Duration-constrained regular expressions

Li Xuandong¹, ², Zhao Jianhua¹, ², Zheng Tao¹, ², Li Yong¹, ² and Zheng Guoliang¹, ²

¹State Key Laboratory of Novel Software Technology, Nanjing University, Nanjing, P.R. China
²Department of Computer Science and Technology, Nanjing University, Nanjing, P.R. China

Abstract. This paper investigates the logic-automata-connection for Duration Calculus. It has been frequently observed that Duration Calculus with linear duration terms comes close to being a logic of linear hybrid automata. We attempt to make this relation precise by constructing Kleene-connection between duration-constrained regular expressions and a subclass of linear hybrid automata called loop-reset automata in which any variable tested in a loop is reset in the same loop. The formalism of duration-constrained regular expressions is an extension of regular expressions with duration constraints, which are essentially formulas of Duration Calculus without negation, yet extended by a Kleene-star operator. In this paper, we show that this formalism is equivalent in expressive power to loop-reset automata by providing a translation procedure from expressions to automata and vice versa.

Keywords: Duration Calculus; Hybrid automata; Regular expressions

1. Introduction

Duration Calculus [CHR91] is a logic to specify and reason about requirements for real-time and hybrid systems. It is an extension of Interval Temporal Logic [Mos85] which can be used to reason about integrated constraints over time-dependent and Boolean value states without explicit mention of absolute time. In Duration Calculus, states are modeled as Boolean functions from reals (representing continuous time) to {0, 1}, where 1 denotes state presence, and 0 denotes state absence. Let $S$ be a finite set of states. For a state $s \in S$, integral variable $\int s$ of DC is a function from bounded and closed intervals to reals which stands for the accumulated presence time (duration) of state $s$ over an interval. For bounded interval $[a, b]$ ($b \geq a$), $\int s[a, b] = \int_k^b s(t)dt$. It follows that $\int [a, b] = \int_a^b 1 dt = |b - a|$, i.e. the length of $[a, b]$. A duration constraint is of the form

$$a \leq \sum_{i=1}^m c_i \int s_i \leq b$$

where the $s_i$'s are states and $a, b, c_i$ are real numbers. In this paper, for exploring the logic-automata-connection for Duration Calculus we introduce duration-constrained regular expressions, which are an extension of regular expressions with duration constraints and can be used for describing the behavior of hybrid systems.

Hybrid systems are real-time systems that allow continuous state changes, over time periods of positive duration, as well as discrete state changes, in zero time. The formalism of hybrid automata [Hen96, ACH95, HKP98, HEM00] has become a standard model for real-time and hybrid systems. The language that a hybrid automaton accepts describes the behavior of a hybrid system, which is a set of timed traces. A timed trace is of the form

Correspondence and offprint requests to: Li Xuandong, Department of Computer Science and Technology, Nanjing University, Nanjing, Jiangsu, P.R. China 210093. Email: lxd@nju.edu.cn
(s₁, t₁)∗(s₂, t₂)∗...∗(sₙ, tₙ), which represents a behavior of a system where the system starts at the state s₁, stays there for t₁ time units, then changes to s₂ and stays in s₂ for t₂ time units, and so on. The values t₁, t₂, ..., tₙ have to satisfy some time constraints enforced by the system. Since the number of timed traces to express the behavior of a hybrid system may be infinite, we introduce duration-constrained regular expressions as a finite representation of behavior of systems.

This paper investigates the logic-automata-connection for Duration Calculus. It has been frequently observed that Duration Calculus with linear duration terms comes close to being a logic of linear hybrid automata (piecewise constant-slope hybrid automata) [KPS94, ACH93, CJL94, XHT97, XTJ98, XUH96]. We set out to make this relation precise by constructing Kleene-connection between duration-constrained regular expressions and a subclass of linear hybrid automata called loop-reset automata in which any variable tested in a loop is reset in the same loop. Duration-constrained regular expressions are essentially formulas of Duration Calculus without negation, yet extended by a Kleene-star operator. In this paper, we show that this formalism is equivalent in expressive power to loop-reset automata by providing a translation procedure from expressions to automata and vice versa.

The paper is organized as follows. In the next section, we introduce the notion of duration-constrained regular expressions. In section 3 we review the basic definitions of linear hybrid automata, and introduce duration-constrained regular expressions. In section 4 and 5 show that duration-constrained regular expressions and loop-reset automata are equivalent in expressive power. The last section is the conclusion of the paper.

2. Duration-constrained regular expressions

Let S be a finite set of states, R⁺ be the set of nonnegative real numbers. A timed trace over S is a list of elements in S × R⁺, which can be written as

\[ \sigma = (s₁, t₁)^∗(s₂, t₂)^∗...^∗(sₙ, tₙ) \]

where sᵢ ∈ S, tᵢ ∈ R⁺, and sᵢ ≠ sᵢ₊₁. We call \( \sum_{i=1}^{n} t_i \) the length of \( \sigma \). In this paper, we use \( ^∗ \) to denote the concatenation of timed traces. For timed traces \( \sigma₁ = (s₁₁, t₁₁)^∗(s₁₂, t₁₂)^∗...^∗(s₁m, t₁m) \) and \( \sigma₂ = (s₂₁, t₂₁)^∗(s₂₂, t₂₂)^∗...^∗(s₂n, t₂n) \), if \( s₁m ≠ s₂₁ \), then

\[ \sigma₁^∗\sigma₂ = (s₁₁, t₁₁)^∗(s₁₂, t₁₂)^∗...^∗(s₁m, t₁m)^∗(s₂₁, t₂₁)^∗(s₂₂, t₂₂)^∗...^∗(s₂n, t₂n) \]

otherwise

\[ \sigma₁^∗\sigma₂ = (s₁₁, t₁₁)^∗(s₁₂, t₁₂)^∗...^∗(s₁m, t₁m + t₂₁)^∗(s₂₂, t₂₂)^∗...^∗(s₂n, t₂n) \]

Let \( \varepsilon \) be an empty timed trace such that for any timed trace \( \sigma \), \( \sigma^∗\varepsilon = \varepsilon^∗\sigma = \sigma \).

For a duration constraint \( d = a ≤ \sum_{i=1}^{m} c_i \int sᵢ \leq b \), for a timed trace \( \sigma = (s₁', t₁')^∗(s₂', t₂')^∗...^∗(sₙ', tₙ) \), the integrated duration of state sᵢ (1 ≤ i ≤ m) over \( \sigma \) can be calculated as

\[ \int sᵢ = \sum_{u \in S} c_u \int u \leq n ∧ sᵢ' \Rightarrow sᵢ \]

Consequently, \( \sigma \) satisfies \( \delta \) if and only if \( a ≤ \sum_{i=1}^{m} c_i (\sum_{u \in S} t_u) \leq b \).

While a regular expression over a set of states/transitions (alphabet) is a finite representation of a (infinite) set of sequences of states/transitions, a duration-constrained regular expression (DRE) will be a finite representation of a set of timed traces. By incorporating duration constraints into regular expressions, we get duration-constrained regular expressions.

Definition 1 For an DRE \( R \), its language over a finite set S of states is denoted by \( L(R) \). DREs are defined recursively as follows.

1. \( \varepsilon \) is a DRE, and \( L(\varepsilon) = \{ \varepsilon \} \).
2. If \( s \in S \), then \( s \) is a DRE, and \( L(s) = \{(s, t) \mid t \in R⁺ \} \).
3. If \( R₁ \) and \( R₂ \) are DREs, then \( R₁ \cup R₂ \) is a DRE, and

\[ L(R₁ \cup R₂) = \{ \sigma₁^∗\sigma₂ \mid \sigma₁ \in L(R₁), \sigma₂ \in L(R₂) \} \]

4. If \( R₁ \) and \( R₂ \) are DREs, then \( R₁ \oplus R₂ \) is a DRE, and

\[ L(R₁ \oplus R₂) = L(R₁) \cup L(R₂) \]