ON DISCRETISABLE FORMULAS IN DURATION CALCULUS

Pham Hong Thai

Faculty of Technology, VNU

Abstract Model checking problem for real-time systems is a hard problem and has high complexity because time model of system is dense and continuous. Especially, as known, almost accumulated timed properties which are expressed by duration formulas in Duration Calculus is undecidable or decidable but with very high complexity. However, fortunately for some formulas, to avoid high complexity we can only check them in integral model of time instead of real time model. Such formulas are called discretisable formulas. In this paper, we show a subclass of formulas in Duration Calculus which is constructed from a linear constraint of state durations is discretisable and based on this we also give some ideas for checking them. The our results includes some results of the others.

1. Introduction

Discrete time model of real-time systems was considered widely in recent years. A reason of the consideration is as many verification problems in dense time model are undecidable, even for decidable problems, its complexity is also very high. In the other hand, techniques for verifying real-time systems in discrete time model are simpler and have lower complexity. Such verification methods are based on the assumption that states are observed at integer time points only. A wide class of integral-time verification methods have been shown as model-checking algorithms (eg. [3]) or theorem proving systems [4]...

However, it will be better if answer to verifying in discrete time model also supplies us the answer to dense time model. That means if a property is true in the discrete time model then it is also correct in dense time model. Such properties are called *discretisable properties* and instead of verifying in dense time we only verify them in integer time by simpler techniques and lower complexity.

With this aim in [7] the authors constructed discretising models of timed automata in which generated untimed sequences of symbols are the same as in original model. Or in [5] Thomas Henzinger et al. proved some properties such as time-bounded invariance and time-bounded response are discretisable. These properties is only concerned to instant time of systems and are called instant properties, for example reachability property in [7] and time-bounded reachability in [5].

How about are duration properties ? What properties of them are discretisable? Duration properties are properties concerning to accumulated time of states of system. For these properties, Zhou Chaochen et al. proposed and advanced a logic is called *Duration Calculus* [10] in which these properties can be expressed and calculated. As an example, Linear Duration Invariant (LDI) is a formula in Duration Calculus and is mentioned at first in [11]. This formula expresses a property of real-time systems as "in any observation

for system, if the (time) length of observation interval belongs in a certain interval [B, E] then the time durations of states of the system have to satisfy a certain linear constraint". Many real-time requirements in the practice can be expressed by LDI, for example safety properties of gaz burner [10]. railroad crossing system [14].

There were many works dealing with LDI and its subclass. Model checking algorithms in these works concentrate on two ways : in first one, system is represented by timed regular expressions [11-14] and model checking problem is reduced to solving linear programming problems. In the other one integral region graph of automata is used to solve problem if checking property is discretisable [15] or combine both methods [16,17]. However, most of them only deals with restricted systems as real-time automata, subclass of models of Duration Calculus ... or for subclasses of LDI. For example, *"Duration bounded reachability property"* which was observed in [2]. This is a formula that is the same as LDI but coefficients in the formula are restricted to positive reals only. In [12] the authors proved discretisability of *Linear Duration Constrain* - LDC (a subclass of LDI) with integral coefficients. By a different technique, the authors in [15] proved LDC with real coefficients is also discretisable.

In this paper we prove a lager class of formulas (including LDI) is discretisable. For this. we consider LDC with semantics larger than in [15]. In [15] authors considered LDC with observations for system is started and ended at time points at which transitions of system is taken. In this paper, starting and ending time points of an observation are arbitrary. It is important focus for ability extending proof of discretisability of LDC to LDI and some other formulas.

The remainder of the paper is organized as follows. In the next section we recall some notations of real-time systems as timed automata, duration formulas as LDC and notion of discretisability. In section 3 we give proof discretisability of LDC and based on this in section 4 we prove discretisability of LDI and some others duration formulas. At final, in conclusion we give a short discussion about ability of checking LDI by zone graph of timed automata.

2. Model of Real-Time Systems and Properties

2.1 Timed Automata

In this paper we get timed automata as model of real-time systems. As timed automata have become typical and have been deliberated very well, so in this section we only present summarily about them, the details readers is referred to [6].

A timed automaton has a finite set of states S and a finite set of clock X which are real value variables. Each state transition of automaton is assigned by a time constraint as enabled condition and a subset of clocks which is called reset set. The time constraint represents requirement that a transition may be taken only if the current values of the clocks satisfy this constraint. And, the reset set shows that all clocks in it are reset to zero when transition is taken. Transitions are taken instantaneous, while time can elapse at states of timed automata. The value of a clock equals the time elapsed since the last time it was reset.

Let $\Phi(X)$ be set of time constraints ϕ, which are conjunctions of the simple constraints of form $x \leq c \mid c \leq x \mid x - y \leq c \mid c \leq x - y$ where $x, y \in X$ and c is a natural constant.

As often. we denote sets of natural and nonnegative real number by $\mathbf{N}$ and $\mathbf{R}^+$, respectively. Formally, timed automata can be defined as follows.

Definition 1.[Timed Automata] A timed automaton $\mathcal{A}$ is a tuple $\langle S, s_0, \Sigma, X, E\rangle$, where

- S is a finite set of states,
- s_0 is an initial state,
- Σ is a finite set of symbols,
- X is a finite set of clocks,
- $E \subseteq S \times \Phi(X) \times \Sigma \times 2^X \times S$ is a finite set of transitions. A transition $\langle s, \phi, a, \lambda, s'\rangle \in E$ represents that if system is staying at state s and current values of clocks satisfy time constraint ϕ then system can transit to state s' and then the clocks in λ must be reset to zero. The transition causes an event which be denoted by symbol a.

Definition 2.[Behaviors] A *behavior* of timed automaton $\mathcal{A}$ is a infinite sequence of timed states

$$\rho : (s_0, t_0)(s_1, t_1) \ldots (s_m, t_m) \ldots$$

that satisfies following conditions

1. s_0 is initial state of timed automaton $\mathcal{A}$, $t_0 = 0$.
2. time does not decrease, i. e. $t_i \leq t_{i+1}$ for all $i \geq 0$.
3. time progresses, i. e. for any $T \in \mathbf{R}^+$, there is some $i \geq 0$ such that $t_i \geq T$.
4. t_i is time point that system changes its state to s_i, for all $i \geq 0$. That means, the system stays at s_{i-1} in $d_i = t_i - t_{i-1}$ time units and then transits to s_i by some transition $\langle s_{i-1}, \phi, a, \lambda, s_i\rangle$.

In this paper behavior of timed automata is considered as a sequence of time states instead of sequence of time transition as in other papers, however semantics of timed automata is not changed. In the other hand, we only consider discretising of time points so we do not discuss about events (i.e symbols in Σ) here.

A behavior is called *integral behavior* iff for all $i \geq 0, t_i$ is integral.

Example 1. Sequences of timed states $\rho_1 = (s_0, 0)(s_1, 2.3)(s_2, 3.0)(s_3, 4.2) \ldots$ and $\rho_2 = (s_0, 0)(s_1, 2)(s_2, 3)(s_3, 5) \ldots$ are behaviors of some timed automaton, where ρ_2 is integral behavior.

Definition 3.[Observations] Let $b, e \in \mathbf{R}^+$ are two timed points with $0 \leq b \leq e < \infty$. An *observation on interval* $[b, e]$ ($\sigma_{[b,e]}$) of a behavior ρ is any part of ρ that it starts at time point b and ends at time point e. An observation is called *integral* if for all time point t_i and two endpoints b, e of it are integral values. $\ell = e - b$ be called the length (of time) of observation $\sigma_{[b,e]}$.

For simplicity of notations sometimes we also call observation σ on interval $[b, e]$ by observation σ for short.

Given an observation $\sigma_{[b,e]}$ of a behavior ρ, item 3. in definition 2 guarantees that our system is nonZeno system [6], i.e. in any observation interval of system it has only

finite number of states. Hence, $\sigma_{[b,e]}$ can be formally expressed as a finite sequence of time-states with two timed bounds b, e as follows

$$\sigma : (s_{u-1}, t_{u-1})\ b\ (s_u, t_u)(s_{u+1}, t_{u+1}) \ldots (s_v, t_v)\ e\ (s_{v+1}, t_{v+1})$$

where $1 < u \leq v$, b $(t_{u-1} \leq b \leq t_u)$ is beginning time point of observation before the system transits to state s_u and e $(t_v \leq e \leq t_{v+1})$ is ending time point of observation after the system transits to and stays at state s_v. That means state s_{u-1} occurs in $t_u - b$ time units before the system transits to state s_u, and similarly state s_v appears in $e - t_v$ time units after the system transits to state s_v on σ. Figure 1 illustrates an observation σ in time interval $[b, e]$ of timed automata $\mathcal{A}$.

Fig 1. The observation σ on time interval $[b, e]$

Let $\sigma : (s_{u-1}, t_{u-1})\ b\ (s_u, t_u)(s_{u+1}, t_{u+1}) \ldots (s_v, t_v)\ e\ (s_{v+1}, t_{v+1})$ be an observation on interval $[b, e]$. Then accumulated time that the system stays at state s in time interval $[b, e]$ can be calculated by

$$d_s = \sum_{j=u-1,\ s_j=s}^{v} (t'_{j+1} - t'_j),$$

where $t'_{u-1} = b,\ t'_j = t_j\ (\forall j = u..v),\ t'_{v+1} = e.$

2.2 Formulas in Duration Calculus

Properties (or timed requirements) of real-time systems is often specified by formulas in some real-time logics as temporal logic [1], duration calculus - DC [10]. In this paper we consider duration properties that are properties saying about accumulated time of states and are expressed by formulas of DC. Duration Calculus is a real-time logics and well-known as a logic expressing such duration properties, however it is not presented here. We will directly represent subclasses of formulas in Duration Calculus which are compositions of simpler formulas called Linear Duration Constraint and it is not hard to understand semantics of these formulas.

Definition 4. [Linear Duration Constraint - LDC] Given a timed automaton $\mathcal{A}$ with the set of states S. A linear duration constraint over S is a formula φ of the form :

$$\varphi : \sum_{i=1}^{m} c_i \int s_i \leq M,$$

where coefficients c_i, M are real numbers, $s_i \in S$. $\int s$ (is said be *duration* of s, one of operators in DC) denotes the accumulated time of state s that it occurs in some time interval.

As semantics, LDC represents a property of system which can be informally understood as follows : In any observation time interval of system, presence time durations d_{s_i} of states s_i must satisfy a linear constraint as expression $\sum_{i=1}^{m} c_i d_{s_i} \leq M$. In this semantics system is observed on time interval $[b, e]$ with the endpoints b, e is arbitrary.

2.3. Discretisability

Given a timed automaton $\mathcal{A}$ and a property P, a question is : whether system $\mathcal{A}$ satisfies property P or not ? A system is called satisfying property P if P is evaluated to true on all behaviors of system. There were many methods to solve this problem e.g. model checking algorithms that most of them is used to check properties expressed in timed computational tree logic (TCTL)[8]. Results in field of checking DC formulas are rarely now. Reason of this situation is because potential complexity of checking problem DC formulas is very high. As we known almost of DC formulas is undecidable. Undecidability and high complexity come from real model of time and accumulation of time (on states) of timed requirements. Even under discrete time model, class of decidable duration formulas which was known up to now has still been very small [18].

So for avoiding high complexity whether we can check satisfiability of property for system only on integral behaviors instead of real behaviors. For some properties, this is available, they are called discretisable properties.

Definition 5.[Discretisability] A real-time property P of timed automaton $\mathcal{A}$ is said *discretisable* iff the property P is satisfied by the $\mathcal{A}$ exactly when P is satisfied by all the integral behaviors of $\mathcal{A}$.

The our purpose in this paper is finding class of such formulas in DC. At first, we consider Linear Duration Constraint which is presented in above paragraph. Proof of discretisability of this formula was given in [15]. However, in the next section, we give another proof for advanced semantics of the formula in our paper.

3. Discretisability of LDC

3.1. Notion of ϵ-discretising and Some Properties

Definition 6.[ϵ-discretising] Given positive reals x and $\epsilon\,(0 \leq \epsilon < 1)$. x_ϵ is an integer which defined from x as follows

$$x_\epsilon = \begin{cases} \lfloor x \rfloor & \text{if fraction of x is less than or equal } \epsilon \\ \lceil x \rceil & \text{otherwise.} \end{cases}$$

That is, x will be rounded to floor or ceiling of x depending on values of fraction of x and ϵ. For example, if $x = 4.38$, then $x_{0.3} = 5$ and $x_{0.42} = 4$.

Lemma 1. *Given $a \leq b$ are two integer numbers and t_i, t_j are nonnegative real numbers, where $t_i \geq t_j$. Then we have*

$$a \leq t_i - t_j \leq b \Leftrightarrow a \leq t_{i\epsilon} - t_{j\epsilon} \leq b, \forall \epsilon \in [0,1)$$

Proving the lemma is easily, so we do not present it here.

As a consequence of the lemma, if $t_i \geq t_j$ then $t_{i\epsilon} \geq t_{j\epsilon}, \forall \epsilon \in [0,1)$ (applying lemma with $a = 0$), that means under ϵ-discretising temporal order of states occurring in a behaviors is not changed.

Lemma 2. *Given $\{\alpha_i\}, \{\beta_i\}$ $(i = 1..n)$ are sequences of positive real numbers, where sequence α_i is not decrease and sequence β_i is not increase ($0 < \alpha_1 \leq \alpha_2 \leq \ldots \leq \alpha_n$, $\beta_1 \geq \beta_2 \geq \ldots \geq \beta_n > 0$). Let $\{A_i\}(i = 1..n)$ be a sequence of real numbers which has the property : sum of each really prefixes of sequence is positive. That is $\sum_{i=1}^{v} A_i > 0, (1 \leq v \leq n-1)$. Then we have*

1. $\sum_{i=1}^{n} A_i \leq 0 \Rightarrow \sum_{i=1}^{n} \alpha_i A_i \leq 0,$
2. $\sum_{i=1}^{n} A_i \geq 0 \Rightarrow \sum_{i=1}^{n} \beta_i A_i \geq 0$

Proof.

1. Assume that $\sum_{i=1}^{n} A_i \leq 0$. Let $\Lambda = \sum_{i=1}^{n} \alpha_i A_i = \alpha_1 A_1 + \alpha_2 A_2 + \cdots + \alpha_n A_n$. As $\alpha_1 \leq \alpha_2$ and $A_1 > 0$ so $\Lambda \leq \alpha_2 A_1 + \alpha_2 A_2 + \cdots + \alpha_n A_n = \alpha_2(A_1 + A_2) + \alpha_3 A_3 + \cdots + \alpha_n A_n$. Similarly, as $\alpha_2 \leq \alpha_3$ and $A_1 + A_2 > 0$ so $\Lambda \leq \alpha_3(A_1 + A_2 + A_3) + \alpha_4 A_4 + \cdots + \alpha_n A_n$, ... and so on ... finally, we have $\Lambda \leq \alpha_n(A_1 + A_2 + \cdots + A_n) \leq 0$.
2. Assume that $\sum_{i=1}^{n} A_i \geq 0$. Let $\Lambda = \sum_{i=1}^{n} \beta_i A_i = \beta_1 A_1 + \beta_2 A_2 + \cdots + \beta_n A_n$. As $\beta_1 \geq \beta_2$ and $A_1 > 0$ so $\Lambda \geq \beta_2 A_1 + \beta_2 A_2 + \cdots + \beta_n A_n = \beta_2(A_1 + A_2) + \beta_3 A_3 + \cdots + \beta_n A_n$. Similarly, as $\beta_2 \geq \beta_3$ and $A_1 + A_2 > 0$, so $\Lambda \geq \beta_3(A_1 + A_2 + A_3) + \beta_4 A_4 + \cdots + \beta_n A_n$, ... and so on ... Finally we have $\Lambda \geq \beta_n(A_1 + A_2 + \cdots + A_n) \geq 0$.

Lemma 3. *Given $\{a_i\}$, $\{t_i\}$, $(i = 1..m)$ are two sequences of any real numbers, where $t_i \geq 0, \forall i = 1..m$. Then we always find a real number $\epsilon \in [0,1)$ such that*

$$\sum_{i=1}^{m} a_i t_i \leq \sum_{i=1}^{m} a_i t_{i\epsilon}$$

Proof. Let $\{f_0, f_1, f_2, \ldots, f_q\}$ be a set of fractions of real numbers t_i $(i \in I = \{1, 2, \ldots, m\})$, such that $0 = f_0 < f_1 < f_2 < \cdots < f_q < 1$. Let $I_k, (k = 0..q)$ be a set of indexes of t_i's such that fraction of t_i equals to f_k, that is $I_k = \{i \in I | \delta_i = f_k\}$, where δ_i stands for the fraction of t_i. Let $A_k = \sum_{i \in I_k} a_i$ $(k = 0..q)$.

Now let us partite the sequence $\{A_k\}_{k=1}^{q}$ to d+1 successive segments

$$\{A_1, A_2, \ldots, A_{k_1}\}, \{A_{k_1+1}, A_{k_1+2}, \ldots, A_{k_2}\}, \ldots, \{A_{k_{d-1}+1}, A_{k_{d-1}+2}, \ldots, A_{k_d}\},$$
$$\{A_{k_d+1}, A_{k_d+2}, \ldots, A_q\}$$

such that for each segment the hypothesis about A_i's of Lemma 2 is satisfied. That is indexes $k_1, k_2, \ldots, k_d$ is defined such that sum of A_i's in each really prefix of each segment is greater than 0 and sum of all A_i's in each segment is less than or equal to 0. In general, sum of all A_i's in last segment ($(d+1)^{th}$ segment) is greater than 0. It is easily to see that the indexes $k_1, k_2, \ldots, k_d$ can be found by the following procedure

```
i = 1; sum = 0;
for (k = 1; k ≤ q; k++)
{
  sum += A_k;
  if (sum ≤ 0) { k_i = k; sum = 0; i++; }
}
```

For simplicity, let $p = k_d$. So, in general, p $(0 < p < q)$ divides sequence $\{A_k\}_{k=1}^{q}$ to two parts. The first one consists of d segments, sum of A_i's of each segment is less than or equal to 0. The second one consists of rest A_i's (from A_{p+1} to A_q) and their sum is a positive number. Concretely

$$\sum_{k=k_i+1}^{k_{i+1}} A_k \leq 0 \ (i = 0..d-1), \text{ (with convention } k_0 = 0) \text{ and } \sum_{k=p+1}^{q} A_k > 0.$$

Hence, by applying the Lemma 2 we have

$$\sum_{k=k_i+1}^{k_{i+1}} f_k A_k \leq 0 \Rightarrow \sum_{k=1}^{p} f_k A_k = \sum_{i=0}^{d-1} \sum_{k=k_i+1}^{k_{i+1}} f_k A_k \leq 0, \text{ and } \sum_{k=p+1}^{q} (1-f_k) A_k > 0.$$

From above result it implies that

$$-\sum_{k=1}^{p} f_k A_k + \sum_{k=p+1}^{q} (1-f_k) A_k > 0$$

Now, to prove the lemma, let $\epsilon = f_p$. Then we have

- $t_{i\epsilon} = \lfloor t_i \rfloor = t_i - \delta_i$ if $\delta_i \leq \epsilon = f_p$, i.e. if $i \in I_1 \cup I_2 \cup \ldots \cup I_p$, and
- $t_{i\epsilon} = \lceil t_i \rceil = t_i - \delta_i + 1$ if $\delta_i > \epsilon = f_p$, i.e. if $i \in I_{p+1} \cup I_{p+2} \cup \ldots \cup I_q$.

Therefore,

$$\begin{aligned}
\sum_{i=1}^{m} a_i t_{i\epsilon} - \sum_{i=1}^{m} a_i t_i &= -\sum_{i \in I_1 \cup \ldots \cup I_p} a_i \delta_i + \sum_{i \in I_{p+1} \cup \ldots \cup I_q} a_i (1-\delta_i) \\
&= -f_1 \sum_{i \in I_1} a_i - \cdots - f_p \sum_{i \in I_p} a_i + \\
&\quad + (1-f_{p+1}) \sum_{i \in I_{p+1}} a_i + \cdots + (1-f_q) \sum_{i \in I_q} a_i \\
&= -\sum_{k=1}^{p} f_k A_k + \sum_{k=p+1}^{q} (1-f_k) A_k \geq 0.
\end{aligned}$$

In the rest cases, if $p = 0$, we can easily see that

$$\sum_{i=1}^{m} a_i t_{i\epsilon} - \sum_{i=1}^{m} a_i t_i = \sum_{k=1}^{q} (1-f_k) A_k > 0$$

and if $p = q$, we have

$$\sum_{i=1}^{m} a_i t_{i\epsilon} - \sum_{i=1}^{m} a_i t_i = -\sum_{k=1}^{q} f_k A_k > 0.$$

So, finally we have $\sum_{i=1}^{m} a_i t_{i\epsilon} \geq \sum_{i=1}^{m} a_i t_i$ for all cases. The lemma is completely proved.

Lemma 4. *Given $\rho : (s_0, t_0)(s_1, t_1) \dots (s_m, t_m) \dots$ is a behavior of timed automaton $\mathcal{A}$ and $\sigma : (s_{u-1}, t_{u-1})\ b\ (s_u, t_u)(s_{u+1}, t_{u+1}) \dots (s_v, t_v)\ e\ (s_{v+1}, t_{v+1})$ is an observation of ρ in the time interval $[b, e]$. Then for all $\epsilon \in [0, 1)$*

1. *$\rho_\epsilon : (s_0, t_{0\epsilon})(s_1, t_{1\epsilon}) \dots (s_m, t_{m\epsilon}) \dots$ is integral behavior of $\mathcal{A}$.*
2. *$\sigma_\epsilon : (s_{u-1}, t_{(u-1)\epsilon})\ b_\epsilon\ (s_u, t_{u\epsilon})(s_{u+1}, t_{(u+1)\epsilon}) \dots (s_v, t_{v\epsilon})\ e_\epsilon\ (s_{v+1}, t_{(v+1)\epsilon})$ is also integral observation of ρ_ϵ, i.e list and order of states appearing on time interval $[b_\epsilon, e_\epsilon]$ of integral behavior ρ_ϵ are the same as on interval $[b, e]$ of behavior ρ.*

Proof.

1. To prove ρ_ϵ be also a behavior we need proving following items
   - *Monotonicity:* Consider for all $j \geq i$. As ρ is a behavior, so $t_j \geq t_i$. Applying the lemma 1 we also have $t_{j\epsilon} - t_{i\epsilon} \geq 0$, i.e. $t_{j\epsilon} \geq t_{i\epsilon}, \forall j > i$.
   - *Time progress:* Let any integer number T. As ρ is a behavior so $\exists t_i : t_i \geq T$, this implies $t_{i\epsilon} \geq T$, due to T is integer. Hence, ρ_ϵ also satisfies time progress property.
   - *Transition preserve:* For all $i > 0$, we need proving that $t_{i\epsilon}$ is also time point at which the automaton transits state to s_i. In fact, due to ρ is behavior so at time point t_i the automaton transits to s_i by some transition $< s_{i-1}, \phi, a, \lambda, s_i >$. Assume that ϕ consists of time constraints of form $a \leq x \leq b$ and t_j is last time point clock x is reset before the automaton transits to state s_i. Then, value of x at time point t_i is $t_i - t_j$. That is $a \leq t_i - t_j \leq b$, by the lemma 1 we also have $a \leq t_{i\epsilon} - t_{j\epsilon} \leq b$. Hence, by induction it can see that $t_{j\epsilon}$ is also last time point clock x is reset before time point $t_{i\epsilon}$ along ρ_ϵ and value of x at $t_{i\epsilon}$ is $t_{i\epsilon} - t_{j\epsilon}$ that satisfies time constraint ϕ. By similar proving, if ϕ is of form $a \leq x - y \leq b$ then this inequality is also satisfied at integral time point $t_{i\epsilon}$. Thus, $t_{i\epsilon}$ are also time point at which the automaton transits from s_{i-1} to s_i by the transition $< s_{i-1}, \phi, a, \lambda, s_i >$. In short, ρ_ϵ is also a (integral) behavior of the automaton.
2. We are considered that by Lemma 1 ϵ-discretising does not change list of states occurring on behavior ρ in general (on interval $[b, e]$ in particular) and the order of time points of these states (included b, e). Hence, this item of the lemma is proved. Figure 2 expresses a case of discretising σ on $[b, e]$ to σ_ϵ on $[b_\epsilon, e_\epsilon]$.

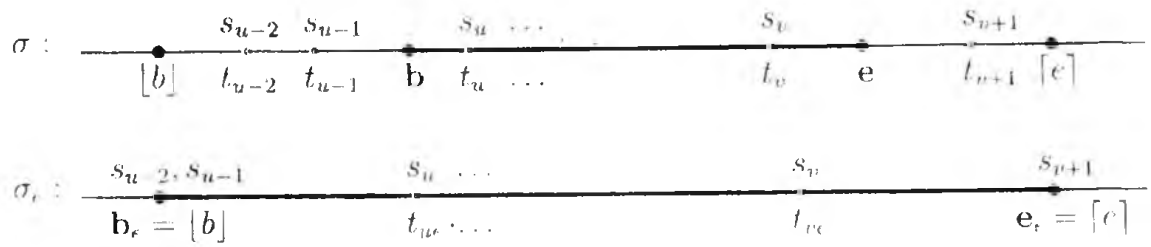

Fig. 2. A case of an observation with $b_\epsilon = \lfloor b \rfloor$ and $e_\epsilon = \lceil e \rceil$

3.2. Discretising LDC

Given a timed automaton $\mathcal{A}$ and a LDC formula φ. Let σ be an observation on time interval $[b, e]$ of $\mathcal{A}$. Let θ denote $\sum_{i=1}^{m} c_i \int s_i$ of φ, where $\int s_i$ is the duration of state s_i. Then $\theta(\sigma)$ is value of θ being valuated on the observation σ. Concretely, with the observation $\sigma : (s_{u-1}, t_{u-1})\ b\ (s_u, t_u)(s_{u+1}, t_{u+1}) \dots (s_v, t_v)\ e\ (s_{v+1}, t_{v+1})$ we have (see fig. 1):

$$\theta(\sigma) = c_{s_{u-1}}(t_u - b) + \sum_{i=1}^{m} c_i \left(\sum_{j=u, s_j = s_i}^{v-1} (t_{j+1} - t_j) \right) + c_{s_v}(e - t_v)$$

where $c_{s_{u-1}}$ and c_{s_v} is coefficients of states s_{u-1} and s_v in φ, corresponding. By expanding sum and let t_i's be common factors, we have.

$$\theta(\sigma) = \sum_{i=u}^{v} a_i t_i + c_{s_v} e - c_{s_{u-1}} b$$

where a_i's are real numbers that depending on c_i's.

Definition 7.[Satisfiable] Given an timed automaton $\mathcal{A}$ and a formula LDC φ

- an observation $\sigma : (s_{u-1}, t_{u-1})\ b\ (s_u, t_u)(s_{u+1}, t_{u+1}) \dots (s_v, t_v)\ e\ (s_{v+1}, t_{v+1})$ on time interval [b, e] is called satisfy φ (be denoted by $\sigma \models \varphi$) iff $\theta(\sigma) \leq M$.
- an behavior $\rho = (s_0, t_0)(s_1, t_1)(s_2, t_2) \dots (s_m, t_m) \dots$ is called satisfy φ (be denoted by $\rho \models \varphi$) iff $\sigma \models \varphi$ for all observations σ on ρ.
- an timed automaton $\mathcal{A}$ is called satisfy φ iff all behaviors of $\mathcal{A}$ satisfy φ, i.e $\rho \models \varphi$ for all behaviors ρ.

In the case φ is not satisfied by σ, ρ or timed automaton $\mathcal{A}$, we denote $\sigma \not\models \varphi$, $\rho \not\models \varphi$ or $\mathcal{A} \not\models \varphi$.

Now we prove that LDC is a discretisable property. That means a timed automaton $\mathcal{A}$ satisfies a formula LDC φ iff all integral behaviors ρ of $\mathcal{A}$ satisfy φ

Theorem 1. *Any linear duration constraint φ is discretisable with respect to timed automaton $\mathcal{A}$.*

Proof : Declaration of $\mathcal{A} \models \varphi \Rightarrow \rho \models \varphi$ for all integral behaviors ρ is obvious. For inverse we will prove that if there exists a behavior ρ of $\mathcal{A}$ such that $\rho \not\models \varphi$, then we also can find ϵ such that integral behavior $\rho_\epsilon \not\models \varphi$.

In fact, assume that behavior ρ does not satisfy φ. That means there exists $\sigma : (s_{u-1}, t_{u-1})\ b\ (s_u, t_u)(s_{u+1}, t_{u+1}) \dots (s_v, t_v)\ e\ (s_{v+1}, t_{v+1}) \not\models \varphi$, i.e. $\theta(\sigma) > M$. By definition of LDC, we have

$$\theta(\sigma) = \sum_{i=u}^{v} a_i t_i + c_{s_v} e - c_{s_{u-1}} b > M$$

From Lemma 3, $\exists \epsilon \in [0, 1)$ such that $\sum_{i=u}^{v} a_i t_{i\epsilon} + c_{s_v} e_\epsilon - c_{s_{u-1}} b_\epsilon \geq \theta(\sigma) > M$.

In the other hand, from the Lemma 4 with this ϵ we receive integral behavior ρ_ϵ and sequence of time states on interval $[b_\epsilon, e_\epsilon]$ is also an observation (integral). Hence, it is

easily to see that $\theta(\sigma_\epsilon) = \sum_{i=u}^{v} a_i t_{i\epsilon} + c_{s_v} e_\epsilon - c_{s_{u-1}} b_\epsilon$. So $\theta(\sigma_\epsilon) > M$ and we receive ρ_ϵ on which there is observation σ_ϵ unsatisfying φ. That is, we find an integral behavior ρ_ϵ and $\rho_\epsilon \not\models \varphi$.

In summary, LDC is discretisable w.r.t the timed automata.

4. Some Discretisable Classes of Duration Properties

On based of discretisability of LDC, in this section we discuss about discretisability of some classes of formulas in DC.

4.1 History Properties

History properties are properties which checking them concerns list and temporal order of states in observations. Often, that are properties requiring behavior of system must go or not through a certain sequence of states. In general, formulas considered in this section are of form $\varphi \mathrel{\widehat{=}} \mathsf{Sequel} \Rightarrow$ LDC with Sequel is sequence of states of system. Given an observation σ on the time interval $[b, e]$, $\sigma \models \varphi$ iff sequence of states on $[b, e]$ is either matches to Sequel and $\theta(\sigma) \leq M$ or does not match.

Theorem 2. *Any history property φ is discretisable with respect to timed automata.*

Proof. Discretisability of these formulas can be proved easily from lemma 4 that it is reminded ϵ-discretising does not change list and occurring order of states in any observation.

For interpretation, we give two such classes of formulas was shown be discretisable in [15,16].

Inter-State Duration Properties [15]

$$\varphi_1 \mathrel{\widehat{=}} \Box([\lceil u \rceil]^0 \frown [\lceil \neg u \rceil] \frown [\lceil u \rceil]^0 \Rightarrow \sum_{s \in S} c_s \int s \leq M),$$

where S is the set of states of $\mathcal{A}$, $u, s \in S$, and all c_s and M are reals.

In formula φ_1, $[\lceil u \rceil]^0$ is a DC formula which is true at an interval $[t_1, t_2]$ iff $t_1 = t_2$ and at point time t_1 system stays at state u. $[\lceil \neg u \rceil]$ is true at an interval $[t_1, t_2]$ iff system does not stay at any time point between t_1 and t_2. Thus, a timed automaton satisfies φ_1 iff for all observation σ on $[b, e]$ such that if timed automaton at time points b and e stays at state u and from b to e, system does not stay at u then $\theta(\sigma) \leq M$.

Temporal Duration Properties - TDP [16]

$$\varphi_2 \mathrel{\widehat{=}} \Box([\lceil s_{i_1} \rceil] \frown [\lceil s_{i_2} \rceil] \frown \ldots \frown [\lceil s_{i_k} \rceil] \Rightarrow \sum_{s \in S} c_s \int s \leq M),$$

where S is the set of states of $\mathcal{A}$, s_{i_j}'s are states and all $c_s (s \in S), M$ are reals.

Semantics of formula φ_2 is if observation σ goes through sequence of states in order $s_{i_1}, s_{i_2}, \ldots, s_{i_k}$ (such that at time point b and e, system stays at states u_1, u_k, respectively) then $\theta(\sigma) \leq M$.

The case studies are used to illustrate for above kinds of formulas reader is refer to [15, 16].

4.2. Combination of LDCs

A class of general duration formulas that is considered by many authors (e.g. [12]) are Disjunctions or Conjunctions of LDCs. In [12] authors only considered these formulas with integral coefficients. Here, we discuss about discretisability of them in general case that means coefficients of formulas are reals.

Conjunction of LDCs

$$\psi_1 = \bigwedge_{k=1}^{n} \left(\sum_{i=1}^{m} c_{ki} \int s_{ki} \leq M_k \right).$$

From proof of discretisability of LDC we can easily see that a conjunction of LDC's is also discretisable. Assume that there exits an observation σ that does not satisfy ψ_1, i.e there exits k such that $\sigma \not\models (\sum_{i=1}^{m} c_{ki} \int s_{ki} \leq M_k)$, hence $\theta(\sigma) > M_k$. By Theorem 1. there exits $\epsilon \in [0, 1)$ such that $\theta(\sigma_\epsilon) > M_k$, too. So $\sigma_\epsilon \not\models \psi_1$, in the other word ψ_1 is discretisable formula.

Disjunction of LDCs

$$\psi_2 = \bigvee_{k=1}^{n} \left(\sum_{i=1}^{m} c_{ki} \int s_{ki} \leq M_k \right).$$

Up to now we have still not known whether this formula is discretisable (even for case of integral coefficients). However, a subclass of ψ_2 which is called Linear Duration Invariant is discretisable. That is formula that is researched in many works [11, 13, 14]. Discretisability of this formula is proved below.

4.3. Linear Duration Invariant - LDI

Definition 8. Given a timed automaton $\mathcal{A}$ with the set of states S. A linear duration invariant over S is a formula in Duration Calculus of the form :

$$\psi \mathrel{\widehat{=}} B \leq \ell \leq E \Rightarrow \sum_{i=1}^{m} c_i \int s_i \leq M,$$

where B, E are integer numbers, and coefficients c_i, M are real numbers. $B \leq E$ (E may be ∞), $s_i \in S$.

Semantics of LDI can be informally understood as follows : In any observation interval of system, if the length ℓ of interval satisfies the premise of ψ (i.e $B \leq \ell \leq E$) then durations d_{s_i} of states s_i of system must satisfy the conclusion of ψ, (i.e $\sum_{i=1}^{m} c_i d_{s_i} \leq M$).

Theorem 3. *Any linear duration invariant ψ is discretisable with respect to timed automaton $\mathcal{A}$.*

Proof. Similar to proof in Theorem 1. we assume that there exists an observation σ on time interval $[b, e]$ such that $\sigma \not\models D$, that means $B \leq e - b \leq E$ and $\theta(\sigma) > M$. By

Theorem 1 we can find an integer observation σ_ϵ such that $\theta(\sigma_\epsilon) > M$. Therefore, we only need prove an extra thing, that is the length of integral observation σ_ϵ on interval $[b_\epsilon, e_\epsilon]$ also must be belong in $[B, E]$, this is easily implied from Lemma 1 and hypothesis $B \leq e - b \leq E$. Thus, from assumption of $\sigma \not\models D$ we also find an integer observation σ_ϵ such that $\sigma_\epsilon \not\models D$, too. And we can see that formula LDI ψ is discretisability.

5.Conclusion

In this paper we made some comments to discretisability of some classes of formulas in duration calculus. Due to as we known verifying such formulas is very hard, so discretisability of them is meaningful. According to [12] formulas of form combination of LDC (with integral coefficients) is checking by mixed integer linear programming. Time complexity of this algorithm is very high by complexity of mixed integer linear programming problem. However, idea of discretising in [5] that was applied in [12] was emotion for later algorithms of checking LDI, LDC, TDP [13, 14, 16]. Especially, in [15,16] authors was given algorithms for checking LDC and TDP with complexity is the same as complexity of reachability problem on based of searching region graphs of timed automata. These algorithms can be improved by using zone graph instead of region graph because size of zone graph [9] is smaller than size of region graph.

Main result of this paper is proof about discretisability of Linear Duration Invariant which is considered in recent years. Especially, discretisability of LDI is an important feature for constructing a checking algorithm which based on traverse zone graph. A zone graph is an abstraction of state space of timed automata [8]. Paths of graph is corresponding to behaviors of timed automata, so we can check true of LDI on every paths of graph. To do this, each vertex of graph is assigned to c_s, where c_s is coefficient of state s in formula LDI and s is state which belongs to vertex is considered. Similarly, we assign a value of length to each edge of graph. This value expressed maximum time length which automata can be taken transition from this vertex to another vertex of edge. Hence, with each fragment on a path of graph which represents an observation σ we can easily calculate ℓ and $\theta(\sigma)$ and hence check conditions in LDI. However, as starting and ending points of observations are arbitrary (in real time model) so number of observations on each path is infinitive. By discretisability of LDI we can choose starting and ending points of observation on paths are integral points, so the number of an observations becomes finite. That is some ideas about checking algorithm based on zone graph. Within the scope of this paper we do not discuss about details of algorithm. We hope that an detail algorithm will be advance and implement in the future.

Acknowledgement. The author would like to thank Dr. Dang Van Hung for his valuable comments and encouragement when writing this paper.

References

1. Rajeev Alur and Thomas A. Henzinger, Logics and models of real time: A survey, *Real Time: Theory in Practice*, LNCS 600, Springer-Verlag, 1992, pp. 74–106.
2. R. Alur, C. Courcoubetis, T.A. Henzinger, Computing accumulated delays in real-time systems, *Proceedings of the Fifth Conference on Computer-Aided Verification*, LNCS 697, 1993, pp. 181-193.
3. E. Harel, O. Lichtenstein and A. Pnueli, Explicit-clock temporal logic, *Proceedings of the Fifth Annual Symposium on Logic in Computer Science*, IEEE Computer Society Press, 1990, pp. 402-413.
4. T. Henzinger, Z. Manna and A. Pnueli, Temporal proof methodologies for real-time systems. *Proceedings of the 18th Annual Symposium on Principles of Programming Languages*, ACM Press, 1991, pp. 353–366.
5. T. Henzinger, Z. Manna, and A. Pnueli, What good are digital clock? *Lecture Notes in Computer Science*, Springer-Verlag, Vol 623(1992), pp. 545-558.
6. R. Alur and D.L. Dill, A Theory of Timed Automata, *Theoretical Computer Science*, 1994, pp. 183-235.
7. A. Puri, A. Gollu and P. Varaiya, Discretization of timed automata. *Proceedings of the 33rd IEEE conference on decision and control*, 1994, pp. 957-958.
8. S. Yovine, Model-checking timed automata, *Lectures on Embedded Systems*, G. Rozenberg and F. Vaandrager (Eds.). LNCS 1494, Springer-Verlag, 1998.
9. S. Tripakis, S. Yovine, Analysis of timed systems based on time-abstracting bisimulations *Formal Methods in System Design*, Kluwer Academic Publishers, Boston, 18(2001), 25-68.
10. Zhou Chaochen, C.A.R. Hoare, Anders P. Ravn, A calculus of durations, *Information Processing Letters*, 40(5), 1994, pp 269-276.
11. Zhou Chaochen, Zhang Jingzhong, Yang Lu, and Li Xiaoshan, Linear Duration Invariants, *Formal Techniques in Real-Time and Fault-Tolerant systems*, LNCS 863. Springer Verlag, 1994.
12. Y. Kesten, A. Pnueli, J Sifakis, and S. Yovine, Integration Graphs: A Class of Decidable Hybrid Systems, *Hybrid Systems*, LNCS 736, Springer Verlag, 1994. pp. 179-208.
13. Li Xuan Dong and Dang Van Hung, Checking Linear Duration Invariants by Linear Programming, *Proceedings of Concurrency and Parallelism, Programming, Networking, and Security*, Joxan Jaffar and Roland H. C. Yap (Eds.), LNCS 1179, Springer-Verlag, Dec 1996, pp. 321-332.
14. Pham Hong Thai and Dang Van Hung, Checking a Regular Class of Duration Calculus Models for Linear Duration Invariants, *Proceedings of the International Symposium on Software Engineering for Parallel and Distributed Systems*, Bernd Kramer, Naoshi Uchihira, Peter Croll and Stefano Russo (Eds). IEEE Press 1998, pp. 61-71.

15. Zhao Jianhua and Dang Van Hung, Checking Timed Automata for Some Discretisable Duration Properties, *Journal of Computer Science and Technology*, Volume 15, Number 5, September 2000. pp. 423–429.
16. Li Yong and Dang Van Hung,Checking Temporal Duration Properties of Timed Automata, *Journal of Computer Science and Technology*, Vol. 17, No. 6, Nov. 2002. pp. 689-698.
17. Pham Hong Thai, Checking Parallel Real-Time Systems for Temporal Duration Properties by Linear Programming, *Journal of Sciences, VNU*, Vol.19, No. 4, Nov. 2003. pp. 49-62.
18. Manoranjan Satpathy, Dang Van Hung, Paritosh K. Pandya, Some Results on The Decidability of Duration Calculus under Synchronous Interpretation, *Proceedings of the 5^{th} International Symposium on Formal Techniques in Real-Time and Fault-Tolerant Systems*, Lyngby, Denmark, September 1998, LNCS 1486, Springer-Verlag 1998, pp. 186-197.