Formal Verification of Real-Time Systems Using a True Concurrency Semantics

Maarouk Toufik Messaoud and Saidouni Djamel Eddine

***Abstract*—In this paper we develop an algorithm of translation between two models for specifying real-time systems. The first model is the timed automata model and the second is the Durational Action Timed Automaton star (DATA*). Our approach is to interpret the behavior described by DATA *'s to Timed Automata. The main difference between the two models is the assumption on the actions, in the first model the actions are assumed atomic and have null durations, for the second the actions are assumed non-atomic and non-null durations, through the true concurrency semantics.**

***Index Terms*—Real-time systems, true concurrency semantics, actions duration, automata theory, LOTOS.**

I. Introduction

The design of real-time systems, is a very important problem today. This design requires methods and tools for formal specification to ensure the unambiguous specification and ensure smooth functioning of their properties. The behavior of such a system is described by the actions performed during its execution. Several formalisms and languages were defined to describe their behavior such as process algebras CCS[1], LOTOS[2] and their temporal extensions TCCS[3], RT-LOTOS[4], and D-LOTOS[5] etc.

The formal verification based on the model known as model checking[6] verification is a process used to demonstrate the functional correctness of a software or hardware design, generally described using a high-level language (or formal) with a well defined operational semantics.

We are interested in our approach to two model specifications: timed automata[7], [8], and DATA*[9], [10], this last was introduced with an aim of take into account the non-atomicity structural and temporal of actions. The idea is based on the principle of maximality semantics[11], [12] in which only the beginnings of actions are modeled, and where several actions can occur simultaneously. The purpose of execution actions being captured by the corresponding durations, this will allow other types of properties related to the quantitative aspects of behavior, such as maximal border of execution of a given trace, to be verified. In addition to the DATA* allow to take into account the main features of real-time systems, namely temporal constraints and the notion of urgency actions. On the other hand the model of DATA* represents the underlying model specification described in a D-LOTOS, which is represents a formal language which escapes the assumption of atomistic structural and temporal of actions, it incorporates both the temporal constraints and the durations of actions.

The timed automata(TA), is a formalism for specifing dense real-time systems. This formalism extends classical automata with a set of real-valued variables -called clocks- the increase synchronously with time and associates guard and update operations with every transition. Each location is constrained by an invariand, which restricts the possible values of the clocks for being in the state which can then enforce an transition to be taken.

The rest of the paper is divided as follows: in the next section we recall the definition and basic facts about TA and DATA*. Section 3 introduce a methodology of interpretation of DATA* in TA. The Section 4 contains an algorithm of transformation of the DATA* to TA, and the proof of termination of this algorithm and that complexity. Finally in section 4 we conclude.

II. Preliminaries

A. Timed Automata

Timed automata, are widely used model for the study and analysis of timed systems[13]. A TA is a tuple $\mathsf{A} = (X, Q, q_{init}, \Sigma, \rightarrow_A, Inv)$ where:

- X is a finite set of clocks,
- Q is a finite set of locations,
- $q_{init} \in$ Q is the initial location,
- Σ is an alphabet of actions,
- $\rightarrow_A \subseteq Q \times C(x) \times \Sigma \times 2^X \times Q$, is a finite set of transitions, a tuple $(q,g,a,r,q') \in \rightarrow_A$ (we write:$q \xrightarrow{g,a,r} q'$) represents an transition from q to q' with action a, g is the guard and r is the set of clocks to be reset to 0. A guard is used to specify when a transition can be performed,
- $Inv : Q \rightarrow C(x)$ is a function that assigns an invariant to each location, and contains usually only atomic formulas of the form $x \leqslant c$ or $x < c$.

The semantics of a TA is given by a timed transition system (TTS). A TTS is a tuple $(S, s_{init}, \Sigma, \rightarrow)$ where:

- $S = \{(q,v) \mid q \in Q \text{ and } v \in R_+^X \text{ with } v \models Inv(q)\}$
- $s_{init} = (q_{\text{init}}, v_0)$
- $\rightarrow \subseteq S \times (\Sigma \times R_+) \times S$ is a set of transition (we write also : $q \xrightarrow{e} q'$), contain two form of transition :
  -Delay transitions : $(q,v) \rightarrow (q', v+d)$, if $v+d \models Inv(q)$ with $d \leqslant d$.

Manuscript received September 30, 2012; revised November 20, 2012.

Maarouk Toufik Messaoud is with the Department of Computer Science and Mathematics, University Abbes Laghrour, Khenchela, Algeria. Member in the laboratory "Modeling and implementation of complex systems", University Mentouri (e-mail: toumaarouk@ yahoo.fr).

Saidouni Djamel Eddine is with the Department of Computer Science, University Mentouri, Constantine, Algeria. Team Leader "formal conception of complex systems" in the laboratory "Modeling and implementation of complex systems", University Mentouri (e-mail: saidounid@ yahoo.fr).

-Action transitions:$(q,v) \rightarrow (q',v')$, if $\exists\ q \xrightarrow{g,a,r} q'$ such the $v \vDash g$, $v'=v[r \leftarrow 0]$ and $v' \vDash Inv(q')$

A configuration of the system is a pair (q,v), where q is a location of the automaton and v is a valuation of the variables. The initial configuration is (q_0,v_0) where all clocks values are equal to 0 in v_0.

B. DATA*s Model

DATA* model is defined in ordre to take into accumt temporal contraints and urgency contraints present in real-time systems.

Let H, ranged over $x,y\ldots$ be a set of clocks with nonnegative values (in a time domain T). The set $\Phi_t(H)$ of temporal constraints over H is defined by the syntax $\gamma ::= x \sim t$, where x is a clock in H, $\sim \in \{=, <, >, \leq, \geq\}$ and $t \in$ T.

A DATA* A is a tuple (S, L_S, s_0, H, T) where:

1) S is a finite set of states,
2) $L_S : S \rightarrow 2^{\Phi_t(H)}_{\text{fn}}$ is a function which corresponds to each state s the set F of ending conditions of actions possibly in execution in s,
3) $s_0 \in S$ is the initial state,
4) H is a finite set of clocks, and
5) T $\subseteq S \times 2^{\Phi_t(H)}_{\text{fn}} \times 2^{\Phi_t(H)}_{\text{fn}} \times Act \times H \times S$ is the set of transitions. A transition(s,G,D,a,x,s') represents switch from state s to state s_0, by starting execution of action a and resetting clock x. G is the corresponding guard which must be satisfied to fire this transition. D is the corresponding deadline which requires, at the moment of its satisfaction, that action a must occur.

The semantics of a DATA* A $=(S, L_S, s_0, H, T)$ is defined by associating to it an infinite transitions system S_A over Act$\cup$T. A state of S_A(or configuration) is a pair $<s,v>$ such as s is a state of A and v is a valuation for H. A configuration $<s_0,v_0>$ is initial if s_0 is the initial state of A and $\forall x \in H$, $v_0(x) = 0$.

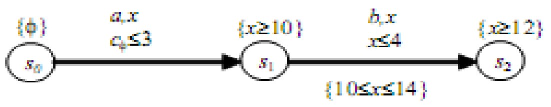

Fig. 1. Example of DATA*.

III. Timed Automata with Duration

In [14], it was introduced a methodology of interpretation of DATA* in TA. This interpretation allows to verify DATA* by using UPPAAL [15].

Let us consider the behavior expression $E=a;b;stop$, assuming that the actions a and b were not null durations and have the respective durations 10,12. Let us suppose that the action a can start only in the first three units of time and b in four units of time, the bebavior of E is given in Fig. 2.

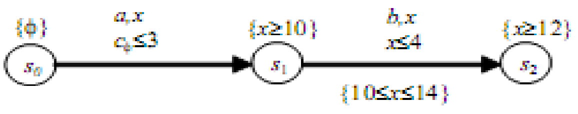

Fig. 2. DATA* represent execution of two successive actions.

From the initial state s_0, the action a can begin its execution while respecting the condition($c_\phi \leq 3$), which means that a can always begin its execution if a particular clock did not reach value 3 units yet since the sensitization of the system. The system passes in the state s_1 and cannot leave before the ending of the action a, in this state the action a is potentially in execution. The same reasoning applies for the action b.

The transition from s_0 to s_1 can be represented in TA by the transition $s_0 \xrightarrow{begin(a), c_\phi \leq 3, x} s_1$ (this transition expresses the beginning of the action a, the guard over the deadline of the offer, and x is the clock to be reset in zero by this transition to count the execution time of a). Once the system is in the state s_1 he cannot leave it neither before the action a finished nor after the expiration of the deadline of affer of the action b. The action b can obviously conply only if the action a finished it execution. The approach consists in modelling an action which has two optional parameters: a deadline of offer and a duration of execution with an timed automaton containing :

- A transition to express the beginning of the action
- A state, where the system lives during the execution of the action.

The guard of the next transition is used to capture, at the same time, the end of the action and the beginning of the next action (the clock x will be reset in 0 once again we say that it will be reused for action b). Finally we obtain the timed automaton representing the behavior E in the Fig. 3.

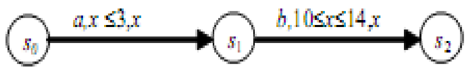

Fig. 3. Timed automaton corresponding DATA*.

By comparing both models illustrated by both Fig. 2 and Fig. 3, we can notice that both express the same behavior without for the timed automaton we lost the information at the end of the action b. It is evident that it is impossible to capture the end of the last action without adding a new transition. For this, it was consider that all the processes use necessarily the particular action δ to mark their ends. And so we obtain the behavior illustrated by the following Fig. 4.

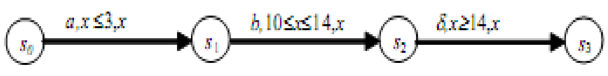

Fig. 4. Adding the action δ.

Now we consider the parallel case, the system S is composed of two subsystems S1 and S2 running in parallel and synchronizing on an action d. The subsystem S1 performs the action a followed by d, while S2 performs the action b followed by d.

Suppose that the actions a, b and d have the respective durations 10, 12 and 4. The temporal restriction of the domain of sensitization of action d is 5 and 4 time units according the source of d from the S1 or S2. The DATA * corresponding to the system S is given by the Fig. 5.

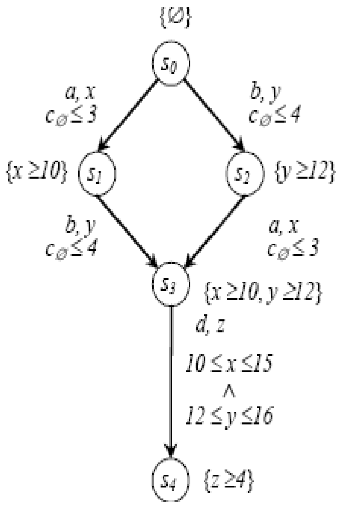

Fig. 5. DATA* of system S.

Applying the above reasoning, we obtain the automaton shown in Fig. 6

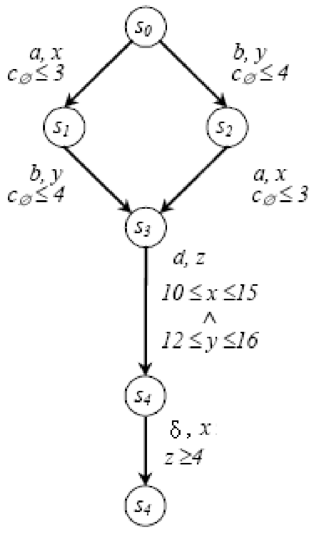

Fig. 6. Timed automaton generated.

The same reasoning applies for the interpretation of the urgency by timed automaton. The expression of the urgency in timed automata is done using the invariants.

IV. Algorithm of Transformation of the DATA* in TA

This section we propose an algorithm for transformation DATA* to TA. The objective of this transformation is the exploitation of the TA for possible formal verification using the model checker UPPAAL.

A. Algorithm DATA*2TA

```
Algorithm DATA*2TA Translate DATA* to TA
Input: DAT A* ($S,L_s, s_0,H, T_D$)
Output: T A($Q,Act, q_0,X,E, Inv$)
$q_0$:= $s_0$;         // intial location
$Q$:={$q_0$};       // set of location
$X$:= $\phi$        // set of clocks
$Act$:=$\phi$      //alphabet of actions
For all transition $t \in T_D$
        // t: transition of form $s_i \xrightarrow{G,D,\alpha,x} s_{i+1}$
  //initialization parameters for each transion and sets of
  output TA
        $q_s$:=$t.s_i$; $q_t$:=$t.s_{i+1}$;
        $a$:=$t.\alpha$;   $x$:=$t.x$;
        $Q$:=$Q \cup \{q_s,q_t\}$;
        $Act$:=$Act \cup \{a\}$
        $X$:=$X \cup \{x\}$;
        $G$:=$t.G$; $D$:=$t.D$;
        if ($D=\emptyset$) then
                //the current action is not urgent
          $I$:=$\emptyset$;
          renameClocks(G,x);
                // Rename the clocks in G with x
          Condition:=G;
        else
          $I$:=Invariant(G,D);
          condition:=$G/D \wedge P_i(D)$
        endif
          $e$:=($q_s,q_t,a,x$,condition);
          $Inv$:=$Inv \cup \{(I,q_t)\}$;
          $E$:=$E \cup e$;
      endfor
  //Get the duration of execution of the last action in the
  model
      $C$:=ExtractDuration($q_t,L_s$);
      $Act$:=$Act \cup \{\delta\}$      // Last transition : action $\delta$
      $x$:=get($H \backslash X$) //clock  for $\delta$
      $X$:=$X \cup \{x\}$;
      $E$:=$E \cup (q_t,q_\delta,\delta,x,C)$;

      $TA$:=($Act,Q,q_o,X,E,Inv$)  //In output the TA
      return $TA$;
  end.
```

B. Functioning of the Algorithm

The algorithm of transformation of the DATA* in TA gets in input a DATA* represented by: a set of state *S*, a function L_s. An initial state s_0, a finite set of clocks, and T_D set of transitions of the form $s \xrightarrow{G,D,\alpha,x} s'$.The temporal constraint *G*(respectively *D*) is described as follows : $\wedge d_i \sim x \sim d_s$ such as: $d_i,d_s \in \mathbb{R}$ and $\sim \in \{<,\leq\}$ and $d_i \sim x$, $x \sim d_s \in G$ (respectively *D*). For a urgency constraint D ($D \neq \phi$): $G/D=\wedge d_i \sim x \sim d_s$ such as $=\wedge d_i \sim x \sim d_s \vDash G \wedge D$

$P_i(D)= \wedge d_i \sim x$; $P_s(D)= \wedge x \sim d_s$

In out the algorithm produces a TA constructed as the following way :

- At first the initial state of the TA it is the same initial state of the DATA*
- For every transition of the form $s \xrightarrow{G,D,\alpha,x} s'$ in DATA*: The algorithm begins by initializing the starting state *s*, the state arrived *s'*, the current action α and the clock *x*.
- Get the two temporal constraints: *G* (guard), *D* (deadline)
- Updating the sets (*Q*, *Act*, *H*), and the set *E* of transitions of the TA$<s,s',a,x,G/D \wedge P_i(D)> \in E$, the invariant $I(s)=P_s(D)$

If the current action is not urgent thus the urgency constraint *D* is empty then the construction of the transition of the TA is as follows:

- Both states of starts and to arrive already are to initialize
- Absence of the invariant on the state to arrive
- The guard *G* of the DATA* becomes a condition on transition

The function renameClocks(G,x), allows to rename the clock of type cϕ in DATA* by the clock x in the constraint *G*. because the time for a clock in DATA* cannot elapse before the assigning of this clock to an action, it is for this reason there that the system is increased with the clock c_ϕ(initialized dices the sensitization of system) to count the time before the onset of any action. On the other hand, the clocks of a TA begin to count the time dices the sensitization of the system. So, we can omit the clock c_ϕ and replace it with *x* without affecting semantics of the model. If the action is urgent: The invariant is calculated from the urgency constraint *D* and the guard *G*. In the last step it is always necessary added the action δ to the TA to be able to capture the end of the last action of DATA*. The function $ExtractDurtion(q_t,L_S)$ it used to calculate the duration of the last action. Finally the algorithm returns the timed automaton TA.

C. Termination

The following theorem establishes the terminate of the algorithm DATA*2TA, in order for Algorithm to be correct the termination condition must ensure that when the algorithm terminates, it givens timed automaton.

*1) Theorem 1(termination): The algorithm DATA*2TA terminate*

Proof **:** To demonstrate this point, let us remind that the DATA* in input represents a finished semantic model,that is the number of state in the model is determined, by construction thus the ensemle of the transition TD is finished, what assures and guarantees the terminate of our algorithm.

D. Complexity

We show that the time complexity of our algorithm for transformation of DATA*2TA.

1) Theorem 2(complexity): Given an DATA, to transform the DATA* into timed automaton is a linear problem in number of transitions.*

*Proof***:** We consider a DATA* (which does not contain either diagonal constraints or constraints of update). The size of TA built is the same that the size of the DATA* more the transition of the action δ and the last state. As the size of the DATA* is linear, the total size of construction is thus linear.

V. Conclusion

In this paper, we have developed an algorithm which produces an TA from DATA*. We also prove the guarantees termination of the algorithm and its complexity. The main interest of the model of the DATA* it is the use of semantics of maximality which allows the explicit expression of durations, and it supports temporal constraints including urgency of actions. Thus, correctness properties relative to system specified can be checked on maximality-based.

References

[1] R. Milner, *Communication and concurrency*, Prentice Hall, 1989.

[2] ISO8807, "LOTOS, a formal description technique based on the ordering of observational behaviour," *ISO*, November 1988.

[3] F. Moller, C. Tofts, "A temporal calculus of communicating systems," in *J. C. Baeten and J. W. Klop, editors, Concur, LNCS*, Springer-Verlag, vol. 458, pp. 401-415, 1990.

[4] J. P. Courtiat, M. S. D. Camargo, and D. E. Saidouni, "RT-LOTOS: LOTOS temporisé pour la spécification de systèmes temps réel," in *Proc. Ingénierie des Protocoles (CFIP'93),* Hermes, pp. 427-441, 1993.

[5] D. E. Saidouni and J. P. Courtiat, "Prise en compte des durées d'action dans les algèbres de processus par l'utilisation de la sémantique de maximalité," in *Ingénierie Des Protocoles (CFIP'2003).* Hermes, France, 2003.

[6] E. M. Clarke, E. A. Emerson, and A. P. Sistla, "Automatic verification of finite state concurrent systems using temporal logic specifications," *ACM Transactions on Programming Languages and Systems*, vol. 8, no. 2, pp. 244-263, April 1986.

[7] R. Alur, "Techniques for automatic verification of real time systems," Ph. D. disseertation, Stanford University, Stanford, CA, USA, 1992.

[8] R. Alur and D. Dill, "A theory of timed automata," *TCS*, vol. 126, pp.183-235, 1994.

[9] D. E. Saidouni and N. Belala, "Actions duration in timed models," in *International Arab Conference on Information Technology (ACIT'2006)*, Yarmouk University, Irbid, Jordan, December 2006.

[10] N. Belala and D. E. Saidouni, "Non-atomicity in timed models," in *International Arab Conference on Information Technology (ACIT'2005)*, Al-Isra Private University, Jordan, 2005.

[11] D. E. Saidouni, "Sémantique de maximalité: Application au raffinement d'actions en LOTOS," Ph. D. dissertation, LAAS-CNRS, Toulouse, France, 1996.

[12] D. E. Saidouni and J. P. Courtiat, "Maximal trees," *Research Report 95192*, LAAS-CNRS, 7 av. du Colonel Roche, 31077 Toulouse Cedex France, June 1995.

[13] R. Alur and D. Dill, "Automata for modeling real-time systems," in *Proc. 17th International Colloquium on Automata, Languages and Programming (ICALP'90), Lecture Notes in Computer Science*, Springer-Verlag, vol. 443, pp.322-335, 1990.

[14] D. E. Saidouni, A. Boumaza, and S. Guellati, "Prise en compte des durées d'actions dans la vérification des automates temporisés," in *Proc. 1st International Conference on Information Systems and Technologies ICIST'2011*, Algeria, Tebessa University, April 2011.

[15] K. Larsen, A. David, G. Behrmann, and W. Yi, "A tool architecture for the next generation of uppaal," Technical report, Uppsala University, 2003.