

# Relative Timed Model for Coordinated Multi Agent Systems

Said Layadi<sup>1</sup>(✉), Jean-Michel Ilić<sup>2</sup>, Ilham Kitouni<sup>1</sup>, and Djamel-Eddine Saidouni<sup>1</sup>

<sup>1</sup> MISC Laboratory, University of Abdelhamid Mehri – Constantine 2, 25000,  
Constantine, Algeria

{layadi, kitouni, saidouni}@misc-umc.org

<sup>2</sup> Universities of Paris UPMC and Paris Descartes 4 place Jussieu, 75005, Paris, France  
jean-michel.ilie@upmc.fr

**Abstract.** The MAS engineering is becoming very important, it is concerned with models, methods and tools. Therefore, verifying the correctness of MAS is the next challenge. We are interested by MAS where each participating agent has its own physical clock of varying frequency, while no global clock is available or desirable. Under such circumstances models must be adapted. In this paper we attempt a novel approach to model the MAS, with a respect of two characteristics, the concurrent aspect and heterogeneity of agents (perceived as a different time rates of agents plan execution). Timed automata with action durations are used; for the circumstance it's extended to deal with relative time rates. Its semantic is abstracted by a novel equivalence relation leading to a region automaton for decidability assessment and proof.

**Keywords:** Relative time rates · Timed automata with action durations · Region graph · Multi agent systems · Timed transport fleets

## 1 Introduction

Multi Agent Systems (MAS) are ever-present in computer science applications. This paradigm is used in different domains where reactivity, mobility, dynamicity and adaptation of the system to uncertain or unpredictable factors should be considered. The MAS engineering (i.e. specification, development, management, deployment...) is becoming very important. It is concerned with models, methods and tools. Therefore, verifying the correctness of MAS becomes a fantastic challenge.

We are interested by MAS where each participating agent has its own physical clock of varying frequency, while no global clock is available or desirable. Under such circumstances it's impossible to model system using discrete semantics of time without considering the clock frequencies of participating components. Hence it's natural to study these systems in terms of different time evolutions.

In this paper we attempt a novel approach to model the MAS, with a respect of two characteristics, the concurrent aspect of the MAS and the heterogeneous of components (agents) perceived as a different time rates of agents plan execution.

*Models.* For this aim, timed models are suitable. The durational actions timed automata (daTA) [2] are a form of timed automata [1], that admits a more natural representation of action durations and advocates carrying true concurrency (which are realistic assumptions for specifying in natural way MAS). It's based on the Maximality semantics [4]. Maximality semantics has been proved necessary and sufficient for carrying both the refinement process and action durations. The daTA model has been defined and a nice characterization of the model was presented in [2] and [7]. So the concurrency aspect of MAS is modeled by the timed automata with action durations.

The daTA model assumes a “global clock” semantics, i.e., all clocks advance simultaneously and at the same rate (and there is a common initial instant). All possible executions of daTA are then represented by an infinite transition system where, for any given state, the system may evolve in two possible ways: either it executes an action or it delays with a given amount of time the potential execution. The decidability of the daTA has been proved using the so-called region graph construction [1]. In this paper we are concerned by the coordination problem in MAS. Mainly, this consists in maintaining the synchronization of the agent plans with respect to some objective in a consistent timed context. We consider real time application wherein the agent plans refers to timed actions whose durations are known. Agents are assumed to be able to communicate via reliable materials, in order to achieve some plan called coordinated plan.

*The paper contribution.* We propose to model plan in a more attractive way using the standard algebraic language based on LOTOS, seeing plans as the execution of concurrent processes [3]. In fact, LOTOS specifications supplies modular concepts useful to describe some plan over several agents. Moreover, LOTOS-basic specifications are translated in daTA relative time rates (daTA-RT). daTA-RT compactly represents the possibly infinite behaviors of the coordinated plans, this model is concerned by the timing constraints defined over the MAS plans, taking into account relative time rates that distinguish the speeds of the agents in coordination. In this paper, we show how to build a (finite) region graph structure from a daTA-RT specification, for decidability assessment and proof. We extend by this proposal the result of a precedent work [12] over timed automata with relative time rates to model heterogeneity property.

*Related work.* In the literature, the two aspects, verification and time management works are generally focused on how to consider clocks and time for example see [13]. In the main cases, clocks are synchronized and used by all processes (read and reset). Or clocks can drift by a certain amount of time  $\Delta$ , particularly as long as the processes do not communicate (via synchronizing actions). In [14], distributed systems are modeled by means of network of timed automata evolving in different rates. However, checking emptiness or universality turns out to be undecidable in the majority of cases. In [12], we investigate the decidability for verification assessments of real time systems with relative speed of clocks (what we call heterogeneity property). More precisely, under same hypothesis we answer this question positively over the timed automata with relative time rates.

Regarding designing MAS, several recent papers focus on the framework based on refinement paradigm. In [6], the authors propose a formal modeling of critical MAS that aims to derive a secure system implementation. The approach is based on Event B language. They are interesting by modeling fault tolerant MAS. In the same context of Event B specification language, authors in [8] propose a formal approach for self-organizing MAS. [9] Addresses a top-down approach for MAS protocol description using Finite State Automata (FSA) and multi-Role Interaction (mRI) abstraction. We don't forget the work [11] in which MAS are specified by AgLOTOS. This latter language captures communication of processes (i.e. agents) by message passing, in addition to classical features of concurrent processes. In our case the basic LOTOS is sufficient for what we intend to present.

*Paper outlines.* In Section 2, the specification of coordinated MAS plans is presented, based on LOTOS concepts, the relative time rates are explained and our running example is built. In Section 3, the daTA model is recalled, its extension to relative time is defined and an informal manner of generating daTA structure from LOTOS expression is shown. The main contribution of this paper is proposed in Section 4, where the semantic of daTA is presented and formally defined. In all the paper the same example is used to clarify concepts and applications. The end section concerns conclusion and some immediate perspectives.

## 2 Coordinated Mas Specification

In this paper, a multi agent system is a tuple  $MAS = (Ag, Plan, Act, \tau, \gamma)$  where  $Ag$  is set of agents;  $Plan$  is a set of agent plans, called coordinated plans since some of them are realized by several agents,  $Act$  is the set of actions mentioned in these plans.  $\tau = (\tau_p)_{p \in Ag}$  is a rate mapping associating a relative time rate with each agent characterizing the speed of the agent to execute their actions,  $\gamma$  is a duration mapping assigning a global duration value to each action of  $Act$ , evaluated in a number of execution cycles (called time units). In the following, we describe the specification of coordinated plans as an extension of the language Basic LOTOS, precising which parts of the plans are dedicated to each agent.

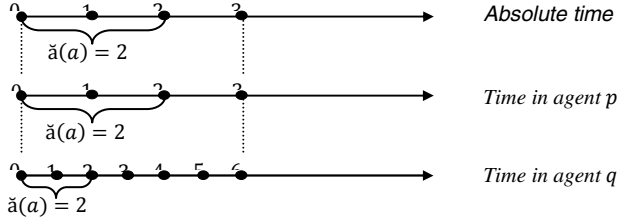
*The relative time model.* In the MAS systems, agents are assumed to have a notion of clocks to achieve their actions. Since agents can have different speeds, we assume that the speeds are relative according to a global time-scale (denoted absolute time), thus the duration performance of some action can be more or less important, depending on the agent considered to execute the action.

As example, Fig.1 shows an action  $a$  of duration 2, which requires 2 times more to be achieved by the agent  $p$ . According to any global time  $t$ , the time rate  $\tau_q = 2t$  and  $\tau_p = t$ , in such a way that at any time  $\frac{\tau_p}{\tau_q} \in \mathbb{Z}$ .

*Coordinated plan specification.* The plan of an agent  $p$  is specified by an agent expression  $E_p$  describing the actions to be executed for achieving the plan. The execution is assumed to be controlled by the agent  $p$ , however  $E_p$  can be composed of

sub-expressions whose execution can be performed by other agents. An agent expression inherits from the syntax of (Basic) LOTOS [3] as follows:  $P ::= E_p$  and  $E_p ::= \text{exit} \mid \text{stop} \mid a; E_p \ (a \in \text{Act}) \mid E_q \sim E_r$ .

Where  $q, r \in \text{Ag}$  and  $\sim \in \{||, ||, |[L]|, [], \gg, [ > \}$  is a LOTOS operator.



**Fig. 1.** The action  $a$  of global duration 2, is achieved two times faster by the agent  $q$  than by agent  $p$

The elementary expression `stop` specifies a plan behavior without possible evolution and `exit` represents the successful termination of some plan. In the syntax, any operator  $\sim$  as in standard LOTOS:  $E_q || E_r$  specifies a non-deterministic choice,  $E_q \gg E_r$  a sequential composition and  $E_q [ > E_r$  the interruption of the left hand side part by the right one. The LOTOS parallel composition, denoted by  $E_q |[L]| E_r$ , can model both synchronous composition when ( $L = \text{Act}$ ), denoted by  $E_q || E_r$ , and asynchronous composition,  $E_q ||| E_r$  when ( $L = \emptyset$ ). In fact, the LOTOS language exhibits a rich expressivity such that the sequential executions of plans appear to be only a particular case.

*Our running example.* The example concerns two trucks A and B such that A initially placed in the location  $l_1$  must pick up the load in the location  $l_2$  and delivers it to the location  $l_4$ . As the load is initially placed in  $l_3$ , B initially placed in the location  $l_2$ , proposes to get the load from  $l_3$  in such a way that A can meet it in  $l_2$  and take the load. The problem for A and B is to meet them in minimum time, in case they start at the same time. In order to coordinate, each truck is equipped with a software agent able to discuss and synchronize with the other agents in the system. Both agents A and B refer to the following coordinated plan:

$P ::= E_A |[meet]| E_B$ ; with  $E_A ::= \text{move}_A(l_2); \text{meet}; \text{move}_A(l_4); \text{exit}$  and  $E_B ::= \text{move}_B(l_3); \text{get\_load}_B; \text{move}_B(l_2); \text{meet}; \text{exit}$ .

Moreover, duration of actions are given by the respective learnt experiences of A and B about transportations. For the simplicity of the example, all durations of actions are assumed equal to 1 time unit.

### 3 Timed Automata with Action Durations and Relative Time Rates for Coordination Plans

To model duration of actions, every edge of the automaton is annotated by constraints on clocks which implicitly enclose them, of course those that are already started. A single clock is reset on every edge. When clock is reset it corresponds to the beginning of event. The termination of action will be captured by information (temporal formulas) on locations of the automaton, precisely on the destination location. In fact, the duration of an action is either in the constraint of the following edge, if there is dependence between the successive actions, otherwise it is in the next locations and that means: action is not over yet. This elegant way to capture the durations is the effect of the maximality semantics. An example of a daTA  $A$  is shown in Fig. 2. The automaton consists of three localities  $l_0, l_1, l_2$  and two clocks  $x, y$ . A transition from  $l_0$  to  $l_1$  represents the start of action  $a$  (indicating the beginning of its execution), the transition from  $l_1$  to  $l_2$  is labeled by  $b$ .

Assuming a time granularity of seconds, the automaton  $A$  starts in locality  $l_0$ . As soon as the value of  $y$  is less than or equal to 4, the automaton can make an a transition to  $l_1$  and reset the clock  $x$  to 0. On the locality  $l_1$  the temporal formula  $\{x \geq 2\}$  represents information about the duration of the action  $a$  (it is important to differentiate it from invariant in timed automata). When  $x$  is at most 2 and is at least 5, transition to  $l_2$  can be started ( $b$  executed) and  $y$  is reset. In the same logic the temporal formula  $\{y \geq 7\}$  represents duration of the action  $b$ .

*Preliminary.* In the following we consider  $R_{\geq 0}$  a set of nonnegative real numbers. Clocks are real variables take values from  $R_{\geq 0}$ . Let  $H$  be a set of clocks, a clock valuation over  $H$  is a function that assigns a nonnegative real number to every clock.  $V_H$  is the set of total valuation functions from  $H$  to  $R_{\geq 0}$ . A valuation is noted  $v \in V_H$ , and for  $d \in R_{\geq 0}$ ,  $v + d$  maps every clock  $x$  to  $v(x) + d$ . For  $\lambda \subseteq H$ , the valuation  $v[\lambda := 0]$  is defined by:  $(v[\lambda := 0])(x) = 0$  if  $x \in \lambda$ ,  $v(x)$  otherwise.

The set  $C(X)$  of clock constraints  $C$  is defined by the grammar:

$C ::= \text{true} \mid \text{false} \mid x \sim c \mid C \wedge C$ , where  $x \in X$ ,  $c \in \mathbb{N}$  and  $\sim \in \{<, >, =, \leq, \geq\}$ . We write  $v \models C$  when the valuation  $v$  satisfies a clock constraint  $C$  over  $X$  iff  $C$  evaluates to true according to the values given by  $v$ .

We also use a subset of constraints where only the atomic form of clocks comparison is allowed. This set is defined by  $C_d(H)$  by the grammar:  $C ::= x \geq c$ , where  $x \in X$  and  $c \in \mathbb{N}$ . This subset represents condition duration over the set of actions noted by  $\text{Act}$ .

*Definition 1 (daTA).* A Durational Actions Timed Automaton daTA  $\mathcal{A}$  is a tuple  $(S, s_0, H, T, L_S)$  over  $\text{Act}$ , where:  $S$  is a finite set of locations.  $s_0 \in S$  is an initial location.  $H$  is a finite set of clocks.  $T \subseteq S \times C(H) \times \text{Act} \times H \times S$  is a finite set of edges. An edge  $e = (s, g, a, x, s') \in T$  ( $s \xrightarrow{G, a, x} s'$ ) represents an edge from location  $s$  to  $s'$  that launches the execution of action  $a$  whenever guard  $g$  becomes true.

$L_S: S \rightarrow 2^{C_d(H)}$  is a maximality function which decorates each location by a set of timed formulas named action durations. These formulas indicate the status of action execution at the corresponding state.  $L_S(s_0) = \emptyset$  means that no action is yet started.

### 3.1 Timed Automata with Action Durations and Relative Speed Clocks Model

In this section we define a daTA with relative speed of clocks for modeling MAS; this is what we designate as the relative time rates in the global system.

*Definition 2 (daTA with relative time rates).* A daTA with relative time rates (daTA-RT) over the set of agents  $Ag$  is a structure  $A = (\mathcal{A}, \pi)$  where  $\mathcal{A} = (S, s_0, H, T, L_S)$  is a daTA and  $\pi$  is a mapping from  $H$  to  $Ag$  such that, for each  $p \in Ag$ , we have  $\pi^{-1}(p) \subseteq H$ .

Note that each clock evolves at the same rate in a particular agent (as the time evolves). This clock is then said to belong to that agent, and the mapping  $\pi$  (owner map) describes this in the above definition. We suppose that, in daTA with relative time rates, all clocks in  $H$  evolve at relative rates. Each rate characterizes the speed of an agent  $p$ . It depends on some absolute time given by the function  $\tau_p: \mathbb{R}_{\geq 0} \rightarrow \mathbb{R}_{\geq 0}$  with  $\tau_p(0) = 0$  and  $\tau_p(t)$  returns the local time in each agent  $p \in Ag$  for the instant  $t$  of absolute time. Moreover,  $\tau$  is a tuple of local time rate functions such that  $\tau = (\tau_p)_{p \in Ag}$ . The function  $\tau_p$  is the  $p$  local time rate. For a time value  $t$ , the mapping function  $\tau: \mathbb{R}_{\geq 0} \rightarrow \mathbb{R}_{\geq 0}^{Ag}$  assigns the tuple  $(\tau_p(t))_{p \in Ag}$  to  $\tau(t)$ .

### 3.2 From Basic LOTOS with Action Durations to daTA with Relative Time Rates

In this section, we informally present the manner of generating in an operational way a daTA-RT starting from a Basic LOTOS specification with action durations. The approach is very close with the one of [4] for the generation of MLTS (Maximality based labeled transition system). In our context it's viewed as an unfolding operation of the behavior expression to state transition structure. Take again the behavior of the

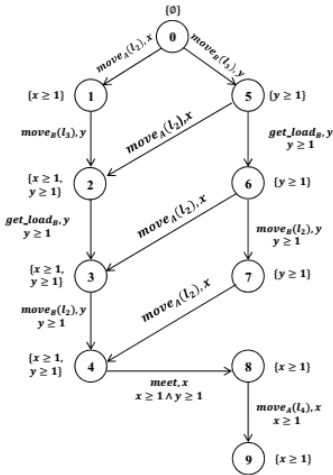


Fig. 3. A daTA-RT of the transport example (Our running example)

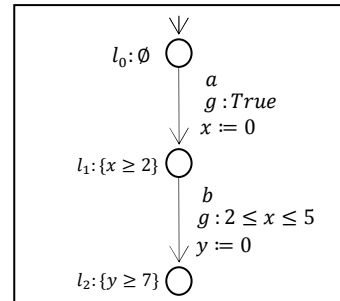


Fig. 2. A daTA

system described in the example of transport which is expressed as the parallel composition of two subsystems with synchronization on the action *meet*.

Recall that the behavior of system is specified by the following Basic LOTOS with action durations expression:  $E_A[[meet]]E_B$ . As already presented (in section 3.2), all states of the daTA-RT encapsulate a timed formula representing duration conditions of actions.

The generation of the daTA-RT's structure starts from the initial expressions, as a form of unfolding structure. To each configuration is associated a daTA-RT state as well as a behavior of sub-expression; (i.e. configurations corresponding daTA-RT's states) combines the expression to developed and duration conditions inherited from the previous step of unfolding, in the form  $F[E]$  where  $F \in 2^{C_d(H)}$ . In the initial state of the coordinated plan  $P ::= E_A[[meet]]E_B$ , no action is running, which explains why the duration conditions set is empty in the initial configuration. The initial configuration (State) of daTA-RT is  $s_0 = \emptyset[E_A[[meet]]E_B]$  where no action was launched yet. From this configuration, action  $move_A(l_2)$  can be allowed. A clock  $x$ , is assigned to this occurrence of  $move_A(l_2)$ , it will be initialized to zero. Action  $move_A(l_2)$  does not await the end of any other action to be able to comply, from where the guard of the transition is true.

$$\underbrace{\emptyset[E_A][[meet]] \emptyset[E_B]}_{\text{config}_0} \xrightarrow{\text{true, } move_A(l_2), x} \underbrace{x \geq 1[meet, move_A(l_4), exit][[meet]] \emptyset[E_B]}_{\text{config}_1}$$

The duration condition of the configuration  $\text{config}_1$  corresponding to state  $s_1$  is  $\{x \geq 1\}$ , which means that action  $move_A(l_2)$  is running at this step of execution. From the configuration  $\text{config}_1$  corresponding to state  $s_1$ , the only possible transition to construct is that corresponding to the launching of the action  $move_B(l_3)$ , from where the derivation

$$\underbrace{\underbrace{\emptyset, move_B(l_3), y}_{\text{config}_1} \xrightarrow{x \geq 1} x \geq 1[meet, move_A(l_4), exit][[meet]]|_{y \geq 1}[get\_load_B; move_B(l_2); meet; exit]}_{\text{config}_2}$$

With the same reasoning, we obtain  $\text{config}_3$  and  $\text{config}_4$ , note the guard in this step is  $\{y \geq 1\}$ , this expresses dependence of loading B and the termination of moving B.

$$\underbrace{\text{config}_2 \xrightarrow{\{y \geq 1\}, get\_load_B, y} x \geq 1[meet, move_A(l_4), exit][[meet]]|_{y \geq 1}[move_B(l_2); meet; exit]}_{\text{config}_3}$$

The clock  $y$  is assigned to the action  $get\_load_B$  because of its discharge at the time of  $get\_load_B$ .

$$\underbrace{\text{config}_3 \xrightarrow{\{y \geq 1\}, move_B(l_2), y} x \geq 1[meet, move_A(l_4), exit][[meet]]|_{y \geq 1}[meet; exit]}_{\text{config}_4}$$

From the configuration  $\text{config}_4$ , action *meet* can comply only if the two actions  $move_A(l_2)$  and  $move_B(l_2)$  finished their execution, in other words, only if duration conditions belonging to the set  $\{x \geq 1, y \geq 1\}$  are all satisfied, from where the set

$\{x \geq 1, y \geq 1\}$  corresponding to the condition  $x \geq 1 \wedge y \geq 1$ . The following transition becomes possible:

$$\text{config}_4 \xrightarrow{\{x \geq 1 \wedge y \geq 1\}, \text{meet}, x} \underbrace{x \geq 1 [\text{move}_A(l_4), \text{exit}] | [\text{meet}] | x \geq 1 [\text{exit}]}_{\text{config}_8}$$

For any action of synchronization, the assigned clock must evolve according to the slowest agent speed. In this example a clock  $x$ , which evolves according to the rate of truck A ( $\tau_A$ ), is assigned to the action *meet*. Thus, the truck B, which is the faster, must wait that truck A finished the synchronization action before going on. This mechanism ensures the synchronization on the end of each synchronized action. The configurations  $\text{config}_9$  and  $\text{config}_{10}$  are obtained as follows:

$$\begin{aligned} \text{config}_8 &\xrightarrow{\{x \geq 1\}, \text{move}_A(l_4), x} \underbrace{x \geq 1 [\text{exit}] | [\text{meet}] | \emptyset [\text{exit}]}_{\text{config}_9} \quad \text{and} \\ \text{config}_9 &\xrightarrow{\{x \geq 1\}, \delta, x} \underbrace{x \geq 0 [\text{stop}] | [\text{meet}] | x \geq 0 [\text{stop}]}_{\text{config}_{10}} \end{aligned}$$

The same reasoning is applied in the following way to the other branch of the transition system where the action  $\text{move}_B(l_3)$  begins its execution before the action  $\text{move}_A(l_2)$ :

$$\text{config}_0 \xrightarrow{\emptyset, \text{move}_B(l_3), y} \underbrace{\emptyset [E_A] | [\text{meet}] | y \geq 1 [\text{get\_load}_B; \text{move}_B(l_2); \text{meet}; \text{exit}]}_{\text{config}_5}$$

The system cannot execute any action starting from configuration  $\text{config}_{10}$ , end of the unfolding operation of  $E_A | [\text{meet}] | E_B$ . The abstraction of this unfolding (transformation) is depicted by the daTA-RT structure in Fig. 3.

## 4 The Semantics of daTA-RT model

In the literature [1], an equivalence relation is proposed to aggregate states of the timed transition system (configurations) such that an equivalence class represents a configurations set. The equivalence classes of clock valuations are named clock regions. The design of Multi agents system becomes coherent if the components share a conjoint perception of time [5]. Therefore, it is important that the general perception is consistent. This perception takes its full dimension when calculating the semantics graph of the model. This motivates the redefinition of concepts like clock regions and region automaton. We will hereafter focus on the effect of the relative clock speeds.



#### 4.1 Equivalence Classes of Clock Valuations

Let  $A = ((S, L_S, s_0, H, T), \pi)$  be a daTA-RT over Act and a set of agents Ag.

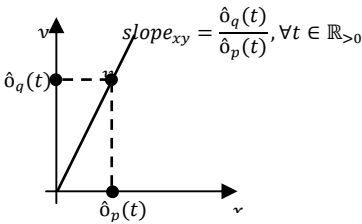
**Definition 3** ( $slope_{xy}$ ). Let  $x$  (resp.  $y$ ) be a clock that belongs to the agent  $p$  (resp.  $q$ ) and evolves according to the rate function  $\tau_p$  (resp.  $\tau_q$ ). We define  $slope_{xy}$  as the ratio of local-time rate functions  $\tau_q$  and  $\tau_p$ , noted  $slope_{xy} = \frac{\tau_q}{\tau_p}$ . (see Fig. 4).

Given a pair of clocks,  $x$  and  $y$  (within respectively agent  $p$  and  $q$ ), their owner speeds will make them diverging from time reference at a certain speed. That is equal to the ratio between their owner rates. It represents the slope of the straight line in Fig. 4. As there are only finitely many clock constraints on clock  $x$ , we can determine the largest integer  $c_x \in \mathbb{N}$  with which  $x$  is compared in some clock constraint (guard) of the daTA-RT  $A$ . In the remainder and for every pair of clocks  $x$  and  $y$ , the parameter  $slope_{xy}$  is assumed be an integer constant whatever the value of time  $t$ .

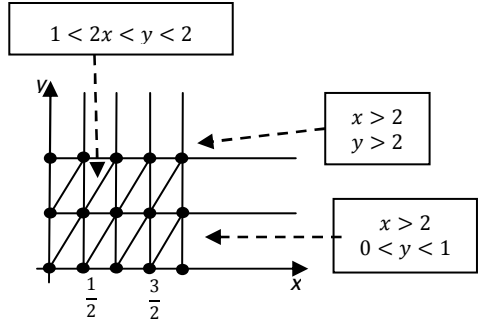
**Definition 4** (*equivalence relation  $\sim$* ). We define the equivalence relation  $\sim$  over the set of all clock valuations,  $v \sim v'$  iff all the following conditions hold:

1. For all  $x \in H$ , either  $\lfloor v(x) \rfloor$  and  $\lfloor v'(x) \rfloor$  are the same, or both  $v(x)$  and  $v'(x)$  are greater than  $c_x$ .
2. For all  $x, y \in H$  with  $v(x) \leq c_x$ ,  $v(y) \leq c_y$  and  $x$  (resp.  $y$ ) evolves according to  $\tau_p$  (resp.  $\tau_q$ ):  
c.  $\frac{1}{slope_{xy}} \leq v(x) \leq (c+1) \cdot \frac{1}{slope_{xy}}$  iff  $c \cdot \frac{1}{slope_{xy}} \leq v'(x) \leq (c+1) \cdot \frac{1}{slope_{xy}}$  for  $c \in \mathbb{N}$  and  
 $\text{fract}(slope_{xy}v(x)) \leq \text{fract}(v(y))$  iff  $\text{fract}(slope_{xy}v'(x)) \leq \text{fract}(v'(y))$

An equivalence class of clock valuations induced by  $\sim$  is a clock region of  $A$ .



**Fig. 4.** Two clocks evolution with different speeds



**Fig. 5.** Clock Regions Deduced by the Relation  $\sim$

## 4.2 The Representation of Clock Regions

Each equivalence class of clock valuations can be specified by a finite set of clock constraints it satisfies. The notation  $[v]$  represents the clock region to which  $v$  belongs.

*Example.* we consider two clocks  $x$  and  $y$  which evolve at different rates such that  $\text{slope}_{xy} = 2$ ,  $c_x = 2$  and  $c_y = 2$ . The clock regions obtained by the Definition 4 (equivalence relation) are depicted in Fig. 5. Thus, we have 15 corner points (e.g.  $[x = 0.5 \wedge y = 2]$ ), 38 open line segments (e.g.  $[0 < 2x = y < 1]$ ) and 23 open regions (e.g.  $[0 < 2x < y < 1]$ ).

*Definition 5 (slope<sub>max</sub>).* For each clock  $x \in H$ , we define  $\text{slope}_{\max(x)}$  as the largest value of  $\text{slope}_{xy}$  for all  $y \in H$ .

Reconsider the example above:  $\text{slope}_{\max(x)} = \max(\text{slope}_{xy}, \text{slope}_{xx}) = \max(\frac{2}{1}, 1) = 2$ ,  $\text{slope}_{\max(y)} = \max(\text{slope}_{yx}, \text{slope}_{yy}) = \max(\frac{1}{2}, 1) = 1$ . Note that if  $x$  is the fastest clock in  $H$  then  $\text{slope}_{\max(x)} = \text{slope}_{xx} = 1$ .  $\frac{1}{\text{slope}_{\max(x)}}$  is the smallest amount of time in which  $x$  cannot stay in the same region.

In the example, the clock  $x$  changes the region each half unit of time corresponding to  $\frac{1}{\text{slope}_{\max(x)}} = \frac{1}{2}$ , when  $y$  do this change each one unit of time (except for regions represented by points). The representation of a clock region accords with the two following points:

For each clock  $x$  which evolves according to  $\tau_p$ , there is one clock constraint taken from the set:

$$\begin{aligned} & \{x = c \mid c = 0, \frac{1}{\text{slope}_{xy}}, 2\frac{1}{\text{slope}_{xy}}, \dots, 1, 1 + \frac{1}{\text{slope}_{xy}}, 1 + 2\frac{1}{\text{slope}_{xy}}, \dots, c_x \text{ for all } y \in H\} \\ & \cup \left\{ \bigwedge_{y \in H} \left( c - \frac{1}{\text{slope}_{xy}} < x < c \right) \mid c = \frac{1}{\text{slope}_{xy}}, 2\frac{1}{\text{slope}_{xy}}, \dots, 1, 1 + \frac{1}{\text{slope}_{xy}}, 1 + 2\frac{1}{\text{slope}_{xy}}, \dots, c_x, \forall y \right\} \\ & \cup \{x > c_x\}. \end{aligned} \quad (1)$$

for each pair of clocks  $x$  and  $y$  which evolve respectively according to  $\tau_p$  and  $\tau_q$  such that  $c < x < c + \frac{1}{\text{slope}_{\max(x)}}$  and  $d < y < d + \frac{1}{\text{slope}_{\max(y)}}$  appear in (1) for some  $c$  and  $d$ , whether  $\text{slope}_{xy}(x - c)$  is less than, equal to or greater than  $y - d$ .

## 4.3 The Time-Successors of Clock Regions

In the following, we introduce the time-successor relation over clock regions. When time advances from any clock valuation  $v$  in the region  $\alpha$ , we will reach all its time-successors  $\alpha'$ . Formally, we say that  $\alpha'$  is a time-successor of the region  $\alpha$  if there are  $v$  in  $\alpha$ ,  $v'$  in  $\alpha'$ ,  $t \in \mathbb{R}_{>0}$  such that  $v' = v \oplus \tau(t)$ , with  $v \oplus \tau(t) = (v(x) + \tau_p(t))\pi^{-1}(x) = p$ .

For example, in Fig. 5 the five time-successors of the region  $\alpha = [(1.5 < x < 2), (1 < y < 2x - 2)]$  are: itself,  $[(x = 2), (1 < y < 2)]$ ,  $[(x > 2), (1 < y < 2)]$ ,  $[(x > 2), (y = 2)]$  and  $[(x > 2), (y > 2)]$ . These regions are those covered by a line drawn from any point in  $\alpha$  parallel to the line  $y = \text{slope}_{xy}x = 2x$  (in the upwards

direction). To compute each time-successor of a region  $\alpha$ , we must give : (i). for every clock  $x$ , a constraint of the form  $(x = c)$  or  $\left(c < x < c + \frac{1}{\text{slope}_{\max}(x)}\right)$  or  $(x > c_x)$  and (ii). for every pair  $x$  and  $y$  such that  $\left(c < x < c + \frac{1}{\text{slope}_{\max}(x)}\right)$  and  $\left(d < y < d + \frac{1}{\text{slope}_{\max}(y)}\right)$  appear in (i), the ordering relationship between  $\text{slope}_{xy}(x - c)$  and  $y - d$ .

To compute the possible time-successors, three cases are distinguished: **First case:** Each clock  $x$  in the region  $\alpha$  satisfies the constraint  $(x > c_x)$ , so  $\alpha$  has only one time-successor, itself. This is the case of region  $[(x > 2), (y > 2)]$  in Fig.5.

**Second case:** This case is considered when there is at least, in the region  $\alpha$ , one clock  $x$  which satisfies the constraint  $x = c$  for some  $c \leq c_x$ . The set  $H_0$  contains all clocks appearing in similar constraint form as  $x$ . The clock region  $\alpha$  will be changed immediately when the time advances, because the fractional part of each clock in  $H_0$  becomes different from 0. The clock regions  $\alpha$  and  $\beta$  have the same time-successors where  $\beta$  is specified by: A set of clock constraints which can be given as follows: For each clock  $x \in H_0$ : (i). If  $\alpha$  satisfies  $(x = c_x)$  then  $\beta$  satisfies  $(x > c_x)$ ; (ii). If  $\alpha$  satisfies  $(x = c)$  then  $\beta$  satisfies  $\left(c < x < c + \frac{1}{\text{slope}_{\max}(x)}\right)$ . For each clock  $x \notin H_0$ , the clock constraint in  $\alpha$  remains the same in  $\beta$ . The ordering relationship between  $\text{slope}_{xy}(x - c)$  and  $y - d$  of each pair of clocks  $x, y$  in  $\alpha$  is the same as that in  $\beta$ , such that  $x < c_x$  and  $y < c_y$  hold in the region  $\alpha$ .

For example, in Fig. 5 the time-successors of the region  $[(x = 0), (0 < y < 1)]$  are the same as the time-successors of the region  $[0 < 2x < y < 1]$ .

**Final case:** If the first and the second case do not apply, then let  $H_0$  be the set of clocks  $x$  for which the region  $\alpha$  satisfies two constraints:  $c < x < c + \frac{1}{\text{slope}_{\max}(x)}$  and  $\text{slope}_{xy}(x - c) \geq y - d$ .

For all clocks  $y$  for which the region  $\alpha$  satisfies  $d < y < d + \frac{1}{\text{slope}_{\max}(y)}$  Thus, when time advances, clocks in  $H_0$  take the values  $c + \frac{1}{\text{slope}_{\max}(x)}$ . Therefore, the time-successors of the region  $\alpha$  are  $\alpha, \beta$  and all the time-successors of  $\beta$  which is specified by :

- 1- A set of clock constraints which can be given as follows: (a) For each clock  $x \in H_0$ , if  $\alpha$  satisfies  $\left(c < x < c + \frac{1}{\text{slope}_{\max}(x)}\right)$  then  $\beta$  satisfies  $\left(x = c + \frac{1}{\text{slope}_{\max}(x)}\right)$ ; (b) For each clock  $x \notin H_0$ , the clock constraint in  $\alpha$  remains the same in  $\beta$ .
- 2- For each pair of clocks  $x$  and  $y$  such that  $\left(c < x < c + \frac{1}{\text{slope}_{\max}(x)}\right)$  and  $\left(d < y < d + \frac{1}{\text{slope}_{\max}(y)}\right)$  appear in (1.b), the ordering relationship between  $\text{slope}_{xy}(x - c)$  and  $y - d$  in  $\alpha$  remains the same in  $\beta$ .

For example, in Fig. 5 the time-successors of the region  $[0 < 2x < y < 1]$  include itself,  $[(0 < x < 0.5), (y = 1)]$  and all time-successors of  $[(0 < x < 0.5), (y = 1)]$ .

*Algorithm (region automaton).* Let  $A = ((S, s_0, H, L_S, T), \pi)$  be a daTA-RT over the set of agents  $Ag$ . The region automaton  $R(A)$  is an automaton over the alphabet  $Act$  such that: - The configurations of  $R(A)$  are of the form  $\langle s|\alpha \rangle$  where  $s$  is a state of  $A$  and  $\alpha$  is a clock region. The initial configuration is of the form  $\langle s_0|[v_0] \rangle$  where  $v_0(x) = 0$  for every  $x \in H$ . - A transition of  $R(A)$ , from the configuration  $\langle s|\alpha \rangle$  to  $\langle s'|\alpha' \rangle$ , is labeled by  $a \in Act$  iff there is a transition  $(s, G, a, x, s')$  in  $T$  and a clock region  $\alpha''$  which satisfies,  $\alpha''$  is a time-successor of the region  $\alpha$ , and  $\alpha' = [\{x\} \mapsto 0]\alpha''$ .

## 5 Conclusion

In this paper we have proposed an extension of the timed automata with action durations model, it has a capacity of describing timed plans with action durations that can be shared in between coordinated agents. We claim that also agents can be heterogeneous, they can reasonably be (re)synchronized to start the execution of any coordinated plan and that they can behave under relative time rates. Taking benefit from the semantics of such model, we demonstrated how to build a finite (time) region graph. The model is illustrated by a simple but realistic use case, wherein the coordination of truck is required. Agents are currently reduced to having one clock; however, the extension to several is immediate. *The implementability investigation:* In its current version our proposal is able to handle timed maximality bisimulation of behaviors; however region graph is not used for implementing practical tools because of the complexity of size and algorithms. A zone graph (based on convex polyhedra called zones) was proposed as an alternative for efficient implementations [10]; we intend to complete this work in this direction particularly in the sense of the scalability domain. As perspectives we intend to explore the ways of real applications and the formal comparison with other famous specification models such as petri net and its various extensions.

## References

1. Alur, R., Dill, D.: A theory of timed automata. *Theoretical Computer Science*, 183–235 (1994)
2. Kitouni, I., Hachichi, H., Bouaroudj, K., Saidouni, D.E.: Durational Actions Timed Automata: Determinization and Expressiveness. *International Journal of Applied Information Systems (IJ AIS)* 4(2), 1–11 (2012); Published by Foundation of Computer Science, New York, USA
3. Bolognesi, T., Brinksma, E.: Introduction to the ISO specification language LOTOS. *Computer Networks and ISDN Systems* 14, 25–59 (1987)
4. Courtiat, J.P., Saidouni, D.E.: Relating Maximality-based Semantics to Action Refinement in Process Algebras. In: *Proceedings of FORTE 1994*, pp. 293–308. Chapman and Hall (1995)
5. Lenzen, C., Locher, T., Wattenhofer, R.: Tight bounds for clock synchronization. In: *Proceedings of the 28th ACM Symposium on Principles of Distributed Computing*, pp. 46–55 (2009)

6. Pereverzeva, I., Troubitsyna, E., Laibinis, L.: Formal Development of Critical Multi-Agent Systems: A Refinement Approach. In: IEEE European Dependable Computing Conference (EDCC), pp. 156–161 (2012)
7. Guellati, S., Kitouni, I., Matmat, R., Saidouni, D.-E.: Timed Automata with Action Durations - From Theory to Implementation. In: Dregvaite, G., Damasevicius, R. (eds.) ICIST 2014. CCIS, vol. 465, pp. 94–109. Springer, Heidelberg (2014)
8. Corchuelo, R., Arjona, J.L.: A top down approach for MAS protocol descriptions. In: Proceedings of the 2003 ACM Symposium on Applied Computing, pp. 45–49 (2003)
9. Graja, Z., Migeon, F., Maurel, C., Gleizes, M.-P., Kacem, A.H.: A Stepwise Refinement based Development of Self-Organizing Multi-Agent Systems. In: Dalpiaz, F., Dix, J., van Riemsdijk, M.B. (eds.) EMAS 2014. LNCS, vol. 8758, pp. 40–57. Springer, Heidelberg (2014)
10. Bouyer, P., Laroussinie, F.: Model Checking Timed Automata. In: Modeling and Verification of Real-Time Systems, pp. 111–140. ISTE Ltd. John Wiley & Sons, Ltd. (2008)
11. Chaouche, A.-C., El Fallah Seghrouchni, A., Ilić, J.-M., Saïdouni, D.E.: A Higher-Order Agent Model with Contextual Planning Management for Ambient Systems. In: Kowalczyk, R., Nguyen, N.T. (eds.) TCCI XVI. LNCS, vol. 8780, pp. 146–169. Springer, Heidelberg (2014)
12. Layadi, S., Kitouni, I., Belala, N., Saidouni, D.E.: About Decidability of Dynamic Timed Automata with Relative Time Rates. Submitted in: IGI-Global International Journal of Embedded and Real-Time Communication Systems, IJERTCS (2014)
13. Dima, C., Lanotte, R.: Distributed time-asynchronous automata. In: Jones, C.B., Liu, Z., Woodcock, J. (eds.) ICTAC 2007. LNCS, vol. 4711, pp. 185–200. Springer, Heidelberg (2007)
14. Akshay, S., Bollig, B., Gastin, P., Mukund, M., Kumar, K.N.: Distributed Timed Automata with Independently Evolving Clocks. *Fundamenta Informaticae* 130(4), 377–407 (2014)

Computer Science and Its Applications

5th IFIP TC 5 International Conference, CIIA 2015, Saida,  
Algeria, May 20-21, 2015, Proceedings

Amine, A.; Bellatreche, L.; Elberrichi, Z.; Neuhold, E.J.;  
Wrembel, R. (Eds.)

2015, XXVI, 638 p. 238 illus. in color., Hardcover

ISBN: 978-3-319-19577-3