Timed-pNets Semantic Model

Yanwen Chen
Supervisor: Yixiang Chen Eric Madelaine

ECNU INRIA

11/02/2014

Outline

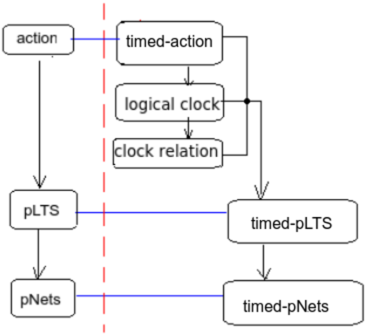
- Motivation
- Basic definition: Logical clock, clock relations, timed specification
- Open/Close timed-pNets and the Compatibility
- Timed-pNets Tree
- Generate timed specification
- Conclusion and future work

Motivation

Extend pNets by adding multi logical clocks and clock relations such that:

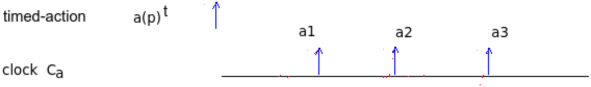
- build a timed model for the distributed systems that have no global physical clock.
- synchronous + asynchronous communications

Our proposal: Timed-pNets

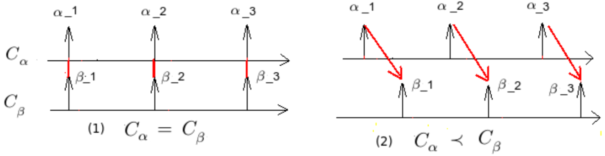


- Logical clocks and clock relations are embedded into timed-pLTS.
- Timed-pNets are built in terms of the communication between timed-pLTSs.
- These communications are represented by clock relations.

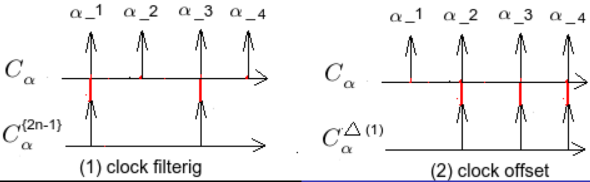
New proposal: logical clock and clock relations



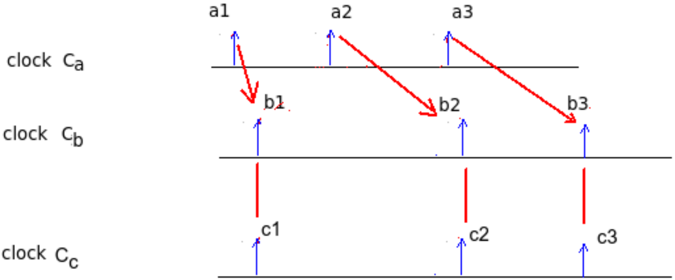
- build logical clock: a sequence of timed-action occurrences
- add constraints between action occurrences



- clock filter and clock offset

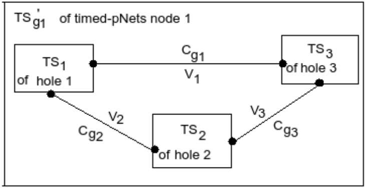


Timed specification: a set of clocks with clock relations.

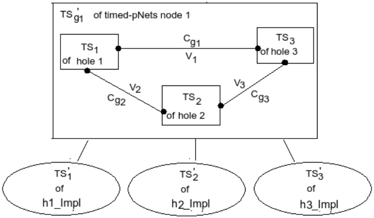


timed specification: TS $= <\{C_a, C_b\ C_c\}, \{C_a < C_b, C_b = C_c \ldots\}>$

Open/Close Timed-pNets node and the compatibility



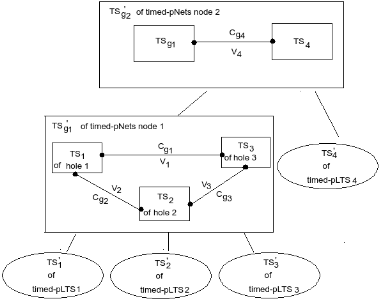
(b) Open Timed-pNet node



(c) Closed Timed-pNets node

Compatibility: $H\_Impl \sqsubseteq H \Leftrightarrow TS \ll TS'$.

Timed-pNets Tree Structure

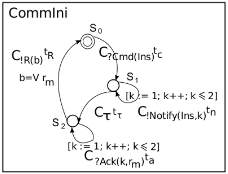


hierarchical model

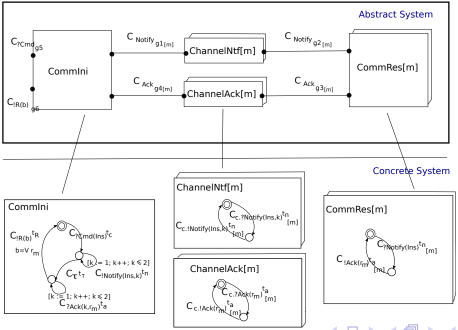
- leaves are timed-pLTS,
- nodes are synchronisation devices
- holes are the timed specifications of the subsystems

Generate Timed Specification

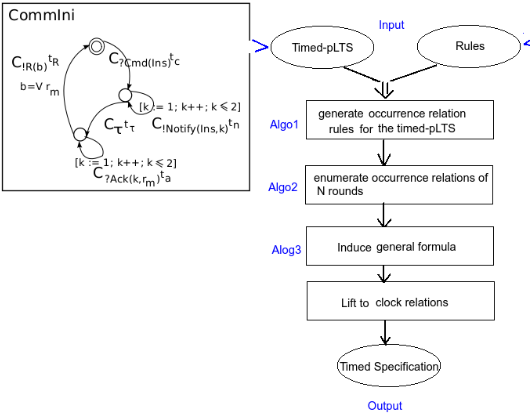
- How to generate TS of a timed-pLTS



- How to generate TS of a timed-pNets node



Procedure of generate TS of a Timed-pLTS

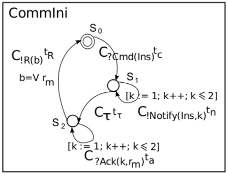


1. If $PreAct(s_0) \notin \emptyset$, then $PreAct(s_0) \prec PostAct(s_0)$,
   [**Assign:** $PostActIndex(s_0) \Leftarrow PreActIndex(s_0) + 1$];
2. $\forall s \backslash s_0$, $PreAct(s) \prec PostAct(s)$,
   [**Assign:** $PostActIndex(s) \Leftarrow PreActIndex(s)$];
3. $\forall s$, if $\exists \alpha . s \xrightarrow{C_\alpha} s$ and the loop executes N times, then
   3.1 go inside the self-loop
      3.1.1 from one preceding action occurrence:
      $PreAct(s) \prec \alpha\_i$,

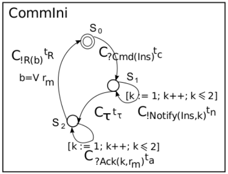
$\vdots$

Algo1: generate concrete rules for a timed-pLTS

- Analysis causality relations on clock occurrences
- set the assignment functions according to the general rules



Transition	Action Occurrence Relations	Index Assignment
$s_2 \xrightarrow{C_{!R}} s_0 \xrightarrow{C_{?Cmd}} s_1$	$!R\_m \prec ?Cmd\_n$	$f(n) : n = m + 1$
$s_0 \xrightarrow{C_{?Cmd}} s_1 \xrightarrow{C_{\tau}} s_2$	$?Cmd\_n \prec \tau\_r$	$f(r) : r := n$
$s_0 \xrightarrow{C_{?Cmd}} s_1 \xrightarrow{C_{!Notify}} s_1$	$?Cmd\_n \prec !notify\_i$	$i := i + 1$
$s_1 \xrightarrow{C_{!Notify}} s_1 \xrightarrow{C_{!Notify}} s_1$	$!notify\_i \prec !notify\_(i + 1)$	
$s_1 \xrightarrow{C_{!Notify}} s_1 \xrightarrow{C_{\tau}} s_2$	$!notify\_i \prec \tau\_r$	$f(r) : r = n$
s_2	$\tau\_r \prec ?ack\_j$	$f(j) : j = j + 1$
	$?ack\_j \prec ?ack\_(j + 1)$	
	$?ack\_j \prec !R\_m$	$f(m) : m = j;$



(d) pLTS

Transition	Action Occurrence Relations	Index Assignment
$s_2 \xrightarrow{C_{!R}} s_0 \xrightarrow{C_{?Cmd}} s_1$	$!R\_m \prec ?Cmd\_n$	$f(n) : n = m+1$
$s_0 \xrightarrow{C_{?Cmd}} s_1 \xrightarrow{C_{\tau}} s_2$	$?Cmd\_n \prec \ \tau\_r$	$f(r) : r := n$
$s_0 \xrightarrow{C_{?Cmd}} s_1 \xrightarrow{C_{!Notify}} s_1$	$?Cmd\_n \prec !notify\_i$	$i := i+1$
$s_1 \xrightarrow{C_{!Notify}} s_1 \xrightarrow{C_{!Notify}} s_1$	$!notify\_i \prec !notify\_(i+1)$	
$s_1 \xrightarrow{C_{!Notify}} s_1 \xrightarrow{C_{\tau}} s_2$	$!notify\_i \prec \tau\_r$	$f(r) : r = n$
s_2	$\tau\_r \prec ?ack\_j$	$f(j) : j = j+1$
	$?ack\_j \prec ?ack\_(j+1)$	
	$?ack\_j \prec !R\_m$	$f(m) : m = j;$

(e) rules

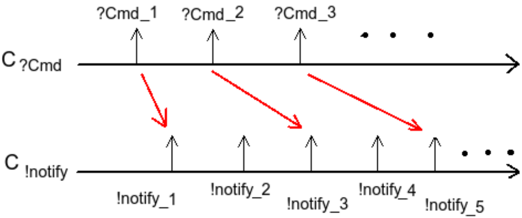
- Initial value: m,n,r,i,j := 0; k := 1
- reset k:=1 when going out self-loop

1^{st} round	2^{nd} round	3^{th} round	...
$!R\_0 \prec ?Cmd\_1$	$!R\_1 \prec ?Cmd\_2$	$!R\_2 \prec ?Cmd\_3$	...
$?Cmd\_1 \prec !notify\_1$	$?Cmd\_2 \prec !notify\_3$	$?Cmd\_3 \prec !notify\_5$	...
$!notify\_1 \prec !notify\_2$	$!notify\_3 \prec !notify\_4$	$!notify\_5 \prec !notify\_6$	...
$!notify\_2 \prec \tau\_1$	$!notify\_4 \prec \tau\_2$	$!notify\_6 \prec \tau\_3$	...
...			
$?ack\_2 \prec !R\_1$	$?ack\_4 \prec !R\_2$	$?ack\_6 \prec !R\_3$	...

Algo3:Induce a general formula

- task: given a set of natural numbers, how to induce its general formula?

1^{st} round	2^{nd} round	3^{th} round	...	s^{th} round
$!R\_0 \prec ?Cmd\_1$	$!R\_1 \prec ?Cmd\_2$	$!R\_2 \prec ?Cmd\_3$	...	$!R\_(s-1) \prec ?Cmd\_s$
$?Cmd\_1 \prec !notify\_1$	$?Cmd\_2 \prec !notify\_3$	$?Cmd\_3 \prec !notify\_5$	...	$?Cmd\_s \prec !notify\_(2s-1)$
$!notify\_1 \prec !notify\_2$	$!notify\_3 \prec !notify\_4$	$!notify\_5 \prec !notify\_6$	...	$!notify\_(2s-1) \prec !notify\_2s$
$!notify\_2 \prec \tau\_1$	$!notify\_4 \prec \tau\_2$	$!notify\_6 \prec \tau\_3$	...	$!notify\_2s \prec \tau\_s$
$\tau\_1 \prec ?ack\_1$	$\tau\_2 \prec ?ack\_3$	$\tau\_3 \prec ?ack\_5$	...	$\tau\_s \prec ?ack\_(2s-1)$
$?ack\_1 \prec ?ack\_2$	$?ack\_3 \prec ?ack\_4$	$?ack\_5 \prec ?ack\_6$	...	$?ack\_(2s-1) \prec ?ack\_2s$
$?ack\_2 \prec !R\_1$	$?ack\_4 \prec !R\_2$	$?ack\_6 \prec !R\_3$	...	$?ack\_2s \prec !R\_s$

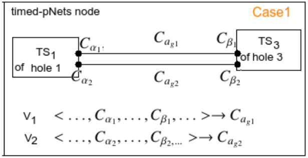


We proof: $C_\alpha^{\{P(i)\}} \prec C_\beta^{\{P'(i)\}} \Leftrightarrow \forall i \in \mathbb{N}, \alpha\_(P(i)) \prec \beta\_(P'(i))$
so that we have :

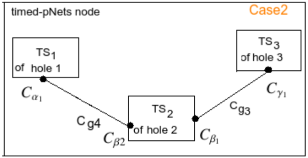
- $C_\alpha \prec C_\beta^{\{2i-1\}} \Leftrightarrow \forall i \in \mathbb{N}, \alpha\_i \prec \beta\_(2i-1)$
- $C_\alpha^{\{2i-1\}} \prec C_\beta^{\{2i\}} \Leftrightarrow \forall i \in \mathbb{N}, \alpha\_(2i-1) \prec \beta\_(2i)$

1^{st} round	2^{nd} round	...	s^{th} round		Clock Relations
$!R\_0 \prec ?Cmd\_1$	$!R\_1 \prec ?Cmd\_2$	...	$!R\_(s-1) \prec ?Cmd\_s$		$C_{!R} \prec C_{?Cmd}^{\Delta(1)}$
$?Cmd\_1 \prec !notify\_1$	$?Cmd\_2 \prec !notify\_3$	...	$?Cmd\_s \prec !notify\_(2s-1)$		$C_{?Cmd} \prec C_{!notif}^{\{2s-1}$
$!notify\_1 \prec !notify\_2$	$!notify\_3 \prec !notify\_4$	...	$!notify\_(2s-1) \prec !notify\_2s$	$\Rightarrow$	$C_{!notify}^{\{2s-1\}} \prec C_{!notif}^{\{2s\}}$
$!notify\_2 \prec \tau\_1$	$!notify\_4 \prec \tau\_2$	...	$!notify\_2s \prec \tau\_s$		$C_{!notify}^{\{2s\}} \prec C_\tau$
$\tau\_1 \prec ?ack\_1$	$\tau\_2 \prec ?ack\_3$	...	$\tau\_s \prec ?ack\_(2s-1)$		$C_\tau \prec C_{?ack}^{\{2s-1\}}$
$?ack\_1 \prec ?ack\_2$	$?ack\_3 \prec ?ack\_4$	...	$?ack\_(2s-1) \prec ?ack\_2s$		$C_{?ack}^{\{2s-1\}} \prec C_{?ack}^{\{2s\}}$
$?ack\_2 \prec !R\_1$	$?ack\_4 \prec !R\_2$	...	$?ack\_2s \prec !R\_s$		$C_{?ack}^{\{2s\}} \prec C_{!R}$

generate TS of a Timed-pNets node



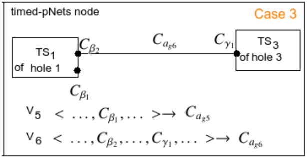
- $C_{a_{g1}} = C_{a_{g1}}$ if and only if $C_{\alpha_1} = C_{\alpha_2} \wedge C_{\beta_1} = C_{\beta_2}$.
- $C_{a_{g1}} < C_{a_{g1}}$ if and only if $C_{\alpha_1} < C_{\alpha_2} \wedge C_{\beta_1} < C_{\beta_2}$.



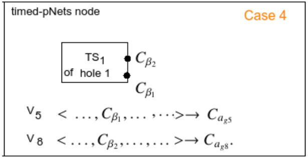
V 3 $<\ldots, C_{\beta_1}, \ldots, C_{\gamma_1} > \rightarrow C_{a_{g3}}$

V 4 $< C_{\alpha_1}, C_{\beta 2}, \ldots, \ldots > \rightarrow C_{a_{g4}}$

- $C_{a_{g3}} = C_{a_{g4}}$ if and only if $C_{\beta_1} = C_{\beta_2}$,
- $C_{a_{g3}} < C_{a_{g4}}$ if and only if $C_{\beta_1} < C_{\beta_2}$.

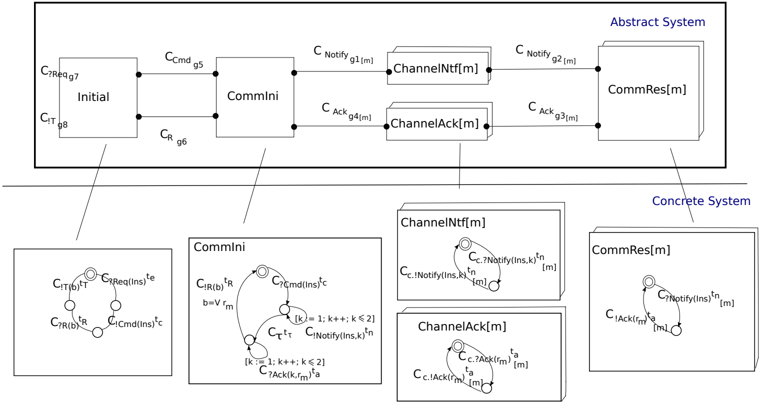


- $C_{a_{g5}} = C_{a_{g6}}$ if and only if $C_{\beta_1} = C_{\beta_2}$,
- $C_{a_{g5}} < C_{a_{g6}}$ if and only if $C_{\beta_1} < C_{\beta_2}$.



- $C_{a_{g5}} = C_{a_{g8}}$ if and only if $C_{\beta_1} = C_{\beta_2}$,
- $C_{a_{g5}} < C_{a_{g6}}$ if and only if $C_{\beta_1} < C_{\beta_2}$.

Example



- $C_{Notify_{g1}} \boxed{?} C_{Notify_{g2}}$: Case 2 $\Rightarrow C_{Notify_{g1}} \prec C_{Notify_{g2}}$
- $C_{Notify_{g1}} \boxed{?} C_{Ack_{g3}}$
- $C_{Cmd_{g5}} \boxed{?} C_{R_{g6}}$: Case1 $\Rightarrow C_{Cmd_{g5}} \prec C_{R_{g6}}$
- $C_{?Req_{g7}}$ … $C_{Cmd_{g5}} \boxed{?} C_{?Req_{g7}}$: Case 3 $\Rightarrow C_{?Req_{g7}} \prec C_{Cmd_{g5}}$
- $C_{?Req_{g7}} \boxed{?} C_{!T_{g8}}$: Case 4 $C_{?Req_{g7}} \prec C_{!T_{g8}}$ ....

Conclusion

- Our model is flexible—logical clock, without physical common clock
- Flexible to compose by adding relations of clocks in TS

future work

- Current work
  - Prove the completeness of the 4 cases to generating TS of timed-pNets
  - Detect wrong composition by checking the relation conflicts
- Future work
  - Extend to duration-pNets so that we can specify the actions that include execution time

Reference

- E.A. Lee. Cyber physical systems: Design challenges.
- J. Bengtsson, K. G. Larsen, F. Larsson, P. Pettersson, and Wang Yi. Uppaal a Tool Suite for Automatic Verification of Real Time Systems.
- T. Barros, R. Ameur-Boulifa, A. Cansado, L. Henrio, and E. Madelaine. Behavioural models for distributed fractal components.
- Frederic Mallet. CCSL: specifying clock constraints with UML/MARTE.
- Julien Deantoni and Frederic Mallet. TimeSquare: Treat your Models with Logical Time.

Questions