]

It is necessary to add a new partial order constraint in order to completely define a scenario
Scenario (5)

- Completely specified scenarios
  - no conflicting tokens (or evident cases), no conflicting transitions, a unique set of partial order relations among the transition firings of the scenario
  - otherwise there is one set of partial order relations for each schedule = for each set of decisions solving the conflicts

- A first step for this new approach
  - define an algorithm for completely specified scenarios
  - when there is no conflicting transitions, the best solutions is always obtained when firing the transitions as soon as possible => this policy allows solving many token conflicts (select the oldest one)

Algorithm (1)

Principle of a canonical proof tree:

- Left introduction rule of linear implication = causality
- Break down marking formulas into blocks of atoms
Algorithm (2)

Computation of scenario duration

- temporal stamps attached to formulas (markings)
  - a token produced by a firing and consumed by another one results in a partial order relations (add duration)
  - to fire a transition, wait for the presence of all tokens (max of temporal stamps)

\[ d_{A1} \leq d_{A2} \]

Example (1)

- The max of the three paths
- The method implicitly constructs the corresponding PERT graph

scenario: A, t_1, t_2, t_3, t_4, t_5, t_6 | I
duration \((d_1 + d_4 + \max((d_2 + d_3), (d_6 + \max(d_2, d_3))))\)
Example (2)

- The max of the two paths (E is no longer a constraint)
- The method implicitly constructs the corresponding PERT graph
- But the PERT graph structure differs from that of the Petri net

scenario: A ⊗ E, t₁, t₂, t₃, t₄, t₅, t₆ |− E ⊗ I
duration (d₁ + d₄ + max((d₂ + d₃), (d₅ + d₆)))

Example (3)

- The Pert graph has five paths but two are included in others
- The net is not safe but the token choice can be done by the earliest policy

scenario: A ⊗ B ⊗ E ⊗ F, t₁, t₂, t₂, t₃, t₄, t₅, t₅, t₆, t₆ |− E ⊗ I ⊗ I
duration (d₄ + max((d₂ + d₆), (d₁ + max((d₂ + d₃), (d₅ + d₆)))))
**Example (4)**

- The Petri net and the scenario may contain loops
- We assume known how many times transitions are fired

**Conclusion (1)**

- **Comparison with the graph class approach**
  - no explosion due to concurrency, absolute time, scenario duration, no reachability

- **Analysis is local (between two markings)**
  - no need of a complete state graph

- **Consistent with Chretienne results**
  - a schedule is a PERT graph, token and transition conflicts, logical formal basis

- **Partial order and concurrency relations remain implicit**
  - trans. firings, a partial order when one token is produced by t1 and consumed by t2
Conclusion (2)

Perspective

• Simple in simple cases, exploit qualitative (logical) constraints
  – cases with conflicts (meta heuristic to generate a good completely specified scenario)
  – take into account quantitative information

• Complexity results from a combination of
  – conflicting transitions in a safe net
  – conflicting tokens (non ordered time stamps)
  – loops (complex causality structures)

• Logic offers a formal framework
  – break down initial marking (or final one) but not the two simultaneously