

PETRI-NETS AND OBJECT-ORIENTED APPROACH FOR THE ANALYSIS OF HYBRID SYSTEMS

EMILIA VILLANI, PAULO EGI MIYAGI

*Escola Politecnica da Universidade de São Paulo
Av. Prof. Mello Moraes, 2231 São Paulo, Brasil
E-mails: evillani@usp.br, pemiyagi@usp.br*

ROBERT VALETTE

*Laboratoire d'Analyse et d'Architecture des Systèmes - CNRS
7, Avenue du Colonel Roche, 31077 Toulouse Cedex 4, France
E-mail: robert@laas.fr*

Abstract The behaviour analysis of industrial processes represented by hybrid models is still an unsolved problem when simulation can not be used. In this context, this paper presents a new approach for the verification of behaviour properties of hybrid systems. By using Petri nets and object oriented concepts the verification of a system property is reduced from a complex proof involving the overall model to a set of simpler proofs involving the model of one or a few objects. Each proof may generate a list of other local proof obligations to be carried out in other object nets. The internal evolution of each object is considered independently from the other object evolutions and the linear logic is used as a formalism to determine the partial order of transition firing. In order to illustrate the proposed approach, a sugar production process is used as an example.

Keywords hybrid systems, analysis, Petri nets, object orientation, linear logic.

1 Introduction

In accordance to recent works (Antsaklis & Koutsoukos, 1998), the use of hybrid models (where discrete events and continuous dynamics are both present) for representing industrial process is becoming more and more frequent. The development of techniques for studying system hybrid behaviour has been a constant topic of research. The most general approach for is simulation. However, simulation shows just one of the possible system evolutions. If the initial condition is not completely known or if the model is not deterministic, simulation can not be used for guaranteeing the absence of errors in system behaviour. Regarding formal approaches, most of the proposed solutions for the verification of system behaviour properties are based on automatas (Gueguen & Zaytoon, 2001). For complex hybrid system based on Petri nets, these approaches can lead to an explosion of the number of states, making impracticable its use.

The main problem related to the verification of behaviour properties for hybrid system is the non decidability, i.e., the non-guarantee that, with a finite number of steps the property can be proved. The decidability problem was first considered for discrete event models associated with time-dense clocks, where time is a continuous variable. In (Alur & Dill, 1994) it was shown that for the timed-automata the properties are still decidable. In the same work, the region graphs were introduced as an analysis tool associating discrete states with intervals for the continuous variables representing the system clocks.

The accessible states are then represented by a finite number of regions, allowing property verifications.

On the other hand, it was also proved that if continuous variables with different growing rates (different derivatives) are included in the model, then the number of regions may become infinite. Generally, this is the case of hybrid systems.

Although decidability can not be guaranteed for hybrid system, some properties can still be verified according to the system characteristics. In this sense, this work investigates possible techniques to simplify the analysis problem of large complex hybrid systems. By the use of Petri nets, object oriented concepts and linear logic, the verification of a system property which would otherwise involves the overall model is divided into a set of local analysis problems involving the model of one or a few objects.

This paper is organised as following. Section 2 presents a brief summary about the association of object-oriented concepts with Differential Predicate Transition Petri net (DPTP net). Section 3 introduces the main points of the analysis approach. The approach is then illustrated in Section 4 using as an example a sugar cane production process. Finally, Section 5 draws some conclusions.

2 Hybrid System Modelling

Briefly, a DPTP net defines an interface between differential equation systems and Petri net elements. Its main features are (Champagnat et al, 1998):

- A set of formal variables (V) is defined for the net;
- Each token is associated with a tuple of formal variables (X_i), which is a subset of V .

- Each place (P_i) is associated with a differential equation system (F_i) that defines the value of the X_i associated to the tokens in P_i , according to the time (θ):

$$F_i(\dot{X}_i, X_i, \theta) = \begin{bmatrix} f_{i-1}(\dot{X}_i, X_i, \theta) = 0 \\ \vdots \\ f_{i-n}(\dot{X}_i, X_i, \theta) = 0 \end{bmatrix}$$

- Each transition (T_i) is associated with an enabling function (e_i) that enables the transition firing according to the value of the X attached to the token of the input places (X_{input_i}).

$$e_i(X_{input_i}, \theta) = 0$$

- Each transition (T_i) is associated with a junction function (j_i) that defines the value X_i attached to the token of the output places (X_{output_i}) after the transition firing:

$$X_{output_i}(\theta^+) = j_i(X_{input_i}(\theta))$$

The introduction of the object-oriented concepts is treated in detail in (Villani et al, 2001). The following statements are defined based on class and object concepts of (Booch, 1994):

- The behaviour of a class A is modelled by a DPTP net (N_A).
- The attributes of the class A is modelled as the set of variables of the DPTP net (V_A).
- The first variable of a token tuple of variables in a class net is the identity of an object.
- An object is represented by a token in the class net, or by a set of tokens with the same identity.

The communication among objects can be discrete or continuous. The discrete interactions are represented by method calls (Paludetto, 1991). The continuous interactions are modelled by sharing continuous variables among objects. The value of the shared variables is determined by an object and could be used in the junction function, the equation systems or the enabling function of other objects.

3 Analysis

3.1 Overview of the proposed approach

One of the advantages of the DPTP net is that the discrete part of the model and the continuous one are clearly identified. This means that it is still possible to analyse each part independently from the other. The Petri net analysis tools could then be used for the discrete part as well as any differential equation analysis tool for the continuous one.

As most of the Petri net analysis tools are developed for ordinary Petri nets, it is necessary to double the class net the number of times of the class objects, in order to not consider any kind of identity for the tokens. Each copy is assigned to an object.

The proposed approach is based on the division of a property verification problem (a *proof* from the logical point of view) into a set of local analysis problems (set of proofs) including only one or a few

objects. Each proof (1st level proof) can then result in the obligation of a new set of proofs (2nd level proofs) in other objects. This means that the property will be true in the first object if new proofs (2nd level proofs) are also true in the other objects. The process goes on until it remains no other proof to be done.

In some cases the proof could not be divided into a set of proofs each one in a single object. If this is the case, the net of the set of objects implied in the proof are merged and the resulted net is analysed as a single object. An example of a proof that cannot be divided is that of liveness of a set of objects.

Proof obligation could be the result of the following situations:

- the proof includes shared variables in the enabling function, junction function or equation system;
- the proof includes transitions that are methods of the object interface (the method should be called by other objects in order to fire the transition or the other object must answer the method call in order to fire the transition).

Another important point is how to interleaves the discrete and continuous analysis. A first proposal would be to find all possible solutions considering only the discrete part of the system and then, for each solution verify if it is still valid when considering the continuous part. Nevertheless, this kind of approach could result in an impracticable number of discrete solutions to be analysed. In order to avoid it each discrete state evolution takes into account the correspondent continuous variables evolution.

3.2 Linear Logic as an aid for Petri net analysis.

The equivalence between Petri net reachability and the proof of a set of sequents in linear logic can be used in two different cases. In the first case, the initial marking (M_0) and final marking (M_f) is known. By solving the equation $M_f = M_0 + C * s$, the possible sets of transition firings 's' that lead to M_f from M_0 are found. Then by using linear logic it is possible to determine if the solution is a feasible one and what is the partial order of transition firings. This is the classical reachability issue. The benefit of linear logic is that instead of a firing sequence a partial order is derived. The partial order highlights the causality among the transition firings. For example, in two parallel process, the partial order shows the order of an internal event of the first process in respect to the order events of that process without considering the order in respect to the internal events of the second process (global or temporal sequence). When two transition firings are concurrent, they are logically independent and when they belong to two different objects, it is possible to exploit object autonomy.

In the second case, only the final marking is known and the scope is to determine a set of initial markings and transition firing partial order that could give rise to the final marking. This could be useful for

safety properties. In both cases, the analysis of the continuous part is then interleaved with the use of linear logic in order to provide an hybrid approach.

The equivalence between linear logic and Petri nets is defined based on the following statements (detailed introductions and examples could be found in (Girault et al., 1997)):

- An atomic proposition “p” is associated with each place “P” of the net and it represents the presence of one token in this place.
- Any marking is represented by a conjunctive formula (using the connector \otimes) involving the marked places. Example: $p_1 \otimes p_2 \otimes p_2$ means one token in P_1 and two tokens in P_2 .
- An implicative formula is defined for each transition (using the connector \multimap), expressing causality between two marking formulas. The left side of \multimap indicates the tokens that are consumed during the transition firing and the right side indicates the tokens that are produced:

$$t: \otimes p_i (i \in \text{Pre}(P_i, t)) \multimap \otimes p_j (j \in \text{Post}(P_j, t)) \quad (1)$$

In linear logic, a sequent is logic expression represented as “ $? \vdash G$ ” and it means that the second part of the sequent is a logic consequence of the first part. The sequent $M_0, t_1, \dots, t_n \vdash M_1$ expresses the reachability of M_1 from M_0 by means of the transition firing of t_1, \dots, t_n . The proof of the sequent is the proof of the reachability for a partial order over t_1, \dots, t_n . This is done using the following rules, where F, G, H, Γ and Δ are formulas or set of formulas:

$$\frac{G, F, G \vdash H}{G, F \otimes G \vdash H} \otimes L \quad (2) \quad \frac{G \vdash F \quad ?, G \vdash H}{G, ?, F \multimap G \vdash H} \multimap L \quad (3)$$

Each rule means that if the upper part of the rule (the sequents over the line) is true, the lower part is true. The rules should be applied in order to obtain identity sequents ($A \vdash A$) in the upper part, which is an identity sequent ($\vdash Id$) and is true by definition. The application of the rule “ $\multimap L$ ” corresponds to the firing of a transition. The application of the rule “ $\otimes L$ ” corresponds to the transformation of a marking into a list of independent atoms (tokens which are not necessarily simultaneously present at the same time point). In a sequent, the exponential connector “ $!$ ” means that t_i can be fired from zero to infinite times.

4 Example

4.1 The system

The example used to illustrate the proposed approach is part of the sugar production process (Figure 1).

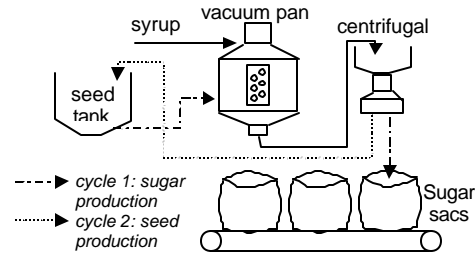


Figure 1. Sugar production process.

The syrup resulting from the evaporation process goes to the vacuum pan, where it could be used to produce sugar (cycle 1 of the vacuum pan) or to produce seeds (cycle 2 of the vacuum pan).

During cycle 1 the syrup is further concentrated by boiling. It is then seeded with small sugar crystals, which are grown to the required size by adding more syrup while boiling continues. When the crystals reach the required size, the mixture of syrup and crystals, called massecuite, is discharged from the pan and sent to the centrifugal. In cycle 2 the syrup is left in the vacuum pan for a longer time, until it gives rise to small sugar crystals. As in cycle 1, the resulted massecuite is then sent to the centrifugal, originating seeds that are sent to the Seeds Tank.

This system can be modelled by a set of 4 classes: *Seed_Tank*, *Vacuum_Pan*, *Centrifugal* and *Sugar_Sac_Filler*. As a general rule all the method calls are considered as synchronous and are modelled as a transition merging (Paludetto, 1991). The DPTP net model of each class is presented as following.

The graphical elements of Figure 2 are adopted to represent an internal transition of the class, an interface transition (that represents a method offered by the class) and a method call (i.e. a transition that is merged with an interface transition of another class).

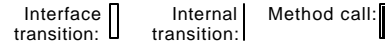


Figure 2. Graphical representation of the transitions.

Class *Seed_Tank*

In the model of the class *Seed_Tank* (Figure 3), the volume of the tank (V_{se}) is changed continuously by the equation associated with P_3 , or instantaneously by the junction function of t_2 . The method associated with t_1 is used by the class *Centrifugal* to deliver the seeds resulting from centrifugation (a discrete event). K_1 is the amount of seed resulting after the centrifugation. The methods associated to t_3 and t_4 are used by the *Vacuum_Pan* to get the seeds during the sugar production cycle (a continuous activity).

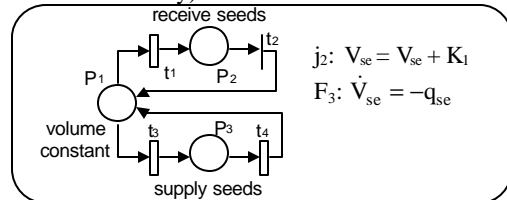


Figure 3. DPTP net for the class *Seed_Tank*.

Class Vacuum pan

In the *Vacuum_pan* class (Figure 4), during the 'Loading syrup' phase the syrup enters in the pan with a fixed rate q_{sy} . After that, if there is enough seeds on the *Seed_Tank*, then sugar production is enabled, else the seed production begins. During the 'Concentration' phase, the syrup is boiled. The water evaporated is replaced by more syrup, maintaining the syrup level in the pan constant. 'Concentration' stops when the Brix ($100 \cdot \text{kg}$ of sucrose/ kg of syrup) of the syrup reaches a fixed value. During the 'Seeding', seeds and water are added to the pan. The 'Evaporation' phase is similar to the 'Concentration', but with no syrup loading. The masseccuite, is sent to the *Centrifugal*.

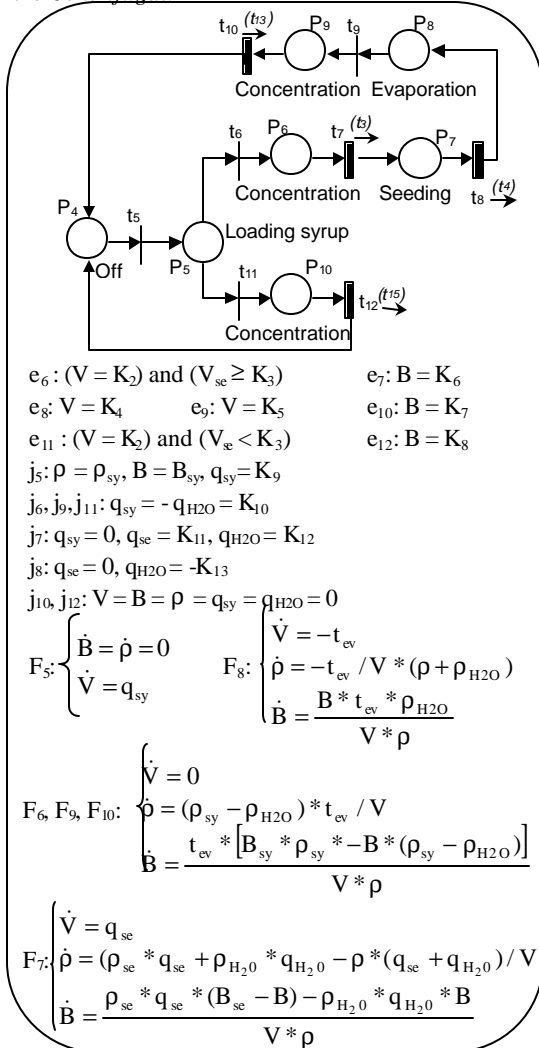


Figure 4. DPTP net for the class *Vacuum_Pan*.

In the equations of Figure 4:

- V, B, ρ are the volume, brix and density of the mixture (class variables)
- q_{se}, q_{sy}, q_{H2O} are the flow of seeds, syrup and water into/out of the pan (class variables)
- V_{se} is the *Seed_Tank* volume (external variable)
- B_{sy}, B_{se} are the brix of the syrup and seed supplied to the pan (constants)

- $\rho_{sy}, \rho_{se}, \rho_{H2O}$ are the density of the syrup and seed supplied to the pan (constants)
- $K_2, K_3, K_4, K_5, K_6, K_7, K_8, K_9, K_{10}, K_{11}, K_{12}, K_{13}$ are process parameters (constants)

Class: Centrifugal

The model of the class *Centrifugal* is presented in Figure 5. The masseccuite stays in the centrifugal for a fixed time (θ_{su} or θ_{se}) and then is delivered to the *Sugar_Sac_Filler* or to the *Seed_Tank*. The receiving of masseccuite and the delivering of sugar or seeds are modelled as discrete events.

Class: Sugar_Sac_Filler

The model of the class *Sugar_Sac_Filler* is presented in Figure 6. Each time the centrifugal delivers sugar, a sac is filled of about 50%. When the sac is full it is automatically delivered and a new sac coming in (t_{19}) No continuous dynamic is associated with this class.

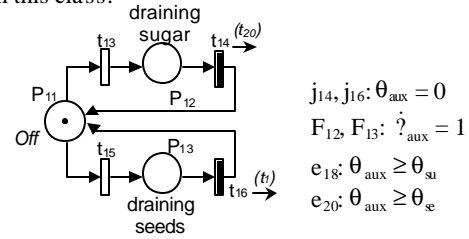


Figure 5. DPTP net for the class *Centrifugal*.

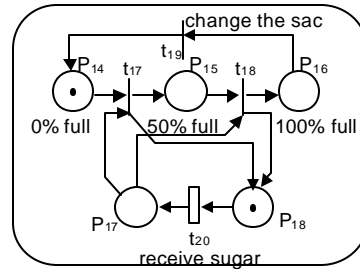


Figure 6. DPTP net for the class *Sugar_Sac_Filler*.

4.2 Analysis of the example

The property that must be verified is the reachability of the state P_{16}, P_{18} (100% full sugar sac) from the initial state P_{14}, P_{18} (empty sugar sac). The following considerations are made:

- The *Seed_Tank* is in the state P_1 and the tank level is unknown;
- The *Vacuum_Pan* and the *Centrifugal* are 'Off';
- $K_1=6, K_2=10, K_3=3, K_4=16, K_5=12, K_6=80, K_7=90, K_8=90, K_9=2.5, K_{10}=1, K_{11}=0.5, K_{12}=1, K_{13}=1; B_{sy}=65, B_{se}=90, \rho_{sy}=2, \rho_{H2O}=1, \rho_{se}=3.25, \theta_{su}=5; \theta_{se}=4;$

As the property concerns only the state of object *Sugar_Sac_Filler*, it is the first to be analysed.

Analysis of Sugar_Sac_Filler

By solving the equation $M = M_0 + C * s$ the possible sets of transition firings that lead to $M_f(P_{16}, P_{18})$ from $M_b(P_{14}, P_{18})$ could be found. The result is the following set of equations, where s_i means the number of firings of transition i :

$$\begin{cases} s_{18} = s_{17} \\ s_{18} = s_{19} + 1 \\ s_{20} = 2 * s_{18} \end{cases}$$

It is supposed that the final marking is not part of the intermediate markings and, therefore, $s_{19}=0$. As a consequence $s_{18} = s_{17} = 1 \text{ e } s_{20}=2$.

The linear logic is then used in order to find the transition firing sequence, which corresponds to proving the sequent $p_{14}, p_{18}, t_{17}, t_{18}, t_{20} \multimap p_{16}, p_{18}$.

The following formulas are related to the *Sugar_Sac_Filler* class net:

- $t_{17}: p_{14} \otimes p_{17} \multimap p_{15} \otimes p_{18}$
- $t_{18}: p_{15} \otimes p_{17} \multimap p_{16} \otimes p_{18}$
- $t_{19}: p_{16} \multimap p_{14}$
- $t_{20}: p_{18} \multimap p_{17}$

From the initial marking, the first enabled transition is t_{20} . Its firing corresponds to the application of the rule $\multimap L$:

$$\begin{array}{c} \text{----- Id} \\ p_{18} \multimap p_{18} \quad p_{14}, p_{17}, t_{17}, t_{18}, t_{20} \multimap p_{16}, p_{18} \\ \text{-----} \multimap L \\ (t_{20}) \end{array}$$

The next transition to be fired is t_{17} :

$$\begin{array}{c} p_{14} \otimes p_{17} \multimap p_{14}, p_{17} \quad p_{15}, p_{18}, t_{28}, t_{20} \multimap p_{16}, p_{18} \\ \text{-----} \multimap L \\ (t_{17}) \end{array}$$

In order to eliminate the sequent on the left side, the rule $\otimes L$ is applied:

$$\begin{array}{c} \text{----- Id} \\ p_{14}, p_{17} \multimap p_{14}, p_{17} \\ \text{-----} \otimes L \\ p_{14} \otimes p_{17} \multimap p_{14}, p_{17} \end{array}$$

The reasoning continues until all the transitions are fired and the sequent is proved. The partial order of transition firing resulted is “ $t_{20}; t_{17}; t_{20}; t_{18}$ ” (in this case it is also a total order). As t_{20} is associated to a method call, it results in a proof obligation for the object *Centrifugal*, which fires t_{20} by its transition t_{14} .

Analysis of Centrifugal

The property to be verified is the viability of firing t_{14} twice from the initial state (P_{11}). The same approach is applied for this object. In this case, the final state is the state resulted from the second firing of transition t_{14} (also P_{11}).

By solving the equation ‘ $M_f = M_0 + C*s$ ’ and considering $s_{14}=2$, the sequent to be proved is determined: $p_{11}, t_{14}, t_{14}, t_{13}, t_{13}, (t_{15}, t_{16})^+ \multimap p_{11}$.

During the sequent proof, when two transitions could be fired, two different scenarios should be defined. Furthermore, the evolution of continuous variables should be calculated in order to verify the enabling functions and define the time interval constrains between the transition firings.

From the initial sequent two transitions could be fired: t_{13} and t_{15} . When t_{13} is fired, the next transition is t_{14} . In order to satisfy the enabling function of t_{14} , the evolution of θ_{\max} should be calculated, resulting in the time interval constrain $\theta(t_{14}) = \theta(t_{13}) + \theta_{su}$ (where $\theta(t_i)$ is the firing time of ϑ). Similarly, the other scenario

results in the firing of t_{15} and t_{16} , and in the time interval constrain $\theta(t_{16}) = \theta(t_{15}) + \theta_{se}$.

The order of transition firing resulted from the sequent proof is: “ $n_1*(t_{15}; t_{16}); t_{13}; t_{14}; n_2*(t_{15}; t_{16}); t_{13}; t_{14}$ ”, where n_1 and n_2 could be any number from zero to infinite. The infinite number of scenarios is a consequence of the fact that there is no restriction for the number of seed centrifugation cycles before and between the two sugar centrifugation cycles. These scenarios generate proof obligations for two objects: the *Vacuum_Pan*, which call the methods associated to t_3 and t_5 , and the *Seed_Tank*, which has the method called by t_6 . As it is impossible to analyse infinite scenarios, the following ones are considered:

Scenario 1: $t_{13}; t_{14}; t_{13}; t_{14}$

Scenario 2: $t_{15}; t_{16}; t_{13}; t_{14}; t_{13}; t_{14}$

Scenario 3: $t_{13}; t_{14}; t_{15}; t_{16}; t_{13}; t_{14}$

Scenario 4: $t_{15}; t_{16}; t_{15}; t_{16}; t_{13}; t_{14}; t_{13}; t_{14}$

Particularly, the analysis of Scenario 1 is illustrated in the following.

Analysis of Vacuum_Pan

For *Scenario 1* the property to be verified is if transition t_{10} (associated with t_{13}) could be fired twice with a minimum interval of θ_{su} and without firing the transition t_{12} (associated with t_{15}). The initial state is P_4 . As for the *Centrifugal*, the final state is the state after the second firing of t_{15} (P_4).

In this case the sequent to be proved is: $p_4, t_5, t_5, t_6, t_6, t_7, t_7, t_8, t_8, t_9, t_9, t_{10}, t_{10} \multimap p_{11}$. The proof results in the following order for the transition firing: $t_5; t_6; t_7; t_8; t_9; t_{10}; t_5; t_6; t_7; t_8; t_9; t_{10}$. As for the *Centrifugal*, the calculus of the continuous variable evolution should also be interleaved with the transition firing in the linear logic proof. The result is presented in Figure 7. It is possible to verify that the interval between the two firings of t_{10} is larger than the minimum interval of θ_{su} .

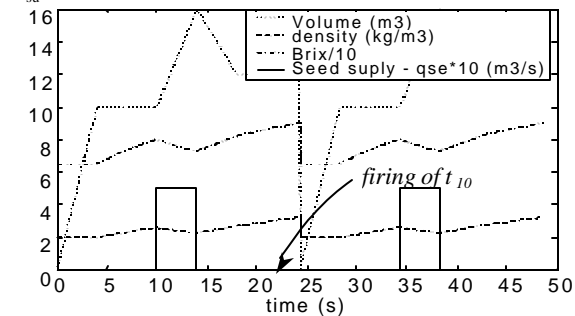


Figure 7. Continuous variable evolution for *Scenario 1*.

The analysis of this object results in the following proof obligations in the object *Seed_Tank*:

- $V_{se} > K_3$ in $\theta=4$ and $\theta=28.32$,
- Firing of the transition associated with t_7 in $\theta=10$ and $\theta=34.32$, and of that associated with t_8 in $\theta=14$ and $\theta=38.32$.

Similarly, the same approach is carried out for the others scenarios under analysis.

Analysis of Seed_Tank

For *Scenario 1*, the property to be verified is the viability of the transition firing sequence of $t_3; t_4; t_3; t_4$ with the following time constraints (where $\theta_{i,j}$ is the time of the i^{th} firing of transition t_j):

- $\theta_{i4,i} = \theta_{i3,i} + 4;$
- $\theta_{i3,2} = \theta_{i4,1} + 20.32;$

Furthermore, the constraint $V_{se} > K_3$ should be verified in $\theta = \theta_{i3,1} - 6$ and at $\theta = \theta_{i3,2} - 6$.

The initial and the final state is p_1 , resulting in the following sequent: $p_1, t_3, t_4, t_3, t_4 + - p_1$.

By calculating the evolution of V_{se} , the following constrains are found ($V_{se,0}$ is the initial value of V_{se}):

$$V_{se}(\theta_{i3,1}-6) = V_{se}(\theta_{i3,1}) = V_{se,0} \geq 3$$

$$V_{se}(\theta_{i3,2}-6) = V_{se}(\theta_{i3,2}) = V_{se}(\theta_{i4,1}) = V_{se,0} - 2 \geq 3$$

As a consequence, the Scenario 1 is a valid if and only if ' $V_{se,0} \geq 5$ '.

Similarly, the analysis of Scenario 2 leads to the constrains ' $V_{se,0} < 3$ ', ' $V_{se,0} + 6 \geq 3$ ' and ' $V_{se,0} + 6 - 2 \geq 3$ '. As the level of seeds in the tank could not be negative, the resulting in the constraint is ' $0 < V_{se,0} < 3$ '. The analysis of Scenario 3 leads to the set of constrains of ' $V_{se,0} \geq 3$ ', ' $V_{se,0} - 2 < 3$ ' and ' $V_{se,0} + 6 - 2 \geq 3$ ', resulting in the constraint is ' $3 \leq V_{se,0} < 5$ '.

On the other hand, the *Scenario 4* leads to a set of constrains with no possible solution and is eliminated from the set of possible scenarios. The same happens with all the other scenarios of the infinite set defined during the analysis of *Centrifugal*.

4.3 Overview of the analysis procedure

The UML Collaboration Diagram illustrates the interactions among objects for a specific scenario and gives an overview of the proof obligations for this example. Basically, each proof obligation is represented by an arrow of discrete interaction (\longrightarrow) or by an arrow of continuous variable sharing (\dashrightarrow). The Collaboration Diagram of *Scenario 1* is presented in Figure 8 as an example.

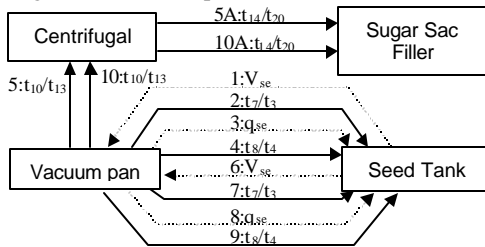


Figure 8. Collaboration Diagram for *Scenario 1*.

The most important advantage of this approach is that, by using the linear logic and the object oriented concepts, the analysis is performed based on the partial order of the events, i.e. the causality among them. The internal evolution of each object is analysed independently of the internal evolution of other objects. Comparing with the approach of building a global reachability tree (i.e., analysing a large number of sequences), in the proposed approach the analysis is based on the parallel

composition of a small number of causally short sequences. For example, to build a global reachability tree it would be necessary, after the event 5 (firing of t_{10}/t_{13}), to consider four different sequences, according to the fact that the transition t_{14} (internal event of object *Centrifugal*) is fired before, after or between the transitions t_5, t_6 and t_7 (internal events of object *Vacuum_Pan*). Using the proposed approach, combinatorial explosion of the number of possible states is avoided in the modelling (by using Petri nets) and in the analysis (by using linear logic).

It is important to highlight that the proof of a sequent (local analysis of an object) is valid independently of the behaviour of the other objects, excepted for time intervals where there are interactions among other objects. If the sequent of an Object 1 ' $p_A, p_B, t_A, t_B + - p_C$ ' is true and the sequent of an Object 2 ' $p_D, p_E, t_E + - p_F$ ' is true, then the global behaviour of the system is also true (the sequent ' $p_A, p_B, p_D, p_E, t_A, t_B, t_E + - p_C, p_F$ ' is obtained by the application of the linear logic rule $\otimes R$ (Girault et al., 1997)). The restrictions imposed by the object interactions are considered by the proof obligation. If all the proof obligations are true, the property is guaranteed during the time intervals of interaction.

5 Conclusion

This paper presents an approach for hybrid system analysis based on the use of Petri nets and object-oriented concepts. By exploiting the object independence a global analysis problem is decomposed into a set of local object proofs. For the analysis of reachability problems, the linear logic proofs show the causality constraints between the transitions, avoiding the enumeration of all sequences of transition firings in a global time. As a result, the number of analysed scenarios between two states are considerably reduced.

Acknowledges

The authors would like to thank the partial financial support of the governmental agencies FAPESP, CNPq, CAPES and RECOPE/FINEP.

Referências Bibliográficas

- Alur, R. & Dill, D. (1994) «A Theory of Timed Automata », Theoretical Computer Science 126:183-235.
- Antsaklis, P., Koutsoukos, X. (1998) «On Hybrid Control of Complex System: a survey », Proc. of the 3rd International Conference on Automation of Mixed Processes (ADPM'98), Reims.
- Booch, G. (1994), Object-Oriented Analysis and Design with Applications, 2nd ed. Addison-

- Wesley Longman, Inc. Harlow.
- Champagnat, R. et al. (1998) "Modelling and Simulation of a Hybrid System through Pr/Tr PN-DAE Model" 3rd International Conference on Automation of Mixed Processes, Reims.
- Girault, F. et al. (1997) «A logic for Petri nets », JESA Vol. 31, n. 3, Editions Hermes.
- Gueguen, H. & Zaytoon, J. (2001) «Principes de la vérification des systèmes hybrides », Colloque Francophone sur la Modélisation des Systèmes Réactifs (MSR 2001), Toulouse.
- Paludetto, M. (1991) Sur la commande des procédés industriels: une méthodologie basée objets et réseaux de Petri Thèse de Doctorat, Université Paul Sabatier, Toulouse.
- Villani, E. et al. (2002) « An Object-Oriented Approach for Hybrid System Modelling », Proceedings of 15TH IFAC World Congress on Automatic Control, Barcelona.