Refinements and Abstractions of Signal-Event (Timed) Languages

Béatrice Bérard[1], Paul Gastin[2], and Antoine Petit[2]

[1] LAMSADE, Université Paris Dauphine & CNRS,
Place du Maréchal de Lattre de Tassigny, F-75775 Paris Cedex 16, France
[2] LSV, ENS de Cachan & CNRS,
61 av. du Président Wilson, F-94235 Cachan Cedex, France

Abstract. In the classical framework of formal languages, a refinement operation is modeled by a substitution and an abstraction by an inverse substitution. These mechanisms have been widely studied, because they describe a change in the specification level, from an abstract view to a more concrete one, or conversely. For timed systems, there is up to now no uniform notion of substitutions. In this paper, we study the timed substitutions in the general framework of signal-event languages, where both signals and events are taken into account. We prove that regular signal-event languages are closed under substitutions and inverse substitutions.

1 Introduction

Refinements and abstractions. Operations of refinements and abstractions are essential tools for the design and the study of systems, real-time or not. They allow to consider a given system at different levels of abstractions. For instance, some procedure or function can simply be viewed at some abstract level as a single action, and can be later expanded into all its possible behaviours at some more concrete level. Or conversely, a set of behaviours are merged together and replaced by a single action, in order to obtain a more abstract description.

These operations can be formally modeled by substitution and inverse substitution respectively. Therefore, substitutions have been extensively studied in the untimed framework, with the underlying idea that interesting classes of languages have to be closed under these operations.

Timed languages. In order to accept timed words, the model of timed automata was first proposed in [1,2]. It has been widely studied for the last fifteen years and successfully applied to industrial cases. For this model, an observation, called a time-event word, may be viewed as an alternating sequence of waiting times and instantaneous actions. A timed automaton is a finite automaton extended with variables called clocks, designed to recognize time-event words: time elapses while the control stays in a given node and an event is observed when a discrete transition occurs.

Another model was introduced by [3], and further studied in [10,4,11] with the aim of describing hardware systems. In this case, an observation is a signal word, i.e., a sequence of factors a^d, where a is a signal and d is its duration. The original model of

E. Asarin and P. Bouyer (Eds.): FORMATS 2006, LNCS 4202, pp. 67–81, 2006.

timed automata was then modified to fit this setting: a signal is emitted while the automaton stays in some state and no event is produced when a discrete transition is fired. In this framework, when a transition occurs between two states with the same signal a, we obtain a^{d_1} followed by a^{d_2}, which are merged into $a^{d_1+d_2}$. This phenomenon is called stuttering.

It was noticed in [4] that both approaches are complementary and can be combined in an algebraic formalism to obtain the so-called signal-event monoid. Timed automata can be easily adapted to take both signals and events into account, thus yielding signal-event automata: states emit signals and transitions produce events.

We consider in this paper both finite and infinite behaviors of signal-event automata and we also include unobservable events (ε-transitions) and hidden signals (τ-labeled states). It turns our that these features are very useful, for instance for handling abstractions. They also allow us to get as special cases the initial models of timed automata and signal automata.

Timed substitutions. Timed substitutions were studied in [8] for regular transfinite time-event languages. In [8], although no signal appear explicitely, actions are handled in a way similar (but not identical) to signals, *without* stuttering. Here we restrict the study of substitutions to finite and ω-sequences but we do handle signal stuttering which is a major difficulty.

Our contribution. The aim of this paper is to study the closure by substitutions and inverse substitutions of the families SEL_ε and SEL of languages accepted by signal-event automata, with or without ε-transitions. We prove that the class SEL_ε is closed under arbitrary substitutions and under arbitrary inverse substitutions. These closure properties are not verified by the class SEL in general. Nevertheless, we show that SEL is closed under inverse substitutions acting on events only, i.e., leaving signals unchanged, and we give a sufficient condition for its closure under substitutions. These results again show the robustness of the class SEL_ε, which is in favour of signal-event automata including ε-transitions.

Outline of the paper. We first give in Section 2 precise definitions of finite and infinite signal-event languages, with the corresponding notion of signal-event automata and we recall some technical results on signal-event automata that will be crucial for further proofs. In Section 3, we define timed substitutions which are duration preserving mappings. We then study in Section 4 the closures of the classes SEL and SEL_ε under reconizable substitutions and their inverses.

For lack of space, this paper does not contain the proofs of the correctness of the different automata constructions we proposed. These proofs are available in the technical report [6].

2 Signal-Event Words and Signal-Event Automata

Let Z be any set. We write Z^* (respectively Z^ω) the set of finite (respectively infinite) sequences of elements in Z, with ε for the empty sequence, and $Z^\infty = Z^* \cup Z^\omega$ the set of all sequences of elements in Z. The set Z^∞ is equipped with the usual partial concatenation defined from $Z^* \times Z^\infty$ to Z^∞.