Electronic Notes in Theoretical Computer Science 42 (2001)
URL: http://www.elsevier.nl/locate/entcs/volume42.html 12 pages

Representation of a Discretely Controlled Continuous System in Tense Arithmetic

Tetsuya Mizutani [a], Shigeru Igarashi [a] and Masayuki Shio [b]

[a] *Institute of Information Science, University of Tsukuba, 1-1-1, Tennodai, Tsukuba 3050006, JAPAN*

[b] *College of Community Development, Tokiwa University, 1-430-1, Miwa, Mito 3108585, JAPAN*

Abstract

Specifications for a vehicle control system, as an example of discretely controlled continuous systems, are represented and verified in a formal system called *tense arithmetic* for concurrent programs with rational number time. In this formalism, computational sequences are characterized, or indexed, by *spurs* that are generalizations of program schedulers and that are also temporal propositions. Our formalism describes analysis of the wake-up time of the next action from an observation time, and we obtain the actual rational time value for when the next action will occur. Additionally, we introduce the *continuous variables* with their first- and second-order derivatives, to analyze and verify programs that control discretely certain continuously physical or other external systems.

1 Introduction

In this paper we represent and analyze a vehicle control system [6], as an example of discretely controlled continuous systems, in *tense arithmetic* [8] (TA for short) for the verification of concurrent programs involving rational number time. This formalism has three characteristics. First, it deals with rational time, called *tense*, explicitly. Second, it interprets each formula P in an extended theory of rationals by its tense, denoted by $\lfloor P \rfloor$, that is, the time length between an *observation time*, say t, and the time when P holds for the first time after t. (∞ is introduced as the abbreviation of $\lfloor \text{false} \rfloor$.) In addition, we introduce the *futurity* operator " ; " that forwards the observation time to the future. Third, the observation time t is treated as the *present tense* or *now*, i.e., 0, in the logic, so that we can dispense with the time variable t.

In this paper, we extend this formalism to treat continuous systems. To do this, the *continuous variables* with their first- and second-order derivatives will be introduced. Using these variables, we can analyze and verify programs that control discretely certain continuously physical or other external systems.

Many formalisms to verify realtime systems have been proposed, including the temporal logic [9], TCSP [4], the duration calculi [2], the hybrid automata [1], [3], [14], and so on, as well as our work [6], [7], [11] that is closely related to the present paper. Specifications of an autonomous vehicle control system are found and verified in [13]. The duration calculi and the hybrid automata can treat only first-order differential equations, while our formalism can treat both lower- and higher-order ones.

Section 2 shows the formal system and the semantics of TA, the original version of which can be found in [8]. A vehicle control system is then introduced and analyzed in Section 3. In this example, we consider the situation of a car merging with traffic at a T junction. We formulate this problem, then formally analyze and verify the conditions under which the merging car does not crash into others.

2 Tense arithmetic

2.1 Syntax of language

Let $T^{\mathcal{Q}}$ be any appropriate first-order theory of rational numbers. Let $V^{\mathcal{Q}}$ be the set of the free variables of $T^{\mathcal{Q}}$. An element of $V^{\mathcal{Q}}$ is called a *rational variable.*

We extend $T^{\mathcal{Q}}$ to $T_C^{\mathcal{Q}}$ by adding individual constants J, J_1, J_2, $\cdots$, corresponding to program variables over rational numbers, called *special rational constants*; by adding predicate constants l, l_1, l_2, $\cdots$, corresponding to labels within programs; and by adding predicate constants $\dot{\gamma}$, $\dot{\gamma}_1$, $\dot{\gamma}_2$, $\cdots$, corresponding to *atomic spurs* by which we construct *spurs.* Those predicate constants are called *special boolean constants.* The spurs are generalizations of program schedulers (see section 2.4). Moreover, we introduce other special rational constants ξ, ξ_1, ξ_2, $\cdots$, η, η_1, η_2, $\cdots$ called *continuous variables* with their (first- and second-order) derivatives ξ', ξ_1', ξ_2', $\cdots$, η', η_1', η_2', $\cdots$, ξ'', ξ_1'', ξ_2'', $\cdots$, η'', η_1'', η_2'', $\cdots$ to describe continuous systems.

Terms and formulas of TA are called *tense terms* and *tense formulas*, respectively, in order to distinguish them from those of $T_C^{\mathcal{Q}}$. Let V^T be the set of countably infinite tense variables.

Definition 2.1 Tense terms are defined as follows:

(i) A term e of $T_C^{\mathcal{Q}}$ is a tense term, which is called a *rational tense term*,

(ii) A tense variable x is a tense term,

(iii) For a formula P of $T_C^{\mathcal{Q}}$, $\lfloor P \rfloor$ is a tense term, called the *tense* of P,

(iv) If s is a tense term, then $s; e$, $s; x$ and $s; \lfloor P \rfloor$ are tense terms. □

The symbol $\lfloor\ \rfloor$ will be called the *tense symbol*, while the semicolon "$;$" as a symbol will be called the *futurity operator*, which is left-associative. s is called a *prefix* of $s; x$. In general, s is called a prefix of $s; s_1$.

Definition 2.2 Tense formulas are defined as follows:

(i) For a formula P of T_C^Q, P is a tense formula,

(ii) If s and s_1 are tense terms, then $s = s_1$ and $s \leq s_1$ are tense formulas,

(iii) If x is a tense variable, r is a rational variable, and F and G are tense formulas, then $\neg F$, $F \vee G$, $\forall x F$ and $\forall r F$ are tense formulas. □

A tense indicates the time relative to the observation time, called *now*. As a tense term, a rational expression e whose value is r represents the tense r. 0 represents now. The tense of P denoted by $\lfloor P \rfloor$ is the earliest time when P holds after now. For the futurity operator "$;$", here are a few examples:

(i) $2.0; \lfloor J = 0 \rfloor$ designates the tense when the program variable J reaches the value 0 after 2.0 time units, and

(ii) $2.0; \lfloor J = 0 \rfloor = \infty$ reads "J will never come to be 0 after 2.0".

2.2 Sequents and proof system

Definition 2.3 For tense formulas $F_1, \cdots, F_m, G_1, \cdots, G_n$ $(m, n \geq 0)$,

$$F_1, \cdots, F_m \rightarrow G_1, \cdots, G_n$$

is a *sequent* (of TA). The case that $m = 0$ is understood as 'true' and the case $n = 0$ as 'false'. □

A sequent $F_1, \cdots, F_m \rightarrow G_1, \cdots, G_n$ intuitively means that $F_1 \wedge \cdots \wedge F_m \supset G_1 \vee \cdots \vee G_n$ holds for any 'worlds' (changing with time) and at any observation time. Thus, the sequent expresses the same thing as the formula $\Box(F_1 \wedge \cdots \wedge F_m \supset G_1 \vee \cdots \vee G_n)$ in temporal logic.

Definition 2.4 The symbol ∞ is an abbreviation of $\lfloor \text{false} \rfloor$, so that

$$\infty = \lfloor \text{false} \rfloor,$$

and "$:$" is defined by

$$\begin{aligned}
x : (s \leq s_1) &\overset{\text{def}}{\Leftrightarrow} (x; s) \leq (x; s_1), & x : (s = s_1) &\overset{\text{def}}{\Leftrightarrow} (x; s) = (x; s_1), \\
x : (F \vee G) &\overset{\text{def}}{\Leftrightarrow} (x : F) \vee (x : G), & x : (\neg F) &\overset{\text{def}}{\Leftrightarrow} \neg(x : F), \\
\forall r(x : F) &\overset{\text{def}}{\Leftrightarrow} x : (\forall r F), & \forall y(x : F) &\overset{\text{def}}{\Leftrightarrow} x : (\forall y F), \textit{and} \\
x : P &\overset{\text{def}}{\Leftrightarrow} x = (x; \lfloor P \rfloor).
\end{aligned}$$

In the definitions of "$:$", each expansion rule on the left in the table precedes the one to the right, and each upper precedes the lower, if there exists ambiguity to expand $x : F$. □

The colon "$:$" as a symbol is called the *coincidental* operator, which intuitively means that the formula F is true at the tense x. As in $s; \lfloor P \rfloor$, s or any prefix within s is called a *prefix* of $s : F$.

Definition 2.5 The *inference rules* of TA consist of the LK-like rules having the same form as the structural rules and the rules for the logical operators $\neg$, $\vee$ and $\forall$ of LK, with the only modification being that the rules for $\forall x$ and $\forall r$ are differentiated from each other. Additionally, the *rule for futurity* is adopted as follows:

$$\frac{\rightarrow\ x : F}{\rightarrow\ F\ \wedge\ 0 \leq s_1\ \wedge \cdots \wedge\ 0 \leq s_n}$$

where $s_1, \cdots, s_n$ are all of the prefixes of tense terms, which are arguments of equalities or inequalities in F. $\square$

Definition 2.6 A *derivation* of TA consists of sequents arranged in tree form. An *initial sequent*, i.e., a leaf, of the derivation is any of the following:

(i) *identic sequent*: $F \rightarrow F$,

(ii) *axiomatic sequent*: $\rightarrow A$, for any axiom A of TA.

Each *inference* results from one of the inference rules by a usual substitution.

The *end sequent*, i.e., the root of the derivation, $\mathcal{S}$ is said to be *derivable* (in TA), and the whole derivation is called a *proof* of $\mathcal{S}$. $\square$

Definition 2.7 If a sequent of the form $\rightarrow F$ is derivable, we say that F is *provable* (in TA) and F is a *theorem* (of TA), which fact will be denoted by $\vdash F$. $\square$

2.3 Axioms

We introduce the logical, or, *proper* axioms of TA. It must be noted that r and r_1 are restricted within rationals (not containing ∞). The symbols $=$ and $\leq$ are *the same as those used in* $T^{\mathcal{Q}}$ *and* $T^{\mathcal{Q}}_C$, so that $0 < 1$, $\forall r r_1 (r + r_1 = r_1 + r)$, $J = J_1 \supset J + 1 = J_1 + 1$, etc., hold, for example.

(i) the ordinal *order axioms* for $\leq$.

(ii) axioms for tense:
- (a) *tense axiom*: $\forall x\ (\exists r\ (x = r)\ \vee\ x = \infty)$,
- (b) *'Never Land' (unreachability) axiom*: $\forall r (r < \infty)$,
- (c) *truth axiom*: $\lfloor \mathrm{true} \rfloor = 0$,
- (d) *futility axiom*: $\forall x\ (\infty = x; \infty)$,

(iii) axioms for futurity ";":
- (a) *present axiom*: $\forall x\ (0 \leq x \supset 0; x = x)$,
- (b) *passage axiom*: $\forall xy\ (x \leq x; y)$,
- (c) *duration axiom*: $\forall r r_1\ (0 \leq r_1\ \wedge\ r; r_1 = r + r_1\ \vee\ r_1 < 0\ \wedge\ r; r_1 = r)$,
- (d) *prefix substitution axiom*: $\forall xyz\ (x = y\ \supset\ x; z = y; z)$,

(iv) axioms for tense symbol $\lfloor\ \rfloor$:
- (a) *idempotent axiom*: $\lfloor P \rfloor = \lfloor P \rfloor ; \lfloor P \rfloor$,

(b) *precedence axiom*: $\forall x\ (0 \le x\ \wedge\ x : P\ \supset\ \lfloor P \rfloor \le x)$,
(c) *advance consequence axiom*: $\lfloor Q \rfloor \le \lfloor P \rfloor$, for any P and Q such that $P \supset Q$ is a theorem of $T_C^{\mathcal{Q}}$ that includes no continuous variable,
(d) *monotonicity axiom*: $\forall xy\ (x \le y\ \supset\ x; \lfloor P \rfloor \le y; \lfloor P \rfloor)$,

(v) theorems of $T_C^{\mathcal{Q}}$ as axioms: Any theorem of $T_C^{\mathcal{Q}}$ is an axiom of TA.

(vi) axiom for continuous variables:

$$\xi' = c_1\ \wedge\ \xi = c_2\ \wedge\ \forall r\ (0 \le r < \lfloor \xi'' \ne c \rfloor\ \supset\ r : (\xi'' = c))\ \supset$$
$$\forall r\ (0 \le r \le \lfloor \xi'' \ne c \rfloor\ \supset\ r : (\xi = \tfrac{c}{2} r^2 + c_1 r + c_2)\ \wedge\ r : (\xi' = cr + c_1))$$

From the axiom (vi), we can treat second-order differential equations in TA. Obviously, our formalism can treat higher-order derivatives when the axioms corresponding to them that are similar to (vi), are introduced.

2.4 Spurs and program axioms

Let us suppose that each atomic spur satisfies the condition that it returns to being false after it has become true within the finite time period.

Definition 2.8 Let $\dot\gamma$ be an atomic spur. A *spur* γ is $\lfloor \dot\gamma \rfloor; \lfloor \neg\dot\gamma \rfloor$, which expresses the time when $\dot\gamma$ changes into false after $\dot\gamma$ happens. □

Using spurs, we can represent each n-step execution of a process even if no actual time value of execution is given. For example, the tense of a 2-step execution can be written as $\gamma; \gamma$, which is an abbreviation of $\lfloor \dot\gamma \rfloor; \lfloor \neg\dot\gamma \rfloor$; $\lfloor \dot\gamma \rfloor; \lfloor \neg\dot\gamma \rfloor$. From axioms (4c) and (3a) with the rule for futurity, the fact that $\gamma < \gamma; \gamma$ is guaranteed, while $\lfloor \dot\gamma \rfloor = \lfloor \dot\gamma \rfloor; \lfloor \dot\gamma \rfloor$ by axiom (4a).

We express a program by some *program axioms* in the form of sequents, each of which represents one action step. Each program axiom is used as an initial sequent of a proof. A program axiom may contain a *spur*, a current program label, some next labels, some conditions and some actions, e.g., assignments. Additionally, we axiomatize the *axiom of conservation* [5], by which the values of the program variables are kept unchanged as long as no action is performed. The spurs are generalizations of schedulers of processes. We consider a multi-CPU parallel program system in which each process has its own CPU. So, we assign distinct spurs $\alpha,\ \beta,\ \cdots$ as the schedulers to the various processes. Program labels are supposed to be exclusive of each other process-wise.

2.5 Semantics

Let $\mathcal{Q}$ be the standard model of rational numbers. An assignment $\rho^{\mathcal{Q}}$ of $T^{\mathcal{Q}}$ is a function such that $\rho^{\mathcal{Q}} : V^{\mathcal{Q}} \to \mathcal{Q}$. $\mathcal{Q}$ as a model of $T^{\mathcal{Q}}$ can be expanded to a model $\mathcal{M}$ of $T_C^{\mathcal{Q}}$ by adding an interpretation of the special constants. Each special constant may have different values in different models.

We regard the set $\mathcal{Q} + \{\infty\}$ as tense. For ∞ we shall assume $r \in \mathcal{Q}\ \Leftrightarrow\ r < \infty$, $r + \infty = \infty + r = \infty$ every $r \in \mathcal{Q}$, and $\inf \emptyset = \infty$, $\emptyset$ denoting the empty set.

An assignment ρ of TA is a pair $\langle \rho^{\mathcal{Q}},\ \rho^{T}\rangle$ of functions, where $\rho^{\mathcal{Q}}$ is an assignment of $T^{\mathcal{Q}}$ as above and ρ^{T} is a function, called an *assignment to* V^T, such that $\rho^T : \mathcal{Q} \to V^T \to \mathcal{Q} + \{\infty\}$, sending a tense variable x onto $\rho^T(t,\ x)$. $\rho^{\mathcal{Q}}(e)$ and $\rho^{\mathcal{Q}}(P)$ denote the expression and the formula obtained from, respectively, e and P by substituting the values specified by the assignment $\rho^{\mathcal{Q}}$ in place of the free variables.

The changes of models are described by a *locus* χ that is a function from a rational time to models, $\chi :\ \mathcal{Q} \to \mathcal{E}$, where $\mathcal{E}$ is the set of such expansions designated by $\mathcal{M}$. For a special rational constant J and a special boolean constant l, $\chi(t,\ J)$ and $\chi(t,\ l)$ designate the rational value of J and the truth value of l at $t(\in \mathcal{Q})$ on χ, respectively. Whenever an assignment $\rho^{\mathcal{Q}}$ of $T^{\mathcal{Q}}$, a locus χ and a time $t(\in \mathcal{Q})$ are given, the truth of a formula P of $T^{\mathcal{Q}}_C$ is determined. $\rho^{\mathcal{Q}}, \chi(t) \models P$ expresses that P holds on $\rho^{\mathcal{Q}}$ and χ, at t, that is, $\chi(t,\ \rho^{\mathcal{Q}}(P)) = \text{true}$.

Definition 2.9 An increasing rational sequence $(r_i)_{i=0,1,2,\cdots}$ is *discrete* if it is either finite or contains arbitrarily large rationals. □

Definition 2.10 A function $f : \mathcal{Q} \to S$, where S is an arbitrary set, is called a *right-continuous discrete step function* if and only if $f(t) = x_i$ for $r_i \le t < r_{i+1}$ for some x_i belonging to S and a discrete rational sequence $(r_i)_{i=0,1,2,\cdots}$. □

Discreteness postulate. Every locus χ, except for continuous variables and their first-order derivatives, and every assignment ρ^T to V^T must be discrete step functions. □

In order to treat continuous variables and their derivatives, we modify the semantics of the original TA [8]. The main modification is that loci are *right-continuous* step functions, instead of *left*-continuous.

Postulate for continuous systems. We suppose that each "variable" in the continuous system that we deal with in this paper, as a function sending a real time in a real value [5], [12], is continuous in realtime and is derivable except for (finite or infinite) discrete rational time points. For each rational time value, the value of the function must be rational. Every locus must reflect the values of the continuous variables corresponding to these functions. Additionally, every first-order derivative must be continuous. □

Given an assignment ρ of TA, a locus χ and an observation time $t(\in \mathcal{Q})$, a tense term s is interpreted into a value in $\mathcal{Q} + \{\infty\}$ defined below.

Definition 2.11 The valuation $\sim$ of a tense term for given ρ, χ and t is defined as follows:

$$
\begin{array}{lrcl}
1. & e^{\sim}_{\rho,\chi}(t) & \stackrel{\text{def}}{=} & \chi(t,\ \rho^{\mathcal{Q}}(e)),\\
2. & x^{\sim}_{\rho,\chi}(t) & \stackrel{\text{def}}{=} & \rho^{T}(t,\ x),\\
3. & \lfloor P\rfloor^{\sim}_{\rho,\chi}(t) & \stackrel{\text{def}}{=} & \min\{u|\ 0\le u,\ \chi(t+u,\ \rho^{\mathcal{Q}}(P)) = \text{true}\},\ (\min\emptyset=\infty),\\
4. & (s;s_1)^{\sim}_{\rho,\chi}(t) & \stackrel{\text{def}}{=} & (s^{\sim}_{\rho,\chi}(t)<\infty\to\\
 & & & s^{\sim}_{\rho,\chi}(t) + (0\le {s_1}^{\sim}_{\rho,\chi}(t+s^{\sim}_{\rho,\chi}(t))\to {s_1}^{\sim}_{\rho,\chi}(t+s^{\sim}_{\rho,\chi}(t)),\ 0),\\
 & & & \infty).
\end{array}
$$

where s_1 is either e, x or $\lfloor P\rfloor$. $(a \ \to\ b,\ c)$ is McCarthy's operator [10], meaning "if a then b else c." □

Definition 2.12 The truth valuation # of a tense formula for given ρ, χ and t is defined as follows:

$$
\begin{array}{lrcl}
1. & P^{\#}_{\rho,\chi}(t) = \text{true} & \stackrel{\text{def}}{\Leftrightarrow} & \lfloor P\rfloor^{\sim}_{\rho,\chi}(t) = 0,\\
2. & (s=s_1)^{\#}_{\rho,\chi}(t) = \text{true} & \stackrel{\text{def}}{\Leftrightarrow} & s^{\sim}_{\rho,\chi}(t) = {s_1}^{\sim}_{\rho,\chi}(t),\\
 & (s\le s_1)^{\#}_{\rho,\chi}(t) = \text{true} & \stackrel{\text{def}}{\Leftrightarrow} & s^{\sim}_{\rho,\chi}(t) \le {s_1}^{\sim}_{\rho,\chi}(t),\\
3. & (\neg F)^{\#}_{\rho,\chi}(t) = \text{true} & \stackrel{\text{def}}{\Leftrightarrow} & F^{\#}_{\rho,\chi}(t) = \text{false},\\
4. & (F\vee G)^{\#}_{\rho,\chi}(t) = \text{true} & \stackrel{\text{def}}{\Leftrightarrow} & F^{\#}_{\rho,\chi}(t) = \text{true}\ \text{ or }\ G^{\#}_{\rho,\chi}(t) = \text{true},\\
5. & (\forall x F)^{\#}_{\rho,\chi}(t) = \text{true} & \stackrel{\text{def}}{\Leftrightarrow} & F^{\#}_{\langle\rho^{\mathcal{Q}},\ \rho'^{T}\rangle,\chi}(t) = \text{true},\quad \text{every } \rho'^{T}\\
 & & & \ \text{such that } \rho'^{T}(u,\ y) = \rho^{T}(u,\ y)\\
 & & & \ \text{each } u \text{ and each } y\in V^{T}\ \text{ other than } x,\\
6. & (\forall r F)^{\#}_{\rho,\chi}(t) = \text{true} & \stackrel{\text{def}}{\Leftrightarrow} & F^{\#}_{\langle\rho'^{\mathcal{Q}},\ \rho^{T}\rangle,\chi}(t) = \text{true},\quad \text{every } \rho'^{\mathcal{Q}}
\end{array}
$$

$$\text{such that } \rho'^{\mathcal{Q}}(v) = \rho^{\mathcal{Q}}(v)$$

$$\text{each } v\in V^{\mathcal{Q}}\ \text{ other than }\ r.\Box$$

Definition 2.13 A locus χ is said to *satisfy* F if and only if

$$F^{\#}_{\rho,\chi}(t) = \text{true},\qquad\qquad \text{every } \rho\ \text{ and } t,$$

in which case we write $\chi\models F$.

Let $\mathcal{P}$ be the set of loci satisfying the discreteness postulate. A locus $\chi\in\mathcal{P}$ is said to *satisfy* the sequent $F_1,\ \cdots,\ F_m\ \to\ G_1,\ \cdots,\ G_n$ if and only if it holds that

$$m+n>0 \textit{ and } \chi\models\ \neg F_1\ \vee\ \cdots\ \vee\ \neg F_m\ \vee G_1\ \vee\ \cdots\ \vee G_n.\ \Box$$

Definition 2.14 Soundness of derivation. For sequents $\mathcal{S}_1,\cdots,\mathcal{S}_n$ and $\mathcal{S}$ of TA, a derivation $\mathcal{S}$ from $\mathcal{S}_1,\cdots,\mathcal{S}_n$ of TA is *sound* if and only if every locus χ satisfying all the sequents $\mathcal{S}_1,\cdots,\mathcal{S}_n$ also satisfies $\mathcal{S}$.

The soundness theorem of the *original* TA is guaranteed in [8]. For the modified system, the following soundness theorem can be easily shown in a similar way.

Theorem 2.15 *If* $\vdash F$*, then* $\chi \models F$ *holds for every* χ*.* □

3 Analysis of a vehicle control system

Let us consider two roads meeting at a T junction. Many cars car_1, car_2, $\cdots$, car_n, car_{n+1}, $\cdots$ are running one way on the road at a constant speed v[m/s], and each pair of cars $\langle i,\ i+1\rangle$ has a distance within this pair, Δ_i. A car car_0 wants to merge between car_n and car_{n+1} with an initial speed 0 and an acceleration $a = a_0$ (> 0) [m/s^2] if Δ_n is less than or equal to d. The driver's decision to merge is delayed 0.5 [s] from the moment at which car_n reaches a certain "safe" zone, and the beginning of the actual acceleration of the vehicle is delayed 1.5 [s] from the moment of the driver's decision. Moreover, when the speed of the car is $v_0(< v)$, the driver decides to release the accelerator with 0.2 [s] delay, and the actual effect is delayed by 0.3 [s] after the decision. The car car_{n+1}, on the other hand, will decide to decrease its speed with a deceleration $-b{=}b_0$ (< 0) [m/s^2] after 0.4 [s] from the time when the distance Δ_0 between car_0 and itself will be less than or equal to e. The actual deceleration begins 0.6 [s] after the driver decides to decelerate.
Question: How large should d and e be to avoid a crash between car_0 and car_{n+1}? (The original problem and its solution are found in [6].)

3.1 Program axioms

Let us consider a one-dimensional coordinate along the road, whose origin is at the junction. The positions of car_0, car_n, car_{n+1} are expressed by η, ξ_n, ξ_{n+1}, respectively. We suppose car_0 is waiting at the crossing, that is, $\eta = 0$, and it finds the interval Δ_n is greater than or equal to d and car_n will be closer than or equal to $-f$. Additionally, let us suppose car_0 does not crash into any car when it is at the crossing and that its speed is 0.

The program axioms of car_0 are as follows:

(1) $$\begin{gathered} \mathit{Stand\text{-}by} \ \wedge\ \neg \mathit{Safe} \ \wedge\ \neg \mathit{Accel} \supset \\ \lfloor \mathit{Safe} \rfloor < \alpha \ \wedge\ \lfloor \mathit{Safe} \rfloor :\ (0.5 = \lfloor \mathit{Accel} \rfloor = \alpha). \end{gathered}$$

(2) $$\neg \mathit{Accel} \supset \lfloor \mathit{Accel} \rfloor :\ (1.5 \ =\ \lfloor \eta'' = a \rfloor \ =\ \alpha),$$

where, $\mathit{Stand\text{-}by} \equiv \eta = 0 \wedge \eta' = 0 \wedge \eta'' = 0$ and $\mathit{Safe} \equiv -f \leq \xi_n$. The axiom (1) intuitively means that when car_0 is at the crossing and finds car_n at f, the decision to accelerate (*Accel*), indicated by the spur α, is done with 0.5 [s] delay. In other words, if *Stand-by* holds, α arises as soon as *Accel* holds, and conversely, α does not hold before *Accel*. The axiom (2) means the actual acceleration begins 1.5 [s] after the decision *Accel*. Similarly, the deceleration after the merge is represented as follows:

(3) $\neg \mathit{OverSpeed} \supset \lfloor \mathit{OverSpeed} \rfloor : \ (0.2 = \lfloor \mathit{ReleaseAccel} \rfloor \ = \ \alpha)$.
(4) $\neg \mathit{ReleaseAccel} \supset \lfloor \mathit{ReleaseAccel} \rfloor : \ (\lfloor \eta'' = 0) \wedge \eta' = v \rfloor \ = \ \alpha \leq 0.3)$,

where $\mathit{OverSpeed} \equiv v_0 \leq \eta'$. This represents that when the speed of the car is $v_0 (< v)$, the car decides to release the accelerator with 0.2 [s] delay and the actual effect is delayed 0.3 [s] after the decision.

In a similar manner, we represent the axiom of car_n as

(5) $\mathit{Cruising}_n$,

and those of car_{n+1} as

$$\begin{aligned} &\mathit{Cruising}_{n+1} \ \wedge \ \neg \mathit{Brake} \ \wedge \ \neg \mathit{Dangerous} \ \supset \\ &\lfloor \mathit{Dangerous} \rfloor < \beta \ \wedge \ \lfloor \mathit{Dangerous} \rfloor : \ (0.4 = \lfloor \mathit{Break} \rfloor = \beta). \end{aligned} \tag{6}$$

(7) $\neg \mathit{Break} \supset \lfloor \mathit{Break} \rfloor : \ (0.6 \ = \ \lfloor \xi''_{n+1} = -b \rfloor = \beta)$,

where $\mathit{Cruising}_i \equiv \xi'_i = v \wedge \xi''_i = 0 (i = n, \ n+1)$ and $\mathit{Dangerous} \equiv \Delta_0 \leq e$.

3.2 Analysis

Let us consider an initial condition in which $\Theta \equiv \mathit{Stand\text{-}by} \wedge \ \neg \mathit{Safe} \wedge \ \neg \mathit{Accel} \wedge \ \neg \mathit{ReleaseAccel} \wedge \ \neg \mathit{Brake} \wedge \ d \leq \Delta_n$.

From (1) and (2) of the axioms of car_0 and the axiom of conservation, it holds that

$$\begin{aligned} \Theta \ \supset \ \eta = 0 \ \wedge \ \eta' = 0 \ \wedge \ \forall r \ (0 \leq r < \lfloor \eta'' = a \rfloor \supset r : \ (\eta'' = 0)) \\ \wedge \ \lfloor \mathit{Safe} \rfloor : \ (\lfloor \eta'' = a \rfloor = 0.5 + 1.5). \end{aligned} \tag{8}$$

By the axiom of continuous variables,

$$\Theta \ \supset \ \forall r \ (0 \leq r \leq \lfloor \eta'' = a \rfloor \supset r : \ (\eta = 0) \ \wedge \ r : \ (\eta' = 0)). \tag{9}$$

Thus,

$$\Theta \ \supset \ \lfloor \eta'' = a \rfloor : \ (\eta = 0 \ \wedge \ \eta' = 0). \tag{10}$$

From this fact and from (3),

$$\begin{aligned} &\Theta \ \supset \\ &\lfloor \mathit{Safe} \rfloor ; 2 : \ (\eta = 0 \ \wedge \ \eta' = 0 \ \wedge \ \forall r \ (0 \leq r < \lfloor \eta'' = 0 \rfloor \ \supset \ \eta'' = a)). \end{aligned} \tag{11}$$

Therefore, we have

$$\begin{aligned} &\Theta \ \supset \\ &\lfloor \mathit{Safe} \rfloor ; 2 : \ \forall r \ (0 \leq r \leq \lfloor \eta'' = 0 \rfloor \ \supset \ r : (\eta = \tfrac{a}{2} r^2 \ \wedge \ r : \eta' = ar)). \end{aligned} \tag{12}$$

from the axiom of continuous variables. This represents the position and speed of car_0 once it has accelerated.

Furthermore,

$$(13)\qquad \begin{aligned}&\neg \mathit{OverSpeed} \ \supset \\ &\lfloor \mathit{OverSpeed} \rfloor :\ \forall r(0 < r \leq \lfloor \eta'' = 0 \rfloor \ \supset \eta' = ar + v_0) \\ &\wedge \lfloor \mathit{OverSpeed} \rfloor :\ (0.2 < \lfloor \eta'' = 0 \rfloor \leq 0.2 + 0.3).\end{aligned}$$

by (12), (3) and (4). Therefore, we have $v - 0.5a \leq v_0 < v - 0.2a$ since $\eta' = v$.

On the other hand, after deceleration begins the car drives as

$$(14)\qquad \begin{aligned}&\Theta \ \supset \\ &\lfloor \eta'' = 0 \rfloor :\ (\eta = c_1 \ \wedge \ \eta' = v \ \wedge \\ &\forall r \ (0 \leq r \ \supset \ r : (\eta = vr + c_1) \ \wedge \ r : (\eta' = v))).\end{aligned}$$

Thus, $c_1 = \frac{v^2}{2a}$ holds by (12).

Similarly, we have the behavior of car_{n+1} as

$$(15)\qquad \begin{aligned}\Theta \ \wedge \ \Delta_n = r_1 \supset \ \lfloor \mathit{Safe} \rfloor :\ (\xi_{n+1} = -f - r_1 \ \wedge \ \xi'_{n+1} = v) \\ \wedge \ \forall r \ (0 \leq r \leq \lfloor \xi''_{n+1} = -b \rfloor \ \supset \\ r : (\xi_{n+1} = vr - f - r_1) \ \wedge \ r : (\xi'_{n+1} = v)),\end{aligned}$$

$$(16)\qquad \begin{aligned}&\Theta \ \supset \ \lfloor \mathit{Dangerous} \rfloor ; 1 \ : \\ &(\xi_{n+1} = c_2 \ \supset \\ &\forall r (0 \leq r \ \supset \ r : (\xi_{n+1} = -\tfrac{b}{2} r^2 + vr + c_2) \ \wedge \ r : (\xi'_{n+1} = v - br))),\end{aligned}$$

We solve the minimum values of d and e. If car_0 decides to accelerate when $f = 2v$, and the condition $d = e$ holds, then these are the minimum values. From (15) and (16), we can get:

$$(17)\qquad \begin{aligned}&\Theta \ \supset \\ &\lfloor \mathit{Safe} \rfloor ; 2 + 1 : \\ &\forall r \ (0 \leq r \ \supset \ r : (\xi_{n+1} = -\tfrac{b}{2} r^2 + vr + v - d) \ \wedge \ r : (\xi'_{n+1} = v - br)),\end{aligned}$$

which indicates the position and the speed of car_{n+1}.

From this formula with (8) under the conditions $\eta = \xi_{n+1}$ and $\eta' = \xi'_{n+1}$, i.e., car_0 has just collided with car_{n+1}, we have the minimum values of d and e whenever the cars do not crash as:

$$(18)\qquad d = e = \frac{ab - 2bv + v^2}{2(a - b)}.$$

Finally, we calculate and get the solution when the actual (and reasonable) values are given. For example, suppose $a = -b = 5$[m/s^2] and $v = 50/3$[m/s] (60 km/h). Hence, we have a solution that if $d = e > 755/36$[m] (about 21m), car_{n+1} does not crash into car_0.

4 Conclusions

We have briefly demonstrated a new formalism TA, for parallel and realtime controlled programs, to analyze two intelligently controlled systems, briefly. In the analysis of a vehicle control system, we have shown, using the axiom of continuous variables, the actual and reasonable values necessary to avoid a collision between two vehicles.

Our future work is that we will make the axiom more sophisticated, in order to describe every n–th order derivative in a uniform manner.

Acknowledgements

This work was supported in part by Grants-in-Aid for Scientific Research from the Japanese Ministry of Education, Science and Culture (Nos. 09680322 and 10680332) and by Fuji Xerox.

References

[1] Alur, R., Fix, L. and Henzinger, T. A. : Event-clock automata: a determinizable class of timed automata, *Theor. Comp. Sci.*, (1999), pp. 252–273.

[2] Chaochen, Z.: Duration calculi: an overview, *International Institute for Software Technology, The United Nations University, UNI/IIST Report*, **10** (1993).

[3] Henzinger, T. A. and Kopke, P. W.: Discrete-time control for rectangular hybrid automata, *Theor. Comp. Sci.*, **211** (1999), pp. 369–392.

[4] Hoare, C. A. R. : Communicating sequential processes, *Prentice-Hall Internat.*, 1985.

[5] Igarashi, S : The ν-conversion and an analytic semantics, R. E. A. Mason (ed.), *Information Processing 83, Elsevier Sci. Publ. B. V.,* (1983), pp. 769–774.

[6] Igarashi, S., Mizutani, T., Shirogane, T. and Shio, M : Formal analysis for continuous systems controlled by programs, *Concurrency and Parallelism, Programming, Networking, and Security, Lecture Notes in Comp. Sci.,* **1179** (1996), pp. 347–348.

[7] Igarashi, S, Shio, M., Shirogane , T. and Mizutani, T. : Formal verification and evaluation of execution time in the envelope theory, *Concurrency and Parallelism, Programming, Networking, and Security, Lecture Notes in Comp. Sci.,* **1179**, (1996), pp. 299–308.

[8] Igarashi, S., Shirogane, T., Shio, M. and Mizutani, T. : Tense arithmetic I: formalization of properties of programs in rational arithmetics, *Tensor, N. S.*, **59** (1998), in printing.

[9] Kröger, F : Temporal logic of programs, *Springer-Verlag, New York*, 1987.

[10] McCarthy, J. : A basis for a mathematical theory of computation, P. Braffort and D. Hirschberg (eds.), *Computer Programming and Formal Systems, Series of Studies in Logic and the Foundations of Mathematics*, *North-Holland Publ. Co., Amsterdam*, 1963, pp. 33–70.

[11] Mizutani, T., Igarashi, S., Tomita, K. and Shio, M. : Representation of discretely controlled continuous systems in software-oriented formal analysis, *Advances in Computer Science, Lecture Notes in Comp. Sci.*, **1345** (1997), pp. 110–120.

[12] Takeuti, G : Two applications of logic to mathematics, *Princeton Univ. Press, Princeton*, 1978.

[13] Tomita, K., Igarashi, S., Hosono, C., Mizutani, T. and Tsugawa, S. : Representations of Autonomous Realtime Systems, *TENSOR, N. S.*, **59** (1998), in printing.

[14] Xuandong, L., Hung, D. V. and Tao, Z : Checking hybrid automata for linear duration invariants, *Advances in Computer Science*, *Advances in Computer Science, Lecture Notes in Comp. Sci.*, **1345** (1997), pp. 166–180.