Dynamic Analysis of ECATNets

Noura Boudiaf * & Allaoua Chaoui**
Département d'Informatique, Université d'Oum El Bouaghi, Algérie
boudiafn@yahoo.com

**Department of Software Engineering, Faculty of IT,
University of Philadelphia, Jordan
chaoui@philadelphia.edu.jo

Moca'04, Aarhus, Denmark, October 2004

Outline

- ECATNets
- Motivations
- Study of Unbounded Places Case
- Dynamic Analysis
- Example
- Conclusion & future work

ECATNets

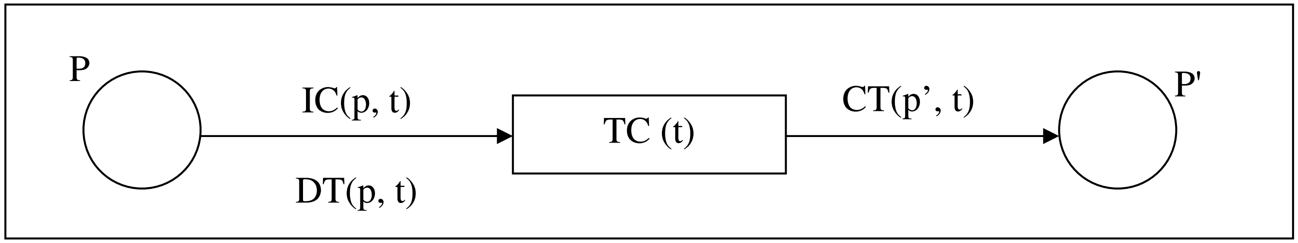
- **Characteristics**
  - Sound semantics based on rewriting logic
  - Simulation
  - Model checking verification
  - Static analysis
  - Presence of tool based Maude system

ECATNets

Syntax

- Integration of :

  Abstract data types & Petri Nets



IC(p,t), DT(p,t) and CT(p’,t) : multi-sets of algebraic terms

TC(t) : boolean expression

ECATNets

Semantic

- A transition t is enabled if :
  - IC(p, t) is included in M(p) for every input place p
  - Tc(t) is true
  - the addition of CT(t, p') to each output place p' must not result in p' exceeding its capacity when this capacity is finite
- If t is fired then :
  - DT(p, t) is removed from the input place p
  - CT(t, p') is added to the output place p'

ECATNets

Semantic : Rewriting Rules

We use the notation :

- $(p, [m]_{\oplus})$,
  - p is a place of the net,
  - $[m]_{\oplus}$ a multi-set of algebraic terms
  - The multiset union on the pairs $((p, [m]_{\oplus})$ will be denoted by $\otimes$ w.r.t. the ACI (*Associativity Commutativity Identity*) axioms for $\otimes$

ECATNets

Semantic : Rewriting Rules

- **IC(p,t) is of the form $[m]_{\oplus}$**
- **case1** $[IC(p,t)]_{\oplus} = [DT(p,t)]_{\oplus}$
  - The form of the rule is then given by:

    $$t : (p, [IC(p ,t]_{\oplus} \rightarrow (p', [CT(t,p')]_{\oplus}$$
- **case2** $[IC(p,t)]_{\oplus} \cap [DT(p,t)]_{\oplus} = \phi_M$
  - The form of the rule is given by:

    $$t : (p, [IC(p,t)]_{\oplus} \otimes (p, [DT(p,t)]_{\oplus} \cap [M(p)]_{\oplus}$$

    $$\rightarrow (p, [IC(p,t)]_{\oplus} \otimes (p',[CT(t,p')]_{\oplus})$$
- **Case3** $[IC(p,t)] \cap [DT(p,t)] \neq \phi_M$
  - that it could be brought to the two already treated cases

ECATNets

- **IC(p,t) is of the form ~[m]$_\oplus$**
  - The form of the rule is given by:

    $t : (p, [DT(p,t)]_\oplus \cap M(p)]_\oplus \rightarrow (p',[CT(t,p')]_\oplus)$

    $\text{if } ([IC(p,t] _\oplus \backslash ([IC(p,t)] _\oplus \cap [M(p)]_\oplus) = \phi_M \rightarrow [false]$
- **IC(p,t) = empty**
  - The form of the rule is given by:

    $t: (p, [DT(p,t] _\oplus \cap [M(p) \cap) \rightarrow$

    $(p',[CT(t,p') \cap) \text{ if } ([M(p) \cap \rightarrow \phi_M$
- When the place capacity C(p) is finite, the conditional part of the rewrite rule will include the following component:
  - $([CT(p,t)] _\oplus \otimes [M(p] _\oplus) \cap [C(p)] _\oplus \rightarrow$

    $[CT(p,t)] _\oplus \otimes [M(p)]] _\oplus$ (**Cap**)

Motivations

- Maude Model Checking requires finite state space
- Lack of any tool for Reachability Analysis of ECATNets
- Easy implementation of Dynamic Analysis tool under Maude system
  - Reflectivity of rewriting logic : The power of this logic to interpret itself allows the modeling of an ECATNet and then act on it

Motivations

Main Aim

- Enrichment of ECATNets' possibilities verification
  - Static proprieties of infinite system or finite system
    - Liveness properties, Deadlock detection, …

Study of Unbounded Places Case

- Unbounded place
  - The number of algebraic terms in this place increases infinitely
- We exclude from our study the cases :
  - (IC(p,t) is of the form $\sim[m]_{\oplus}$, and IC(p,t) = empty)

Study of Unbounded Places Case

- **ECATNet's places with only infinite capacity**
  - (Absence / Presence) of transitions conditions
    - Monotony respected
- **ECATNet's places with finite capacity & infinite capacity**
  - (Absence / Presence) of transitions conditions
    - Monotony is not always respected : The following proved proposition allows detecting infinite increase of terms in places with infinite capacity

Study of Unbounded Places Case

Proposition. If $M \xrightarrow{S} M'$ and $M \subseteq M'$ and the first transition in S become not enabled since M'. If it exists S' such that $M' \xrightarrow{S'} M''$. S' stops when S become enabled. If we have the following case :

$M \xrightarrow{S} M' \xrightarrow{S'} M'' \xrightarrow{S} M'''$ and if $M' \subseteq M'''$ and $M''(p) \subseteq M(p)$ for each place p with bounded capacity yielding S disabled at M', then every infinite place p' such $M'(p') \subset M'''(p')$ is an unbounded place.

Recovery of the Infinite Increase of Elements Numbers in Multi-set

- For every multi-set *m* whose elements are of type *T*, we create ω_T which covers the increase of number of elements in this multi-set :
  - $\forall m : T$
  - $m \cup \omega_T = \omega_T$
  - $m \subseteq \omega_T$
  - $\omega_T \setminus m = \omega_T$
  - $\omega_T \subseteq \omega_T$
  - $\omega_T \cup \omega_T = \omega_T$

Recovery of the Infinite Increase of Elements Numbers in Multi-set

In rewriting logic

- $(p, \omega_T) \otimes (p, \omega_T) = (p, \omega_T)$
- $(p, e) \otimes (p, \omega_T) = (p, \omega_T)$ where e is an algebraic term of type T. By recurrence:
- $(p, e_1) \otimes \ldots \otimes (p, e_n) \otimes (p, \omega_T) = (p, \omega_T)$ where $e_1, \ldots,$ en are of type T and
- If we have a rewriting rule of the form : $(p, e_1) \otimes \ldots \otimes (p, e_n) \otimes m \rightarrow m'$ if C (m doesn't concern p), then we add rewriting rule :

  $(p, \omega_T) \otimes m \rightarrow (p, \omega_T) \otimes m'$ if C.

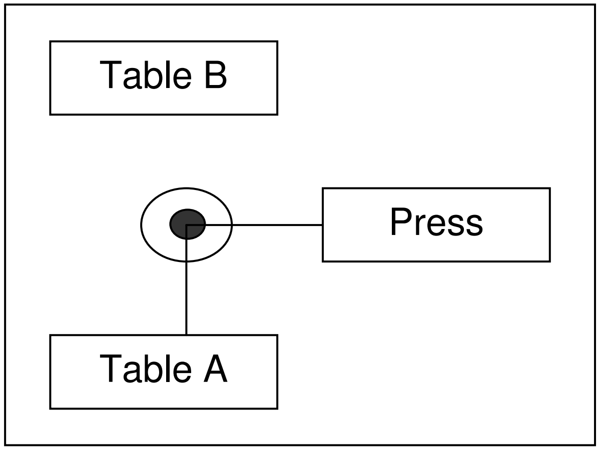
Dynamic Analysis

- The idea of the algorithm is simple :

  We check the path from root (initial marking) until the node of current marking to be calculated :
  - If conditions of unbounded places are true
    - We calculate this marking by covering these places
  - Else
    - We calculate the marking in usual way

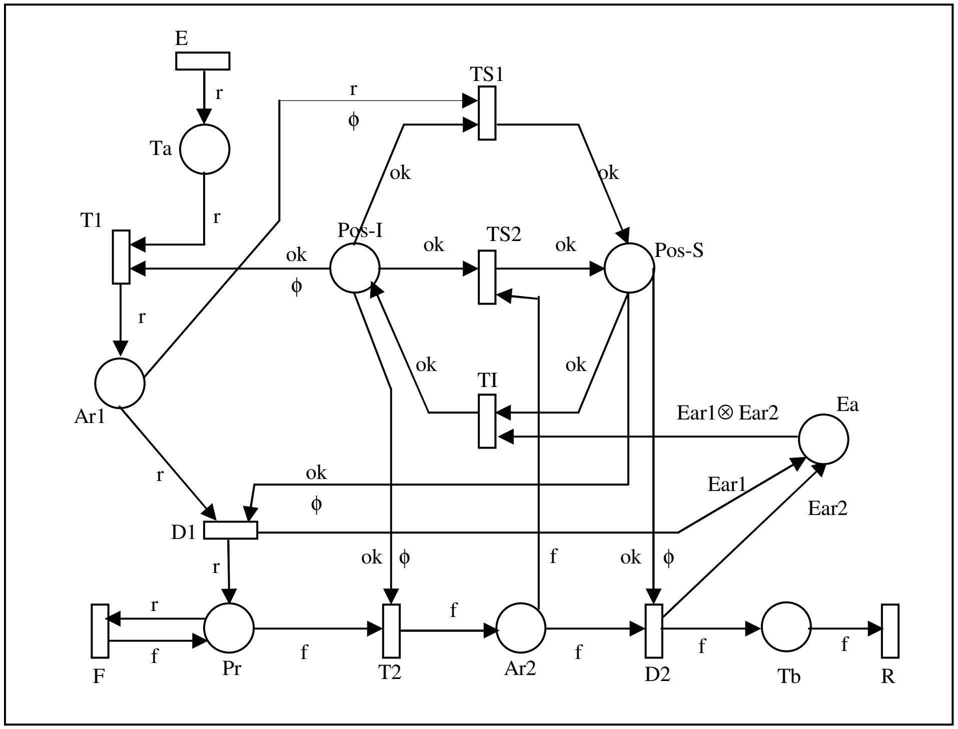
Example : Production Cell

- Cell of production that manufactures forged pieces of metal with the help of a press
- The cell is composed of a table *A* that serves to feed the cell by raw pieces, of a robot of handling, a press and a table *B* that serves to the storage of forged pieces
- The robot includes two arms, disposed at right angles on one same horizontal plan, interdependent of one same axis of rotation and without vertical mobility possibility



Production Cell

Example : ECATNet Modeling Production Cell



•ECATNet Model of the cell

Example : Places

- **ECATNet Places.**
- **Ta** : table A ; set, possibly empty, of raw pieces.
- **Tb** : table B ; set, possibly empty, of raw pieces.
- **Ar1** : arm 1 of robot ; at most a raw piece.
- **Ar2** : arm 2 of robot ; at most a forge piece.
- **Pr** : press ; at most a raw piece or a forge piece.
- **Pos-I** : initial spatial position of robot ; it is marked "ok" if it is the current position of robot.
- **Pos-S** : secondary spatial position of robot ; it is marked "ok" if it is the current position of robot.
- **EA** : this place is added for testing if the tow arms of robot are empty.

Example : Transitions

- **ECATNet Transitions.**
- T1 : Taking of a raw piece by the arm 1 of the robot.
- T2 : Taking of a forge piece by the arm 2 of the robot.
- D1 : deposit of a raw piece in the press.
- D2 : deposit of a forge piece on the table B.
- TS1, TS2 : rotation of the robot from its initial position towards its secondary position.
- TI : rotation of the robot from its secondary position towards its initial position.
- F : forge of the raw piece introduced in the press.
- E : deposit of a raw piece on the table A.
- R : removing forge pieces from the table A.

Example : Rewriting Rules

Rewriting Rules.

$[T1] : (Ta, raw) \rightarrow (Ar1, raw) \quad if\ (M(Pos\text{-}I) \rightarrow ok)$

$and\ ((Ar1, raw) \otimes M(Ar1) \cap C(Ar1)) \rightarrow (Ar1, raw) \otimes M(Ar1)$

$[T2] : (Pr, forge) \rightarrow (Ar2, forge) \quad if\ (M(Pos\text{-}I) \rightarrow ok)$

$and\ ((Ar2, raw) \otimes M(Ar2) \cap C(Ar2)) \rightarrow (Ar2, raw) \otimes M(Ar2)$

$[D1] : (Ar1, raw) \rightarrow (Pr, raw) \otimes (Ea, Ear1) \quad if\ (M(Pos\text{-}S) \rightarrow ok)$

$[D2] : (Ar2, forge) \rightarrow (Tb, forge) \otimes (Ea, Ear2) \quad if\ (M(Pos\text{-}S) \rightarrow ok))$

$[TS1] : (Pos\text{-}I, ok) \rightarrow (Pos\text{-}S, ok)\ if\ (M(Ar1) \rightarrow raw)$

$[TS2] : (Pos\text{-}I, ok) \rightarrow (Pos\text{-}S, ok)\ if\ (M(Ar2) \rightarrow forge)$

$[TI] : (Pos\text{-}S, ok) \otimes (Ea, Ear1) \otimes (Ea, Ear2) \rightarrow (Pos\text{-}I, ok) \quad if\ (M(Pos\text{-}S) \rightarrow ok)$

$[F] : (Pr, raw) \rightarrow (Pr, forge)$

$[E] : \ \phi \rightarrow (Ta, raw)$

$[R] : (Tb, forge) \rightarrow \ \phi$

Example : Reachability Graph

1. $(Pos\text{-}I, ok) \otimes (Ea, Ear2) \xrightarrow{E}$

2. $(Ta, \omega_{Raw}) \otimes (Pos\text{-}I, ok) \otimes (Ea, Ear2) \xrightarrow{T1}$

3. $(Ta, \omega_{Raw}) \otimes (Pos\text{-}I, ok) \otimes (Ar1, r) \otimes (Ea, Ear2) \xrightarrow{TS1}$

4. $(Ta, \omega_{Raw}) \otimes (Pos\text{-}S, ok) \otimes (Ar1, r) \otimes (Ea, Ear2) \xrightarrow{D1}$

5. $(Ta, \omega_{Raw}) \otimes (Pos\text{-}S, ok) \otimes (Pr, r) \otimes (Ea, Ear1) \otimes (Ea, Ear2) \xrightarrow{F}$

6. $(Ta, \omega_{Raw}) \otimes (Pos\text{-}S, ok) \otimes (Ea, Ear1) \otimes (Ea, Ear2) \otimes (Pr, f) \xrightarrow{TI}$

7. $(Ta, \omega_{Raw}) \otimes (Pos\text{-}I, ok) \otimes (Pr, f) \xrightarrow{T1}$

8. $(Ta, \omega_{Raw}) \otimes (Pos\text{-}I, ok) \otimes (Pr, f) \otimes (Ar1, r) \xrightarrow{T2}$

9. $(Ta, \omega_{Raw}) \otimes (Pos\text{-}I, ok) \otimes (Ar2, f) \otimes (Ar1, r) \xrightarrow{TS1, TS2}$

10. $(Ta, \omega_{Raw}) \otimes (Pos\text{-}S, ok) \otimes (Ar2, f) \otimes (Ar1, r) \xrightarrow{D1}$

11. $(Ta, \omega_{Raw}) \otimes (Pos\text{-}S, ok) \otimes (Ar2, f) \otimes (Pr, r) \otimes (Ea, Ear1) \xrightarrow{D2}$

12. $(Ta, \omega_{Raw}) \otimes (Pos\text{-}S, ok) \otimes (Pr, r) \otimes (Ea, Ear1) \otimes (Ea, Ear2) \otimes (Tb, \omega_{Forge})$

Conclusion & future work

- Rewriting logic based-tool for the implementation of such algorithm is under development
- Not Mature
  - Space Reduction
- Extending this algorithm for ECATNets with arcs inhibitor