Automata and logics over finitely varying functions[☆]

Fabrice Chevalier [a], Deepak D'Souza [b,*], M. Raj Mohan [b], Pavithra Prabhakar [c]

[a] Laboratory for Specification and Verification, ENS de Cachan, France
[b] Department of Computer Science and Automation, Indian Institute of Science, Bangalore, India
[c] Department of Computer Science, University of Illinois at Urbana-Champaign, USA

ARTICLE INFO

ABSTRACT

We extend some of the classical connections between automata and logic due to Büchi (1960) [5] and McNaughton and Papert (1971) [12] to languages of finitely varying functions or "signals". In particular, we introduce a natural class of automata for generating finitely varying functions called ST-NFA's, and show that it coincides in terms of language definability with a natural monadic second-order logic interpreted over finitely varying functions Rabinovich (2002) [15]. We also identify a "counter-free" subclass of ST-NFA's which characterise the first-order definable languages of finitely varying functions. Our proofs mainly factor through the classical results for word languages. These results have applications in automata characterisations for continuously interpreted real-time logics like Metric Temporal Logic (MTL) Chevalier et al. (2006, 2007) [6,7].

1. Introduction

The classical literature contains a rich theory connecting automata and logic over words. Büchi showed that languages definable in monadic second logic (MSO) over words are precisely the class of languages accepted by finite state automata [5]. Kamp [10] showed that languages definable in Linear-Time Temporal Logic (LTL) were precisely the languages definable in the first-order (FO) fragment of Büchi's MSO. And McNaughton and Papert [12] showed that the class of counter-free finite state automata (where a "counter" in an automaton is a loop with at least two hops, each hop being on a common word u) define exactly the FO-definable languages. This last result uses a characterisation due to Schutzenberger [17] of the class of counter-free languages in terms of star-free regular expressions.

Our aim in this article is to lift these connections to languages of finitely varying functions from the non-negative reals to a finite alphabet. These functions are finitely varying in that they have only a finite number of discontinuities in any bounded interval of time. Such functions are often called "signals" in the literature, are of interest to the computer science community as they model the behaviour of timed and hybrid systems [2,1,4]. For example, non-zeno timed words [2] are special kinds of signals, as are the piece-wise constant behaviours of [4].

We first introduce a class of automata called ST-NFA's that run over signals and hence accept languages of signals. We should point out here that unlike timed automata, we are interested in formalisms without a "metric" or operators that measure time distance. As a consequence, these languages are essentially "untimed" in that they can be characterised as the set of all possible "timings" of a (regular) language of classical words. We then consider a natural monadic second-order logic introduced earlier by Rabinovich [15] and called here MSO[s], which is interpreted over signals, and in which the second-order quantification is restricted to subsets of non-negative reals whose characteristic functions are finitely varying. We show that
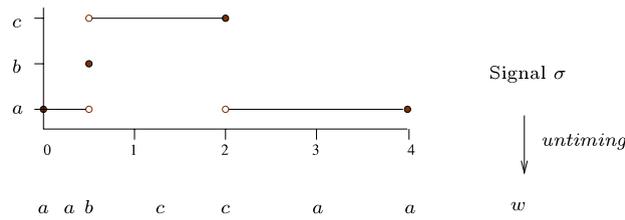

Fig. 1. A signal and its untiming.

the class of signal languages defined by sentences in this logic is precisely the class of signal languages defined by ST-NFA's. This gives an automata-theoretic proof of a similar result obtained in [15] using logical techniques. We note that this proof also gives us a simple automata-theoretic proof of the fact that the monadic second-order logic of the non-negative reals (where second-order quantification ranges over finitely varying subsets) is decidable.

Next, along the lines of the Schutzenberger and McNaughton–Papert results, we identify a counter-free subclass of ST-NFA's and show that they precisely characterise the class of signal languages definable by the first-order fragment FO^s of MSO^s. The notion of a counter in an ST-NFA is similar to the classical one except that we require the ST-NFA to be "proper" in a certain sense. Our proof of this result factors transparently through the aforementioned results of Schutzenberger, McNaughton–Papert, and Kamp for word languages. The main difficulty, in a series of steps we perform, is to translate an LTL formula θ interpreted over word models, into one interpreted over signals which accepts precisely the timings of the word models of θ. As in the classical case for words, we show that this characterisation allows us to decide whether a given MSO^s sentence is FO^s-definable.

As a minor by-product, we re-prove the expressive completeness of LTL interpreted over signals (i.e. its expressiveness coincides with FO^s over signals). This result also follows from Kamp's result showing the expressive completeness of LTL over arbitrary functions over the reals [10]. Nonetheless, our proof gives a more accessible proof of this result since it uses only Kamp's result for classical words, for which there are simpler proofs in the literature (see [20,9]).

Turning now to more details on related work, as already mentioned, this paper builds on the classical results due to Büchi [5], Schutzenberger [17], McNaughton and Papert [12], and Kamp [10] for word languages. The work of Rabinovich contains many relevant results on signal languages. Rabinovich and Trakhtenbrot [16] introduce automata similar to ST-NFA's called signal acceptors. In [15], Rabinovich shows how to translate an MSO sentence φ to a MSO^s sentence that accepts precisely the timings of φ, and vice versa. This leads to a proof of the claim in [16] that signal languages definable by signal acceptors and MSO^s coincide. In contrast, our equivalence of ST-NFA's and MSO^s uses an automata-theoretic argument similar to the proof of Büchi's result (see [19]), and helps us identify the counter-free fragment. In [14], Rabinovich also shows a star-free regular expression characterisation of FO^s-definable signal languages along the lines of McNaughton and Papert [12].

Though we are mainly concerned with expressiveness in this work, there are a number of related decidability results in the literature. Rabin [13] shows that MSO over reals, with second-order quantification over subsets which are essentially countable unions of closed sets, is decidable. The decidability of MSO^s follows from this result. Shelah [18] showed that MSO over reals with second-order quantification over arbitrary subsets of reals is undecidable.

Preliminary versions of the basic results in this paper appeared in [6,7], where they were used to obtain logical characterisations of versions of timed automata with a continuous interpretation, as well as a counter-free timed automata characterisation of several real-time temporal logics, including MTL [11,3], in their continuous semantics.

2. Preliminaries

For an alphabet A, we use A^* to denote the set of finite words over A. For a word w in A^*, we use $|w|$ to denote its length. The set of non-negative reals and rationals will be denoted by $\mathbb{R}_{\geq 0}$ and $\mathbb{Q}_{\geq 0}$, respectively. We will deal with intervals of non-negative reals, i.e. convex subsets of $\mathbb{R}_{\geq 0}$, and denote by $\mathcal{I}_{\mathbb{R}_{\geq 0}}$ the set of such intervals with end-points in $\mathbb{R}_{\geq 0} \cup \{\infty\}$. Two intervals I and J will be called *adjacent* if $I \cap J = \emptyset$ and $I \cup J$ is an interval.

Let A be a finite alphabet and let $f : [0, r] \to A$ be a function, where $r \in \mathbb{R}_{\geq 0}$. We use $dur(f)$ to denote the duration of f, which in this case is r. A point $t \in (0, r)$ is a point of *continuity* of f if there exists $\epsilon > 0$ such that f is constant in the interval $(t - \epsilon, t + \epsilon)$. All other points in $[0, r]$ are points of *discontinuity* of f. We say f is *finitely varying* if it has only a finite number of discontinuities in its domain. We will refer to such finitely varying functions as *signals* over A, and denote the set of signals over A by $Sig(A)$. Fig. 1 shows a signal σ over the alphabet $\{a, b, c\}$ defined on $[0, 4]$ as follows:

$$\sigma(t) = \begin{cases} a & \text{if } t \in [0, 0.5) \cup (2, 4] \\ b & \text{if } t = 0.5 \\ c & \text{if } t \in (0.5, 2]. \end{cases}$$

An *interval representation* for a signal $\sigma : [0, r] \to A$ is a sequence of the form $(a_0, I_0) \cdots (a_n, I_n)$, with $a_i \in A$ and $I_i \in \mathcal{I}_{\mathbb{R}_{\geq 0}}$ satisfying the conditions that: $0 \in I_0$, the union of the intervals is $[0, r]$, each I_i and I_{i+1} are adjacent, and for each i, σ is constant and equal to a_i in the interval I_i. We can obtain a *canonical* interval representation for σ by putting each point of discontinuity in a singular interval by itself. Thus the above interval representation for σ is
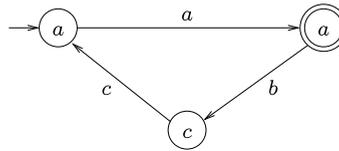
Fig. 2. Example ST-NFA.

canonical if n is even, for each even i the interval I_i is singular (i.e. of the form $[t, t]$), and for no even i such that $0 < i < n$ is $a_{i-1} = a_i = a_{i+1}$. The canonical interval representation for the signal of Fig. 1 is $([0, 0], a)((0, 0.5), a)$ $([0.5, 0.5], b)((0.5, 2), c)([2, 2], c)((2, 4), a)([4, 4], a)$.

A canonical interval representation for a function gives us a canonical way of "untiming" the signal: thus if $(a_0, I_0) \cdots (a_{2n}, I_{2n})$ is the canonical interval representation for a signal σ, then we define $untiming(\sigma)$ to be the string $a_0 \cdots a_{2n}$ in A^*. The untiming thus captures explicitly the value of the function at its points of discontinuity and the open intervals between them. The untiming of the signal in Fig. 1 is thus $aabccaa$. Note that strings which represent the untiming of a signal will always be of odd length, and for no even position i will the letters at positions $i - 1$, i, and $i + 1$ be the same. We refer to words in A^* which satisfy these two conditions as *proper* words over A. We denote the set of proper words over A by $Prop(A)$.

A canonical word w can be "timed" to get a signal in a natural way: thus a signal σ is in $timing(w)$ if $untiming(\sigma) = w$. We extend the definition of *timing* and *untiming* to languages of signals and words in the expected way.

Finally, we say a subset X of $\mathbb{R}_{\geq 0}$ is *finitely varying* if its characteristic function $f_X : \mathbb{R}_{\geq 0} \to \{0, 1\}$ given by $f_X(t) = 1$ if $t \in X$ and 0 otherwise, is finitely varying (in the sense defined above) in every interval of the form $[0, r]$ with $r \in \mathbb{R}_{\geq 0}$.

3. Automata over signals—ST-NFA's

In this section we introduce a variant of classical word automata called ST-NFA's which are a convenient formalism for generating signals.

We recall that a non-deterministic finite state automaton (NFA) over an alphabet A is a structure $\mathcal{A} = (Q, S, \delta, F)$ where Q is a finite set of states, S is the set of initial states, $\delta \subseteq Q \times A \times Q$ is the transition relation, and $F \subseteq Q$ is the set of final states. A run of $\mathcal{A}$ on a word $w = a_0 \cdots a_n \in A^*$ is a sequence of states $q_0, \ldots, q_{n+1}$ such that $q_0 \in S$, and $(q_i, a_i, q_{i+1}) \in \delta$ for each $i \leq n$. The run is accepting if $q_{n+1} \in F$. The word language accepted by $\mathcal{A}$, denoted $L(\mathcal{A})$, is the set of words in A^* over which $\mathcal{A}$ has an accepting run. Languages accepted by NFA's are called *regular* languages. We say the NFA $\mathcal{A}$ is *deterministic* (and call it a DFA) if the set of start states is a singleton and for each $p \in Q$ and $a \in A$, there is at most one outgoing transition of the form (p, a, q) in δ.

A *state-transition-labelled* NFA (ST-NFA for short) over A is a structure $\mathcal{A} = (Q, S, \delta, F, l)$ similar to an NFA over A, except that $l : Q \to A$ labels states with letters from A. As a recogniser of words, the ST-NFA $\mathcal{A}$ accepts strings of the form $A(AA)^*$. A run of $\mathcal{A}$ on a string $w = a_0 a_1 \cdots a_{2n}$ in $A(AA)^*$, is a sequence of states $q_0, \ldots, q_{n+1}$ satisfying $q_0 \in S$, $(q_i, a_{2i}, q_{i+1}) \in \delta$ for each $i \in \{0, \ldots, n\}$, and $l(q_i) = a_{2i-1}$ for each $i \in \{1, \ldots, n\}$. The run is *accepting* if $q_{n+1} \in F$. We define the word language accepted by $\mathcal{A}$, denoted $L(\mathcal{A})$, to be the set of strings $w \in A^*$ on which $\mathcal{A}$ has an accepting run. Fig. 2 shows an ST-NFA over the alphabet $\{a, b, c\}$ which accepts the word language $a(abccaa)^*$. We will use the convention that start states are indicated by sourceless incoming arrows, and final states are indicated by double circles.

An ST-NFA $\mathcal{A}$ also generates signals in a natural way: we begin by taking a transition emanating from the start state, emitting its label, and then spend time at the resulting state emitting its label all the while before taking a transition again, and so on till we choose to stop at a final state. The language of signals generated by an ST-NFA $\mathcal{A}$ is defined to be $timing(L(\mathcal{A}))$, and will be denoted by $S(\mathcal{A})$. The signal shown in Fig. 1 is accepted by the ST-NFA shown in Fig. 2.

We say that an ST-NFA $\mathcal{A} = (Q, S, \delta, F, l)$ is *deterministic* if S is singleton, and there do not exist states p, q and r, with $q \neq r$, such that $(p, a, q) \in \delta$ and $(p, a, r) \in \delta$ with $l(q) = l(r)$. We denote the class of deterministic ST-NFA's by ST-DFA.

Here are some properties of proper words which will be useful in what follows.

Proposition 3.1. *Let A be an alphabet.*

1. *If w and w' are proper words over A then $timing(w) \cap timing(w') \neq \emptyset$ iff $w = w'$.*
2. *$timing(Prop(A)) = Sig(A)$.*
3. *The word language $Prop(A)$ is ST-DFA-definable.*

Proof. Parts 1 and 2 follow easily from the definitions. For part 3, the required ST-DFA $\mathcal{A}_{Prop}$ for the alphabet $\{a, b\}$ is shown in Fig. 3. □

Let us call a state in an ST-NFA "originating" if it has no incoming transitions, and "terminating" if it has no outgoing transitions. We say a transition $p \xrightarrow{a} q$ in an ST-NFA $\mathcal{A} = (Q, S, \to, F, l)$ is *non-proper* if $l(p) = l(q) = a$, *except* for the case when p is originating or q is terminating. We say the ST-NFA $\mathcal{A}$ is *proper* if it has no non-proper transitions. The ST-NFA of Fig. 2 is proper. Clearly for a proper ST-NFA $\mathcal{A}$ over A we have $L(\mathcal{A}) \subseteq Prop(A)$.

Lemma 3.1. *For every ST-NFA $\mathcal{A}$ over an alphabet A there is a signal language equivalent* proper *ST-NFA $\mathcal{A}'$ over A (i.e. $S(\mathcal{A}') = S(\mathcal{A})$).*
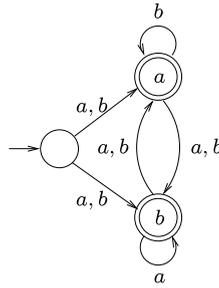
Fig. 3. ST-DFA $\mathcal{A}_{Prop}$ accepting the word language $Prop(\{a, b\})$.
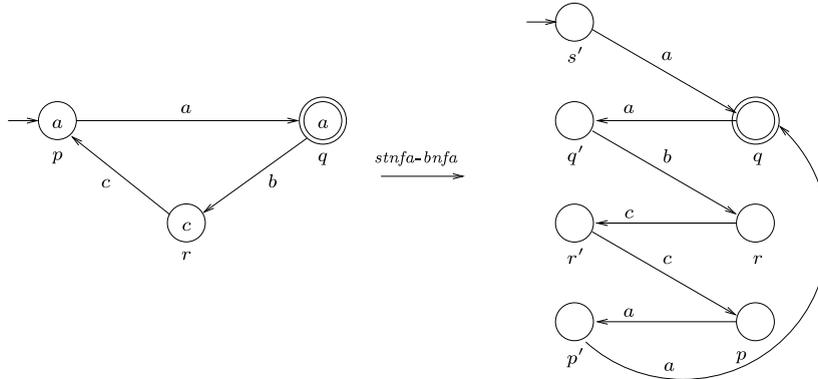

Fig. 4. The translation *stnfa-bnfa*.

Proof. Let $\mathcal{A} = (Q, S, \rightarrow, F, l)$. We modify $\mathcal{A}$ as follows:

1. First we transform $\mathcal{A}$ to $\mathcal{A}'$ by making the start states originating and final states terminating. This can be done by adding a new start state s' and a new final state f', and adding transitions $s' \xrightarrow{a} p$ for each transition $s \xrightarrow{a} p$ in $\mathcal{A}$ with $s \in S$, and transitions $p \xrightarrow{a} f'$ for each transition $p \xrightarrow{a} f$ in $\mathcal{A}$ with $f \in F$.
2. Next, we transform $\mathcal{A}'$ to $\mathcal{A}''$ as follows: Pick a non-proper transition $p \xrightarrow{a} q$ and a transition $r \xrightarrow{b} p$, and add the transition $r \xrightarrow{b} q$. Repeat this till no more edges can be added.
3. Now we drop all non-proper edges in $\mathcal{A}''$ to obtain the required proper ST-NFA $\mathcal{B}$.

Step 1 clearly preserves the word (and hence signal) language of $\mathcal{A}$. Step 2 clearly preserves the signal language of $\mathcal{A}'$. Step 3 also preserves the signal language of $\mathcal{A}''$, since any signal σ generated by $\mathcal{A}''$ using a run of non-proper edges can be simulated by using a single proper edge in $\mathcal{A}''$. □

We now want to show some closure properties of the word and signal languages accepted by ST-NFA's. For this it will be useful to go over to a class of classical NFA's which we call "bipartite" NFA's.

A *bipartite* NFA, or B-NFA for short, over an alphabet A is an NFA $\mathcal{B} = (Q, S, \delta, F)$ over A such that there exists a partition of the set of states Q into Q_1 and Q_2 satisfying

- $S \subseteq Q_1$ and $F \subseteq Q_2$, and
- $\delta \subseteq (Q_1 \times A \times Q_2) \cup (Q_2 \times A \times Q_1)$.

ST-NFA's and B-NFA's accept the same class of word languages. To see this we show how we can go from an ST-NFA to a language equivalent B-NFA and vice versa. Let $\mathcal{A} = (Q, S, \delta, F, l)$ be an ST-NFA over A. We define the B-NFA $\mathcal{B}$ corresponding to $\mathcal{A}$, denoted *stnfa-bnfa*($\mathcal{A}$), as follows. The states of $\mathcal{B}$ are $(\{s'\} \cup Q) \cup \{q' \mid q \in Q\}$, where s' is a new start state, the finals states are F, and we have transitions $s' \xrightarrow{a} q$ whenever $(p, a, q) \in \delta$ with $p \in S$; plus transitions of the form $p' \xrightarrow{a} q$ for each transition $(p, a, q) \in \delta$; plus transitions of the form $q \xrightarrow{l(q)} q'$ for each $q \in Q$. Fig. 4 shows the translation applied to the example ST-NFA of Fig. 2.

Conversely, given a B-NFA $\mathcal{B}$, we can give an ST-NFA $\mathcal{A}$, denoted *bnfa-stnfa*($\mathcal{B}$), whose word language is the same as that of $\mathcal{B}$. Let Q_1 and Q_2 be the assumed partition of states of $\mathcal{B}$. The states of $\mathcal{A}$ comprise the *transitions* of $\mathcal{B}$ which go from Q_2 to Q_1, an initial state s, and a final state f if there is a final state of $\mathcal{B}$ which is terminating. Each state is labelled by the label of the transition to which it corresponds. The start state s and terminal final state f can be labelled by any letter as their labels will not contribute to the language of $\mathcal{A}$. The set of final states comprise the final state f above, along with the states corresponding to the transitions going out of final states of $\mathcal{B}$. There is a transition from $e_1 = (p, b, q)$ and $e_2 = (r, c, t)$ on
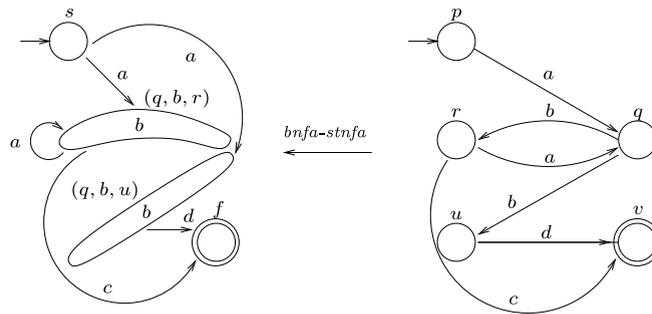
Fig. 5. The translation *bnfa-stnfa.*.

a in $\mathscr{A}$ if there is a transition (q, a, r) in $\mathscr{B}$. There is also a transition (s, a, e_2) in $\mathscr{A}$ if there is a transition (p, a, q) out of an initial state of $\mathscr{B}$ and e_2 is a transition out of q in $\mathscr{B}$. Finally we also have a transition from (p, a, q) to f labelled b, if (q, b, r) is a transition in $\mathscr{B}$. It is not difficult to see that the word languages of $\mathscr{B}$ and $\mathscr{A}$ are the same. Fig. 5 shows the translation *bnfa-stnfa* applied to the B-NFA on the right.

We first note some closure properties of B-NFA's which in turn will help us to show closure properties of ST-NFA's. We say a B-NFA is *deterministic*, and call it a B-DFA, if it is also a DFA.

Proposition 3.2. *Let A be an alphabet.*

1. *The class of* B-NFA*'s is determinizable.*
2. *The class of word languages accepted by* B-NFA*'s over A is closed under the boolean operations of union, intersection, and complementation with respect to* $A(AA)^*$.

Proof. Given a B-NFA $\mathscr{B}$, we can apply the standard subset construction to determinize it to get a language equivalent DFA $\mathscr{B}'$. The subset construction preserves the bipartite structure of $\mathscr{B}$, and hence $\mathscr{B}'$ is also a B-DFA.

For closure under intersection, we note that the standard product construction also preserves the bipartite structure of the two B-NFA's. For closure under complementation with respect to $A(AA)^*$, given a B-NFA $\mathscr{B}$ we can first determinize it to get a language equivalent B-DFA $\mathscr{B}'$. Let the partition on states of $\mathscr{B}'$ be Q_1 and Q_2. We can now "complete" $\mathscr{B}'$ with respect to $A(AA)^*$ by adding two new states d_1 and d_2, with d_1 added to Q_1 and d_2 added to Q_2, and adding transitions (p_1, a, d_2) for each state p_1 in Q_1 with no outgoing transition on a, and similarly (p_2, a, d_1) for each state p_2 in Q_2 with no outgoing transition on a. We also add transitions (d_1, a, d_2) and (d_2, a, d_1) for each a in A. Finally, we simply "flip" the final states in Q_2: i.e. the new set of final states F' is $Q_2 - F$, where F is the set of final states of $\mathscr{B}'$. The resulting automaton is a B-DFA which accepts the complement (with respect to $A(AA)^*$) of the language accepted by $\mathscr{B}$. $\square$

It follows that the class of word languages accepted by ST-NFA's over an alphabet A coincides with the class of word languages definable by B-NFA's over A, which in turn is precisely the class of regular subsets of $A(AA)^*$, and further that the class of word languages accepted by ST-NFA's over A are closed under union, intersection, and complement wrt $A(AA)^*$.

Using these observations we can now prove some closure properties of the class of ST-NFA-definable signal languages.

Lemma 3.2. *Let A be an alphabet. Then*

1. *The class of* ST-NFA*'s over A is determinizable: that is, for any* ST-NFA *over A we can give an* ST-DFA *which accepts the same signal language.*
2. *The class of signal languages definable by* ST-NFA*'s over an alphabet A is closed under union, intersection, and complement.*

Proof. For the first part, let $\mathscr{A}$ be an ST-NFA over A. We can determinize $\mathscr{A}$ as follows: We first go over to the word language equivalent B-NFA $\mathscr{B}$ by applying the translation *stnfa-bnfa* to $\mathscr{A}$. We now determinize $\mathscr{B}$ to get a B-DFA $\mathscr{B}'$. Finally we apply the translation *bnfa-stnfa* to $\mathscr{B}'$ to obtain an ST-NFA $\mathscr{A}'$. In fact $\mathscr{A}'$ is an ST-DFA. To see this, suppose there were states $e = (p, c, q), e_1 = (p_1, b, q_1)$, and $e_2 = (p_2, b, q_2)$ in $\mathscr{A}'$, with e_1 and e_2 distinct, and transitions (e, a, e_1) and (e, a, e_2) on some $a \in A$. Then, since e_1 and e_2 are distinct, it must be the case that p_1 and p_2 are distinct, otherwise it would contradict the fact that $\mathscr{B}'$ was deterministic. But then we have transitions (q, a, p_1) (q, a, p_2) in $\mathscr{B}'$ with $p_1 \neq p_2$, which contradicts the fact that $\mathscr{B}'$ is a B-DFA.

For the second part, closure under union is immediate. For closure under complementation, let $\mathscr{A}$ be an ST-NFA over A. We first make $\mathscr{A}$ proper, to get a signal equivalent proper ST-NFA $\mathscr{A}'$. Then $Sig(A) - S(\mathscr{A}) = Sig(A) - S(\mathscr{A}') = timing(Prop(A) - L(\mathscr{A}'))$ (using Proposition 3.1) $= timing(L(\mathscr{A}''))$ for some ST-NFA $\mathscr{A}''$ (using closure properties of ST-NFA-definable word languages) $= S(\mathscr{A}'')$. The closure under intersection follows from that of union and complementation. $\square$

4. Equivalence of ST-NFA's and MSO^s

In this section, we introduce a natural monadic second-order logic interpreted over signals and show that the class of signal languages it defines coincides with the class of signal languages definable by ST-NFA's.

In the logics to follow, we assume a countable supply of first-order variables and second-order variables. For an alphabet A, the syntax of monadic second-order logic over A, denoted $\mathrm{MSO}^s(A)$, is given by:

$$\varphi ::= Q_a(x) \mid x < y \mid x \in X \mid \neg\varphi \mid (\varphi \vee \varphi) \mid \exists x \varphi \mid \exists X \varphi,$$

where $a \in A$, x, and y are first-order variables and X is a second-order variable.

We interpret a formula φ of the logic over a signal σ in $Sig(A)$, along with an interpretation $\mathbb{I}$ with respect to σ which assigns to each first-order variable a value in $[0, dur(\sigma))$, and to each set variable a *finitely varying* subset of $[0, dur(\sigma))$. We use $X \subseteq_{fv} Y$ to denote that X is a finitely varying subset of Y. For an interpretation $\mathbb{I}$, we use the notation $\mathbb{I}[t/x]$ to denote the interpretation which sends x to t and agrees with $\mathbb{I}$ on all other variables. Similarly, $\mathbb{I}[B/X]$ denotes the modification of $\mathbb{I}$ which maps the set variable X to a subset B of $\mathbb{R}_{\geq 0}$, and the rest to the same as that mapped by $\mathbb{I}$. We also use the notation $[t/x]$ to denote an interpretation which sends x to t when the rest of the interpretation is irrelevant.

Given a formula $\varphi \in \mathrm{MSO}^s(A)$, $\sigma \in Sig(A)$, and an interpretation $\mathbb{I}$ with respect to σ to the variables in φ, the satisfaction relation $\sigma, \mathbb{I} \models \varphi$, is defined inductively as:

$$
\begin{aligned}
\sigma, \mathbb{I} &\models Q_a(x) &&\text{iff} && \sigma(\mathbb{I}(x)) = a, \text{ where } a \in A. \\
\sigma, \mathbb{I} &\models x < y &&\text{iff} && \mathbb{I}(x) < \mathbb{I}(y). \\
\sigma, \mathbb{I} &\models x \in X &&\text{iff} && \mathbb{I}(x) \in \mathbb{I}(X). \\
\sigma, \mathbb{I} &\models \neg\varphi &&\text{iff} && \sigma, \mathbb{I} \not\models \varphi. \\
\sigma, \mathbb{I} &\models \varphi_1 \vee \varphi_2 &&\text{iff} && \sigma, \mathbb{I} \models \varphi_1 \text{ or } \sigma, \mathbb{I} \models \varphi_2. \\
\sigma, \mathbb{I} &\models \exists x \varphi &&\text{iff} && \exists t \in [0, dur(\sigma) : \sigma, \mathbb{I}[t/x] \models \varphi. \\
\sigma, \mathbb{I} &\models \exists X \varphi &&\text{iff} && \exists B \subseteq_{fv} [0, dur(\sigma) : \sigma, \mathbb{I}[B/X] \models \varphi.
\end{aligned}
$$

For a sentence φ (a formula without free variables) in $\mathrm{MSO}^s(A)$, the interpretation does not play any role, and we write the satisfaction relation $\sigma, \mathbb{I} \models \varphi$ as simply $\sigma \models \varphi$. We define the language of signals defined by φ to be $S(\varphi) = \{\sigma \in Sig(A) \mid \sigma \models \varphi\}$.

As an example, the formula

$$\varphi_{cont}(x) = \exists y \exists z \left(y < x \wedge x < z \wedge \bigvee_{a \in A} \forall u (y < u \wedge u < z \Rightarrow Q_a(u)) \right)$$

asserts that the point x is a point of continuity. The formula $\varphi_{disc}(x) = \neg\varphi_{cont}(x)$ asserts that x is a point of discontinuity.

We denote by $\mathrm{FO}^s(A)$ the first-order fragment of $\mathrm{MSO}^s(A)$ obtained by disallowing second-order quantification and atomic formulas of the form $x \in X$.

Theorem 4.1. *A signal language over an alphabet A is definable by a $\mathrm{MSO}^s(A)$ sentence iff it is definable by an ST-NFA over A.*

We prove this theorem in the rest of this section. The proof proceeds in a similar manner to the proof of Büchi's MSO characterisation of classical automata [5] (see [19]).

We first show how to go from a formula in $\mathrm{MSO}^s(A)$ to an equivalent ST-NFA over A. We will represent models of formulas with free variables in them as signals with the interpretations built into them. We assume an ordering on the countable set of first-order variables given by $x_1, x_2, \ldots$, and similarly $X_1, X_2, \ldots$ for the set variables. For a formula φ with first-order free variables among $U = \{x_{i_1}, \cdots, x_{i_m}\}$ and second-order free variables among $V = \{X_{j_1}, \ldots, X_{j_n}\}$ (in order), we represent a signal σ and an interpretation $\mathbb{I}$ as a signal $\sigma_{\mathbb{I}}^{U,V} : [0, dur(\sigma) \rightarrow A \times \{0, 1\}^{m+n}$ given by $\sigma_{\mathbb{I}}^{U,V}(t) = (\sigma(t), b_1, \cdots, b_{m+n})$, where for $k \in \{1, \ldots, m\}$, $b_k = 1$ iff $\mathbb{I}(x_{i_k}) = t$, and for $k \in \{m+1, \ldots, n\}$, $b_k = 1$ iff $t \in \mathbb{I}(X_{j_{k-m}})$. Thus for a formula φ with free variables in (U, V), we have the notion of a (U, V)-model of φ.

We note that for the U, V above, not every signal over $A \times \{0, 1\}^{m+n}$ is a "valid" (U, V)-model, in the sense that the components of the signal corresponding to a first-order variable in U may not have *exactly* one 1 entry. Nonetheless, the proposition below says that the valid (U, V)-models are ST-NFA-recognisable.

Proposition 4.1. *Let A be a finite alphabet and let U and V be a sets of first- and second-order variables respectively. Then we can construct an ST-NFA $\mathcal{A}_{valid}^{U,V}$ which accepts precisely the set of signals over $A \times \{0, 1\}^{|U|+|V|}$ which represent valid (U, V)-models over A.*

Proof. For each $x_{i_k} \in U$ we can construct an ST-NFA over the alphabet $A \times \{0, 1\}^{|U|+|V|}$ which accepts signals which are valid encodings as far as the component corresponding to x_{i_k} is concerned. Fig. 6 shows an ST-NFA which accepts the valid models with respect to x, when the alphabet is $\{a, b\}$, and $U = \{x, y\}$ and $V = \emptyset$. We can then take the intersection of the resulting ST-NFA's to obtain the ST-NFA $\mathcal{A}_{valid}^{U,V}$. $\quad\square$

Proposition 4.2. *Let φ be an $\mathrm{MSO}^s(A)$ formula with first- and second-order free variables U and V respectively. Let $\mathcal{A}$ be an ST-NFA accepting the (U, V)-models of φ. Then for any set of variables (U', V') such that $U \subseteq U'$ and $V \subseteq V'$, we can construct an ST-NFA $\mathcal{A}'$ which accepts precisely the (U', V')-models of φ.*

Lemma 4.1. *Let φ be an $\mathrm{MSO}^s(A)$ formula and let (U, V) be the set of free variables in it. Then we can construct an ST-NFA $\mathcal{A}_\varphi^{U,V}$ which accepts precisely the (U, V)-models of φ.*
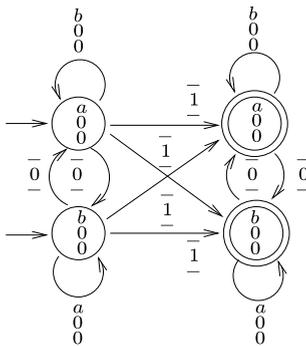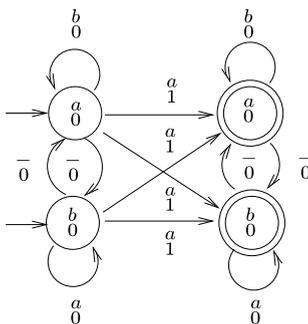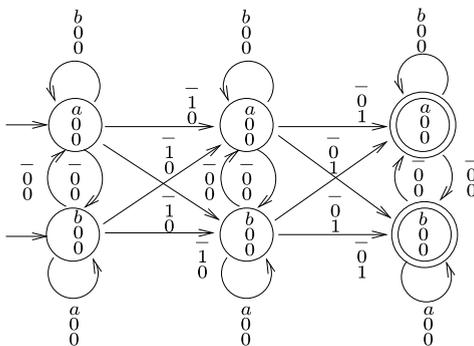
Fig. 6. ST-NFA accepting x-valid ($\{x, y\}, \emptyset$)-models over the alphabet $\{a, b\}$.

Proof. We construct the ST-NFA $\mathcal{A}_\varphi^{U,V}$ by induction on the structure of φ.

1. $\varphi = Q_a(x)$: Assuming $A = \{a, b\}$, the automaton $\mathcal{A}_\varphi^{\{x\},\emptyset}$ is shown below.

2. $\varphi = x < y$: The automaton $\mathcal{A}_\varphi^{\{x,y\},\emptyset}$ (assuming x occurs before y in the variable ordering) is:

3. For the case $\varphi = x \in X$, the automaton $\mathcal{A}_\varphi^{\{x\},\{X\}}$ is defined similarly.

4. $\varphi = \neg\psi$: Let $\mathcal{A}_\psi^{U,V}$ be the automaton for ψ, where (U, V) is the set of free variables in ψ. Then $\mathcal{A}_\varphi^{U,V}$ is the intersection of $\mathcal{A}_{valid}^{U,V}$ with the ST-NFA that recognises the complement of the signal language of $\mathcal{A}_\psi^{U,V}$ (cf. Lemma 3.2).

5. $\varphi = \psi \vee \nu$: Let $\mathcal{A}_\psi^{U_1,V_1}$ be the ST-NFA for ψ, where (U_1, V_1) is the set of free variables in ψ, and let $\mathcal{A}_\nu^{U_2,V_2}$ be the ST-NFA for ν, where (U_2, V_2) is the set of free variables in ν. Let $U = U_1 \cup U_2$ and $V = (V_1 \cup V_2)$. By Proposition 4.2, we obtain ST-NFA's $\mathcal{A}_\psi^{U,V}$ and $\mathcal{A}_\nu^{U,V}$. Then $\mathcal{A}_\varphi^{U,V}$ is the ST-NFA that accepts the union of the signal languages accepted by $\mathcal{A}_\psi^{U,V}$ and $\mathcal{A}_\nu^{U,V}$.

6. $\varphi = \exists x \psi$: Let (U', V') be the set of free variables in ψ, so that $U = U' - \{x\}$ and $V = V'$. Let $\mathcal{A}_\psi^{U',V'}$ be an ST-NFA for ψ. Now we simply project away the component corresponding to x in the symbols labelling the transitions and states of $\mathcal{A}_\psi^{U',V'}$ to obtain the required ST-NFA $\mathcal{A}_\varphi^{U,V}$.

7. $\varphi = \exists X \psi$: Let (U', V') be the set of free variables in ψ, so that $U = U'$ and $V = V' - \{X\}$. Let $\mathcal{A}_\psi^{U',V'}$ be an ST-NFA for ψ. Again we simply project away the component corresponding to X in the symbol labelling the transitions and states of $\mathcal{A}_\psi^{U',V'}$ to obtain the required counter-free ST-NFA $\mathcal{A}_\varphi^{U,V}$. □

From the above lemma, it now follows that for a sentence $\varphi \in \text{MSO}^s(A)$ we have an ST-NFA $\mathcal{A}_\varphi$ over A such that $S(\varphi) = S(\mathcal{A}_\varphi)$.

We now prove the converse direction of Theorem 4.1. Let $\mathcal{A} = (Q, S, \rightarrow, F, l)$ be an ST-NFA over A. Without loss of generality we assume that $\mathcal{A}$ is proper. We give an $\mathrm{MSO}^s(A)$ sentence $\varphi_{\mathcal{A}}$ such that $S(\mathcal{A}) = S(\varphi_{\mathcal{A}})$. The sentence $\varphi_{\mathcal{A}}$ describes the existence of an accepting run of the automaton on a given signal. Let $\{e_i = p_i \overset{a_i}{\rightarrow} q_i \mid i = 1, \ldots, m\}$ be the set of transitions in $\mathcal{A}$. The second-order variables $X_1, \ldots, X_m$ will be used to capture the points in the signal at which the transitions $e_1, \ldots, e_m$ are taken respectively. Note that since we are assuming $\mathcal{A}$ is proper, the union of the X_i's must correspond exactly to the points of discontinuities in the given signal. We will use the abbreviation $consec(x, y, X)$ to mean that x and y are "consecutive" points in the set X, and define it to be:

$$consec(x, y, X) = x \in X \wedge y \in X \wedge \neg \exists z (x < z \wedge z < y \wedge z \in X).$$

We also use $first(x)$ as an abbreviation for $\neg \exists y (y < x)$ and $last(x)$ as an abbreviation for $\neg \exists y (x < y)$.

The formula $\varphi_{\mathcal{A}}$ is given below. We assume that i and j range over $0, \ldots, m$.

$$\exists X_1 \cdots \exists X_m \exists X (\forall x \ ((x \in X \iff \bigvee_i x \in X_i) \ \wedge$$
$$(\bigwedge_{i \neq j}(x \in X_i \Rightarrow \neg x \in X_j)) \ \wedge$$
$$(x \in X \iff disc(x)) \ \wedge$$
$$(first(x) \Rightarrow \bigvee_{i: p_i \in S} x \in X_i) \ \wedge$$
$$(last(x) \Rightarrow \bigvee_{i: q_i \in F} x \in X_i) \ \wedge$$
$$(\bigwedge_i(x \in X_i \Rightarrow (Q_{a_i}(x) \wedge ((\exists y(consec(x, y, X))) \Rightarrow$$
$$\forall z((x < z \wedge z < y) \Rightarrow Q_{l(q_i)}(z)))))))))).$$

This completes the proof of Theorem 4.1.

Before we close this section, we observe that the version of MSO^s called *weak* MSO^s, in which we restrict the second-order quantification to *finite* subsets of the domain of the signal (rather than finitely varying subsets), is as expressive as the version we have defined. The justification is as follows. We note that the clause $x \in X \iff disc(x)$ forces the second-order variables X and X_i's to be interpreted as *finite* subsets of the domain since the signal model has only finitely many discontinuities. Hence quantification over finite subsets suffices to capture the ST-NFA-definable signal languages. Further, signal languages definable by second-order quantification restricted to finite subsets are clearly ST-NFA-definable (by an argument similar to the one above, where we allow components corresponding to second-order variables to have 1's only on transition (and not state) labels). Hence the expressiveness of the two variants coincide with ST-NFA-definable signal languages.

Corollary 4.1. *The class of signal languages definable in* $\mathrm{MSO}^s(A)$ *and weak* $\mathrm{MSO}^s(A)$ *coincide.*

5. Counter-free signal languages

In this section we introduce a counter-free version of signal languages which will be shown in the next section to characterise FO^s-definable signal languages.

We recall that a *counter* in an NFA $\mathcal{A}$ is a sequence of distinct states $q_0, \ldots, q_n$ with $n \geq 1$, along with a word $u \in A^*$, such that there is a path labelled u in $\mathcal{A}$ from q_i to q_{i+1} (for each $i \in \{0, \ldots, n-1\}$) and from q_n to q_0. We call the counter an *even-counter* if u has even length (i.e $u \in (AA)^+$). An NFA is said to be *counter-free* (respectively *even-counter-free*) if it does not contain a counter (respectively even-counter). A regular language is said to be *counter-free* if there exists a counter-free NFA for it.

We now define the counter-free version of ST-NFA's. A counter in an ST-NFA is similar to one in an NFA except that by the "label" of a path in the automaton, we mean the sequence of alternating transition and state labels along the path. The label of the path $q_0 \overset{a_0}{\rightarrow} q_1 \overset{a_1}{\rightarrow} \cdots q_n \overset{a_n}{\rightarrow} q_{n+1}$ is $a_0 l(q_1) a_1 \cdots l(q_n) a_n l(q_{n+1})$.[1] Thus a *counter* in an ST-NFA $\mathcal{A}$ is a sequence of distinct states $q_0, \ldots, q_n$ with $n \geq 1$, along with a word $u \in A^*$, such that there is a path labelled u in $\mathcal{A}$ from q_i to q_{i+1} (for each $i \in \{0, \ldots, n-1\}$) and from q_n to q_0. We say an ST-NFA is *counter-free* if it does not contain a counter. We say a signal language is *counter-free* if it is definable by a counter-free *proper* ST-NFA. The properness requirement is important as without it we can give counter-free ST-NFA's for signal languages we would otherwise like to consider as not being counter-free. Fig. 7 shows a non-proper counter-free ST-NFA and its equivalent proper version which contains a counter.

We will show in the next section that the class of first-order definable signal languages coincide with the class of counter-free signal languages. Our aim in the rest of this section is to show some closure properties of counter-free signal languages that will be useful there.

We first observe that the classical subset construction for determinizing NFA's preserves counter-freeness.

Lemma 5.1. *Let $\mathcal{B}$ be an* NFA *and let $\mathcal{C}$ be the* DFA *obtained by the standard subset construction on $\mathcal{B}$. Then if $\mathcal{B}$ was counter-free, so is $\mathcal{C}$. Also, if $\mathcal{B}$ was* even-*counter-free, so is $\mathcal{C}$.*

---

[1] we ignore the label of the first state in the path, but count the label of the last state in the path.
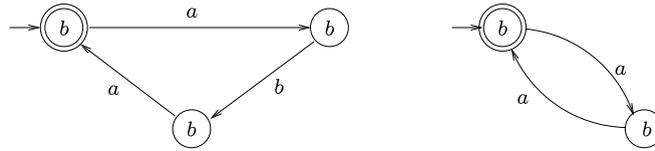
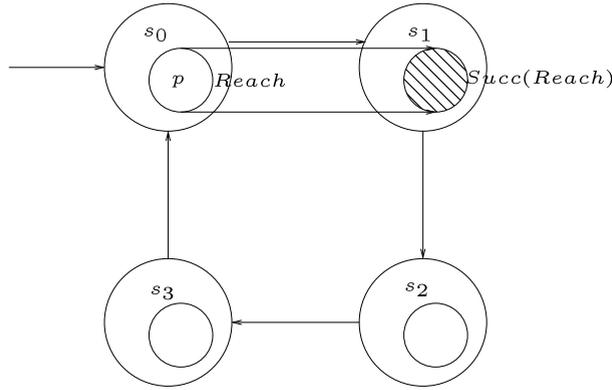**Fig. 7.** Example showing how proper conversion may introduce counters.

Fig. 8. Example depicting the set *Reach* and *Succ*(*Reach*).

Proof. Suppose $\mathcal{C}$ has a counter. We show that $\mathcal{B}$ has a counter.

Let $S_0, S_1, \ldots, S_{n-1}$ be a counter in $\mathcal{C}$, and let w be the word associated with it, i.e, S_0 on w goes to S_1, S_1 on w goes to S_2, and so on. Choose the counter such that $|S_0| + |S_1| + \cdots + |S_{n-1}|$ is minimum. For a set X of states of $\mathcal{B}$, define $Pred(X)$ to be the set of all states y of $\mathcal{B}$ such that y on w goes to some state in X. We will write $Pred(x)$ for $Pred(\{x\})$. Similarly $Succ(X)$ is the set of all states reachable from states in X on reading w.

Form a sequence of states $q_0, q_1, \ldots$ of $\mathcal{B}$ as follows. Begin with some state q_0 in S_0. Let $q_i \in S_j$. Then q_{i+1} is some state in $Pred(q_i) \cap S_{j-1}$ (where $j-1$ is taken modulo n). q_{i+1} is chosen to be different from q_i whenever possible. Consider the sequence of states $q_0, q_n, q_{2n} \ldots$ and let $l < k$ be such that $q_{nl} = q_{nk}$ (such an l and k exist as each subset is finite). If for some $nk \leq i < j < nk, q_i \neq q_j$ then $q_{nl}, \ldots, q_{nk}$ contains a counter on w and we are done.

Suppose this is not the case, i.e. for all $i \in \{nl, \ldots, nk\}$ q_i is equal to say p. Then from the construction of the sequence one can easily see that for all $i \in \{0, \ldots, n-1\}$ p is in every S_i. We also note that $Pred(p) \cap S_i = p$ since otherwise we would have chosen a predecessor which is different from p. Define $Reach$ to be the smallest set containing p and closed with respect to $Succ$, i.e. if $q \in Reach$ and $q' \in Succ(q)$, then $q' \in Reach$. The fact that p is in every S_i implies that $Reach$ is a subset of every S_i (See Fig. 8).

Let $Y_0 = S_0 - \{p\}$ and for every $j \geq 0$ let $Y_{j+1} = Succ(Y_j)$. One can inductively argue that for every $j \geq 0, S_i - Reach \subseteq Y_{i+nj}$ and therefore $Y_{i+nj} \neq Y_{i+nj+1}$ because otherwise $S_i = S_{i+1}$ ($i+1$ taken modulo n) as $S_i = Reach \cup Y_{i+nj}$. Since all the Y_{i+nj}'s cannot be distinct there exist $k < l$ such that $Y_{i+nk} = Y_{i+nl}$. Let $X_0 = Y_{i+nk}$ and let $X_{j+1} = Succ(X_j)$. Note that we always maintain the invariant $p \notin Y_j$ and therefore none of the X_j's contain p. It can be seen that $X_0, \ldots, X_{n-1}$ forms a counter in $\mathcal{C}$ such that $|X_0| + \cdots + |X_{n-1}| < |S_0| + \cdots + |S_{n-1}|$, which contradicts the choice of the counter.

We note that it also follows from the above argument that if $\mathcal{B}$ had no *even* counters, $\mathcal{C}$ will also not have any even counters. □

Next, as we did for ST-NFA's, it will be convenient to characterise CF-ST-NFA's in terms of bipartite NFA's. We define the class of *even-counter-free* B-NFA's over an alphabet A, denoted ECF-B-NFA, to be the class of B-NFA's which have no even-counters.

Proposition 5.1. *The class of word languages accepted by* CF-ST-NFA*'s and* ECF-B-NFA*'s over an alphabet A coincide.*

Proof. It is easy to verify that the translations *stnfa-bnfa* and *bnfa-stnfa* given in Section 3 already give us the required translations: that is if $\mathcal{A}$ is a CF-ST-NFA over A then *stnfa-bnfa*($\mathcal{A}$) is an ECF-B-NFA, and if $\mathcal{B}$ is an ECF-B-NFA then *bnfa-stnfa*($\mathcal{B}$) is indeed a CF-ST-NFA. □

Lemma 5.2. *The class of even-counter-free bipartite* NFA*'s over A are determinizable and closed under union, intersection, and complementation wrt $A(AA)^*$.*

Proof. Given an ECF-B-NFA $\mathcal{B}$ over A, we determinize it using the subset construction to get $\mathcal{A}'$. We have already argued that $\mathcal{A}'$ is a B-DFA over A. By Lemma 5.1 $\mathcal{A}'$ is also even-counter-free, and we are done.

To show closure under intersection, we argue that the above properties are preserved by the standard product construction for intersection. Let $\mathcal{B}$ and $\mathcal{C}$ be two ECF-B-NFA's, with B_1, B_2 and C_1, C_2 being the respective partitions. In the product automaton $\mathcal{D}$ accepting the intersection of their word languages, the only reachable states are in $D_1 = B_1 \times C_1$ and $D_2 = B_2 \times C_2$, with D_1 and D_2 being the partitions. It remains to be shown that $\mathcal{D}$ does not have an even-counter. Suppose $\mathcal{D}$ has such a counter with the sequence $(s_1, t_1) \cdots (s_n, t_n)$. Then it cannot be the case that all the s_i's are the same and all the t_j's are the same, since otherwise the sequence is not a counter. Assume without loss of generality that all the s_i's are not same. Then there is a subsequence of distinct states $s_{i_1}, s_{i_2}, \ldots, s_{i_k}$ which forms an even-counter in $\mathcal{B}$. This contradicts our assumption that $\mathcal{B}$ was even-counter-free.

To show closure under complementation wrt $A(AA)^*$, we first determinize the automaton. As observed in the proof of Proposition 3.2, the resulting automaton is a B-DFA. Furthermore, by Lemma 5.1, it does not contain an even-counter. Once again, we "complete" the automaton as in the proof of Lemma 3.2. It is easy to see that this completion does not introduce any even counters. We can now "flip" the final states in the second partition, to obtain a ECF-B-DFA which accepts the complement of the word language of the given B-NFA wrt $A(AA)^*$. $\square$

Using the preceding lemmas we can now argue that:

Lemma 5.3. *The class of counter-free signal languages over an alphabet A is determinizable, and closed under union, intersection, and complementation.*

Proof. To see that the class of CF-ST-NFA's is determinizable, let $\mathcal{A}$ be a counter-free ST-NFA. We go over to a word language equivalent ECF-B-NFA $\mathcal{B}$ from $\mathcal{A}$ using the translation *stnfa-bnfa*. We now determinize $\mathcal{B}$ to get $\mathcal{B}'$, and applying *bnfa-stnfa* on $\mathcal{B}'$, we get a deterministic counter-free ST-NFA.

For the closure under boolean operations, let F_1 and F_2 be two counter-free signal languages over the alphabet A. Let L_1 and L_2 be the word languages of their defining proper counter-free ST-NFA's. Once again, using Proposition 3.1, we can convince ourselves that $F_1 \cup F_2 = timing(L_1 \cup L_2)$, $F_1 \cap F_2 = timing(L_1 \cap L_2)$, and $Sig(A) - F_1 = timing(Prop(A) - L_1) = timing((A(AA)^* - L_1) \cap Prop(A))$.

Using the closure properties of the word languages accepted by CF-ST-NFA's (or equivalently ECF-B-NFA's), and the fact that $Prop(A)$ is ECF-B-NFA-definable, we can conclude that the signal languages definable by CF-ST-NFA's are closed under boolean operations. $\square$

6. Counter-free characterisation of FO signal languages

In this section, our aim is to show that FO^s-definable signal languages, counter-free signal languages, and temporal logic definable signal languages, all coincide.

We recall briefly the temporal logic LTL and its two interpretations, one over discrete words and the other over signals. For an alphabet A, the syntax of LTL(A) is given by:

$$\theta ::= a \mid (\theta U \theta) \mid (\theta S \theta) \mid \neg\theta \mid (\theta \vee \theta),$$

where $a \in A$. The logic is interpreted over words in A^*, with the following semantics. Given a word $w = a_0 \cdots a_n$ in A^* and a position $i \in \{0, \ldots, n\}$, we say $w, i \models a$ iff $a_i = a$; and $w, i \models \theta U \eta$ iff there exists j such that $i < j \leq n$, $w, j \models \eta$ and for all k such that $i < k < j$, $w, k \models \theta$. The "since" operator S is defined in a symmetric way to U in the past, and Boolean operators in the usual way. We denote by $L(\theta)$ the set $\{w \in A^* \mid w, 0 \models \theta\}$.

The logic LTL can also be interpreted over functions as done in [10]. Here we restrict the models to finitely varying functions in $Sig(A)$ and we denote this logic by $LTL^s(A)$. Given a signal $\sigma \in Sig(A)$, $t \in [0, dur(\sigma))$, and $\theta \in LTL^s(A)$, the satisfaction relation $\sigma, t \models \theta$ is defined as follows:

$$\begin{aligned}
\sigma, t &\models a &\text{iff}\quad &\sigma(t) = a. \\
\sigma, t &\models \theta U \eta &\text{iff}\quad &\exists t' : t < t' \leq dur(\sigma),\ \sigma, t' \models \eta,\ \text{and } \forall t'' : t < t'' < t', \sigma, t'' \models \theta. \\
\sigma, t &\models \theta S \eta &\text{iff}\quad &\exists t' : 0 \leq t' < t,\ \sigma, t' \models \eta, \forall t'' : t' < t'' < t,\ \sigma, t'' \models \theta.
\end{aligned}$$

Boolean operators are interpreted in the expected way. We set $S(\theta) = \{\sigma \in Sig(A) \mid \sigma, 0 \models \theta\}$.

As an example, the $LTL^s(A)$ formulas $\theta_{cont} = \bigvee_{a \in A}(a \wedge (aSa) \wedge (aUa))$ and $\theta_{disc} = \neg\theta_{cont}$ characterise the points of continuity and discontinuity respectively in a signal over A.

Theorem 6.1. *Let A be an alphabet, and let F be a signal language over A. Then the following statements are equivalent:*

1. *F is definable by an $FO^s(A)$-sentence.*
2. *F is definable by a counter-free proper ST-NFA over A.*
3. *F is definable by an $LTL^s(A)$-formula.*

The rest of this section is devoted to a proof of this theorem. Our proof will factor through some classical results connecting counter-free languages and temporal logics. The route we follow is given schematically in the figure below, via steps labelled (a), and (b) to (e).
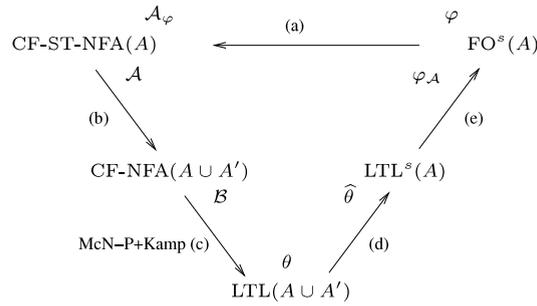
Step (a): We show how to go from a formula in $FO^s(A)$ to a counter-free proper ST-NFA, accepting exactly its models. It can be checked that the inductive construction carried out in Section 4 for Theorem 4.1 produces a counter-free proper ST-NFA at each step. This is true for the base cases $Q_a(x)$ and $x < y$. For Boolean operators, it follows by the closure properties of counter-free signal languages (cf. Lemma 5.3).

For the case of first-order quantification, let $\varphi = \exists x \psi$. Let $\mathcal{A}_\psi$ be a counter-free proper ST-NFA which accepts the valid models for ψ. Without loss of generality, we assume that $\mathcal{A}_\psi$ has no unreachable or dead states, and that its start and final states are respectively originating and terminating.

We now project away the x-component in transition and state labels of $\mathcal{A}_\psi$ to get an ST-NFA $\mathcal{A}'$ accepting the valid models of $\exists x \psi$. Now we can argue that $\mathcal{A}'$ cannot have a counter. If it did, then there are two cases: either the counter is such that no symbol in it was obtained by projecting away a '1' in the x-component, or there is a symbol in it which was obtained by projecting away a '1' in the x-component. In the first case, this would mean a counter in $\mathcal{A}_\psi$ itself, contradicting the inductive assumption that $\mathcal{A}_\psi$ was counter-free. In the second it would mean $\mathcal{A}_\psi$ has a cycle containing a transition on a symbol with a '1' in the x-component, which would contradict the validity of the models generated by $\mathcal{A}_\psi$.

However, we are not yet done as $\mathcal{A}'$ might have a non-proper edge. Now let us make $\mathcal{A}'$ proper to get $\mathcal{A}''$ using the algorithm described in Section 3. Recall that the algorithm adds edges and finally deletes all the non-proper edges. But it satisfies the property that the state space is the same, and every added edge from p to q has a corresponding path from p to q in $\mathcal{A}'$ which uses a non-proper transition.

Now we claim that $\mathcal{A}''$ is counter-free (and, by construction, proper). Suppose $\mathcal{A}''$ had a counter on states $q_0, \ldots, q_n$ on a string u. Now two possibilities exist:

- No u path in the counter uses an "added" edge. In this case this would be a counter in $\mathcal{A}'$ also, which is a contradiction.
- Some u path in the counter uses an "added" edge. So in $\mathcal{A}'$ the u-path has a corresponding u'-path which uses a non-proper edge in $\mathcal{A}'$. But non-proper edges in $\mathcal{A}'$ could only have come from a projection of a '1' in the x-component of a transition in $\mathcal{A}_\psi$. So the corresponding "unprojected" u'-path contains a symbol with a '1' in the x-component in $\mathcal{A}_\psi$. Once again, being part of a loop in $\mathcal{A}''$ and hence also in $\mathcal{A}_\psi$, this contradicts the validity of $\mathcal{A}_\psi$.

This completes the inductive proof of the claim that the set of signal models of a first-order formula is counter-free.

Steps (b) to (d) prove that we can go from an arbitrary counter-free proper ST-NFA $\mathcal{A}$ over the alphabet A to a signal-language-equivalent $FO^s(A)$-sentence $\varphi_\mathcal{A}$.

Step (b): Let us denote by A' the alphabet $\{a' \mid a \in A\}$. For a proper word $w = a_0 \ldots a_{2n} \in Prop(A)$, we define $ann(w)$ to be the word $a'_0 a_1 a'_2 \ldots a_{2n-1} a'_{2n}$ in $(A \cup A')^*$, and apply it to work on subsets of $Prop(A)$ as well. Now let $\mathcal{A}$ be a counter-free proper ST-NFA over A. By the characterisation of counter-free languages in the proof of Lemma 5.3, there is a word language equivalent NFA $\mathcal{B}_0$ that is bipartite and has no counters except possibly on odd length u's. However, if we annotate the labels of the edges going from left to right (with the convention that the start states are in the left partition) by replacing each label a by a', then it is easy to see that the resulting NFA is counter-free and accepts $ann(L(A))$. Let us call this NFA over $(A \cup A')$ as $\mathcal{B}$. Thus $L(\mathcal{B}) = ann(L(\mathcal{A}))$ and is a classical counter-free word language.

Step (c): By the results due to Schutzenberger [17], McNaughton–Papert [12], and Kamp [10] for classical word languages, the class of counter-free, star-free, FO-definable, and LTL-definable word languages all coincide. Thus for a counter-free NFA $\mathcal{B}$ over $(A \cup A')$, we have an $LTL(A \cup A')$ formula θ which defines the same word language as $\mathcal{B}$. Thus $L(\mathcal{B}) = L(\theta)$.

Step (d): For a formula θ in $LTL(A \cup A')$ such that $L(\theta) \subseteq A'(AA')^*$ and the "un-annotation" of $L(\theta)$ (i.e. $ann^{-1}(L(\theta))$) is proper, we can construct a formula $ltl\text{-}ltls(\theta)$ in $LTL^s(A)$ which is such that $S(ltl\text{-}ltls(\theta)) = timing(ann^{-1}(L(\theta)))$.

We will use the abbreviation $\theta_1 U_d \theta_2$ to mean that at a point of discontinuity, "$\theta_1 U \theta_2$" is true in an untimed sense, and define it to be $(\theta_2 U \theta_2) \vee (\theta_1 U (\theta_{disc} \wedge (\theta_2 \vee (\theta_1 \wedge (\theta_2 U \theta_2)))))$. Symmetrically, we use $\theta_1 S_d \theta_2$ for $(\theta_2 S \theta_2) \vee (\theta_1 S (\theta_{disc} \wedge (\theta_2 \vee (\theta_1 \wedge (\theta_2 S \theta_2)))))$.

The translation $ltl\text{-}ltls$ is defined as follows [2]:

$ltl\text{-}ltls(a)$ $\qquad = a \wedge \theta_{cont}$ (where $a \in A$).

$$ltl\text{-}ltls(a') \qquad = a \wedge \theta_{disc} \text{ (where } a' \in A').$$
$$ltl\text{-}ltls(\neg\theta_1) \qquad = \neg\widehat{\theta}_1.$$
$$ltl\text{-}ltls(\theta_1 \vee \theta_2) = \widehat{\theta}_1 \vee \widehat{\theta}_2.$$
$$ltl\text{-}ltls(\theta_1 U \theta_2) \quad = (\theta_{disc} \Rightarrow (\widehat{\theta}_1 U_d \widehat{\theta}_2)) \wedge (\theta_{cont} \Rightarrow (\theta_{cont} \, U(\theta_{disc} \wedge (\widehat{\theta}_2 \vee (\widehat{\theta}_1 \wedge (\widehat{\theta}_1 U_d \widehat{\theta}_2)))))).$$
$$ltl\text{-}ltls(\theta_1 S \theta_2) \quad = (\theta_{disc} \Rightarrow (\widehat{\theta}_1 S_d \widehat{\theta}_2)) \wedge (\theta_{cont} \Rightarrow (\theta_{cont} \, S(\theta_{disc} \wedge (\widehat{\theta}_2 \vee (\widehat{\theta}_1 \wedge (\widehat{\theta}_1 S_d \widehat{\theta}_2)))))).$$

Lemma 6.1. *Let θ be an LTL$(A \cup A')$ formula. Let w be a proper word over A, and let $w' = ann(w)$. Let $\sigma \in timing(w)$ with the canonical interval representation $(a_0, I_0) \cdots (a_{2n}, I_{2n})$. Then for each $i \in \{0, \cdots, 2n\}$ and for all $t \in I_i$, we have $w', i \models \theta \iff \sigma, t \models ltl\text{-}ltls(\theta)$.*

Using the above lemma, we can now show that if $\mathcal{A}$ and θ are as in the previous steps, then $S(\mathcal{A}) = S(ltl\text{-}ltls(\theta))$. To show $S(\mathcal{A}) \subseteq S(ltl\text{-}ltls(\theta))$, let $\sigma = (a_0, I_0) \cdots (a_{2n}, I_{2n}) \in S(\mathcal{A})$. Then $w = a_0 \cdots a_{2n} \in L(\mathcal{A})$ and $\sigma \in timing(w)$. Let $w' = (ann(w))$. Then $w' \in L(\mathcal{B})$. Hence $w', 0 \models \theta$. By Lemma 6.1, we have that $\sigma, 0 \models ltl\text{-}ltls(\theta)$. Hence $\sigma \in S(ltl\text{-}ltls(\theta))$.

Conversely, suppose $\sigma \in S(ltl\text{-}ltls(\theta))$ with $\sigma = (a_0, I_0) \cdots (a_{2n}, I_{2n})$ being its canonical representation. That is $\sigma, 0 \models ltl\text{-}ltls(\theta)$. By Lemma 6.1, we have that $w', 0 \models \theta$, where $w = a_0 \cdots a_{2n}$ and $w' = ann(w)$. Hence $w' \in L(\mathcal{B})$. Hence $w \in L(\mathcal{A})$, and $\sigma \in S(\mathcal{A})$.

Step (e): A LTL$^s(A)$ formula θ can be translated to a FO$^s(A)$-formula ψ with one free variable x, such that for all $\sigma \in Sig(A)$, $\sigma, t \models \theta$ if and only if $\sigma, [t/x] \models \psi$. For a first-order formula φ, let us denote by $\varphi[z/x]$ the formula obtained by substituting all free occurrences of x in φ by z. The translation $ltl\text{-}fo$ is now given as follows:

$$ltl\text{-}fo(a) \qquad = a(x)$$
$$ltl\text{-}fo(\theta_1 U \theta_2) \quad = \exists z(x < z \wedge ltl\text{-}fo(\theta_2)[z/x] \wedge \forall y(x < y < z \Rightarrow ltl\text{-}fo(\theta_1)[y/z]))$$
$$ltl\text{-}fo(\theta_1 S \theta_2) \quad = \exists z(z < x \wedge ltl\text{-}fo(\theta_2)[z/x] \wedge \forall y(z < y < x \Rightarrow ltl\text{-}fo(\theta_1)[y/z]))$$
$$ltl\text{-}fo(\neg\theta) \qquad = \neg(ltl\text{-}fo(\theta))$$
$$ltl\text{-}fo(\theta_1 \vee \theta_2) = ltl\text{-}fo(\theta_1) \vee ltl\text{-}fo(\theta_2).$$

We can now translate θ to the FO$^s(A)$ sentence $\psi = \forall x(first(x) \Rightarrow ltl\text{-}fo(\theta))$ so that $S(\theta) = S(\psi)$.

To summarize this direction of the proof: given a counter-free ST-NFA $\mathcal{A}$ over A by steps (b) and (c), we have an LTL$(A \cup A')$ formula θ such that $ann(L(\mathcal{A})) = L(\theta)$. By step (d), we have LTL$^s(A)$ formula $\widehat{\theta}$ such that $S(\mathcal{A}) = S(\widehat{\theta})$. By step (e), we have an FO$^s(A)$ formula $\varphi_{\mathcal{A}}$ such that $S(\varphi_{\mathcal{A}}) = S(\widehat{\theta}) = S(\mathcal{A})$. This completes the proof of Theorem 6.1. □

To conclude this section, we show how we can decide whether a given MSO^s sentence is first-order definable, using our automata characterisation. The procedure is similar to the classical case where we first construct an automaton for the given MSO formula, determinize and minimize it to obtain the canonical DFA for the language, and simply check if the canonical DFA has a counter or not.

We can also define a *canonical* ST-NFA for a given ST-NFA $\mathcal{A}$. This is done as follows: First make $\mathcal{A}$ proper to get $\mathcal{A}'$; then translate via *stnfa-bnfa* to a B-NFA $\mathcal{B}$; determinize $\mathcal{B}$ to get a B-DFA $\mathcal{B}'$; Now minimize $\mathcal{B}'$ to get the canonical DFA $\mathcal{B}''$ for $\mathcal{B}'$. We note that $\mathcal{B}''$ is a B-DFA, *except* for a single non-final sink state d which represents the set of words in $A(AA)^*$ which have no extensions in $L(\mathcal{B}')$, as well as all words in $A^* - A(AA)^*$. This state can be dropped without affecting the language of $\mathcal{B}''$, and the resulting DFA is a B-DFA. We can now apply the translation *bnfa-stnfa* to $\mathcal{B}''$ to get a proper ST-DFA $\mathcal{A}''$. We note that for any two ST-NFA's that accept the same signal language, the canonical ST-DFA associated with them will be identical (up to isomorphism).

Now given an MSO^s formula φ, we can construct the corresponding ST-NFA $\mathcal{A}_{\varphi}$. Next we construct the canonical proper ST-DFA $\mathcal{A}''$ associated with $\mathcal{A}_{\varphi}$ as described above. Check if $\mathcal{A}''$ has a counter and return "Not FO^s-definable" if it does, otherwise return "FO^s-definable".

This procedure can be justified as follows: clearly if the procedure says "FO^s-definable", then the formula is indeed FO^s-definable (using Theorem 6.1). If it says "No", then suppose to the contrary that there does exist a counter-free proper ST-NFA $\mathcal{A}$ accepting the language $S(\varphi)$. Then we can see that each step in the canonicalisation preserves the absence of even-counters (including the minimization step), and hence the canonical ST-DFA will not have a counter, which is a contradiction.

7. Finite variability in FO

In this section, we show that the finite variability of an arbitrary function on the non-negative reals is expressible in FO^s. Apart from illustrating the expressiveness of FO, our aim is to give a correct sentence describing finite variability since such a definition seems to be missing in the literature.

In particular, the first-order sentence given by Hirshfeld and Rabinovich in [8] is incorrect. Their formula requires that every point t have an open interval to its right and to its left (in the latter case only when $t \neq 0$) in which the value of the function is constant. However this formula is satisfied by the function f below which is clearly *not* finitely varying:

$$f(t) = \begin{cases} a & \text{if } t = 1/n \text{ for some } n \in \mathbb{N} \\ b & \text{otherwise.} \end{cases}$$

To say that a function $f : [0, r] \to \Sigma$ is finitely varying, we need to say that it has a finite number of discontinuities in its domain. Since we can already say that a point x is a point of discontinuity of f via the first-order formula φ_{disc} of Section 4, it is sufficient if we can say that a given (bounded) subset of reals is finite.

This can be done as follows. It is sufficient to express that a set is infinite. We can argue using standard results in real analysis, that a subset of reals W is infinite iff it has a strictly decreasing infinite subsequence or a strictly increasing infinite subsequence. This is because we can first construct an infinite sequence $b_0, b_1, \ldots$ of distinct elements in W as follows. Pick any element in W and set it as b_0. Since W is infinite $W - \{b_0\}$ is also infinite, so we can pick another element $b_1 \in W$; and so on. The sequence $\langle b_i \rangle$ constructed this way is clearly a infinite sequence of distinct elements in W. Now every infinite sequence of distinct elements must have an infinite strictly monotonic subsequence. To see this, suppose $b_0, b_1, \ldots$ was the given infinite sequence. Let b_n be called a "peak point" if all elements in the sequence after it are strictly less than it (i.e. $b_i < b_n$ for all $i > n$). If there are infinitely many peak points, then we clearly have a strictly decreasing infinite subsequence. On the other hand if there were only finitely many peak points, let b_n be the last peak point. Then consider b_{n+1}: since it is not a peak point it must have a value b_i strictly greater than it, for some $i > n + 1$. Similarly b_i must also have a point strictly greater than it which occurs later in the sequence. Continuing in this way we have a strictly increasing infinite subsequence.

Further, to express this property in first-order logic, it is useful to observe that by the Bolzano–Weierstrass theorem, every bounded monotonic sequence converges to a limit point. Thus, for a bounded subset of reals W, the formula $decseq(W)$ below asserts that W contains an infinite strictly decreasing sequence:

$$decseq(W) = \exists l \exists a_0 (a_0 \in W \ \wedge l < a_0 \ \wedge \forall x ((x \in W \wedge l < x) \Rightarrow \exists y (y \in W \wedge l < y \wedge y < x))).$$

Similarly the formula $incseq(W)$ asserts that W contains an infinite strictly increasing sequence:

$$incseq(W) = \exists l \exists a_0 (a_0 \in W \wedge a_0 < l \ \wedge \forall x ((x \in W \wedge x < l) \Rightarrow \exists y (y \in W \wedge x < y \wedge y < l))).$$

Using the observations above, we can see that the formula $inf(W)$ below asserts, for bounded subsets of reals W, that W is infinite:

$$inf(W) = decseq(W) \vee incseq(W).$$

Finally the required sentence asserting that a given function is finitely varying is obtained by replacing each atomic formula of the form $x \in W$ by $\varphi_{disc}(x)$, in the formula $\neg inf(W)$.

In conclusion, we have shown that the theory of automata and logics over signals is in close analogy to the classical theory of such formalisms over words. Among the issues that remain to be addressed is the existence of a "canonical" minimum ST-DFA for the class of ST-NFA-definable signal languages. While we have identified a canonical ST-DFA for any given ST-NFA, this was done for the purpose of deciding first-order definability of signal languages. It remains to be investigated whether this is also "the minimum" ST-DFA for a given ST-NFA-definable signal language.

Acknowledgment

References

[1] Rajeev Alur, Costas Courcoubetis, Nicolas Halbwachs, Thomas A. Henzinger, Pei-Hsin Ho, Xavier Nicollin, Alfredo Olivero, Joseph Sifakis, Sergio Yovine, The algorithmic analysis of hybrid systems, Theoret. Comput. Sci. 138 (1) (1995) 3–34.
[2] Rajeev Alur, David L. Dill, A theory of timed automata, Theoret. Comput. Sci. 126 (2) (1994) 183–235.
[3] Rajeev Alur, Tomás Feder, Thomas A. Henzinger, The benefits of relaxing punctuality, J. ACM 43 (1) (1996) 116–146.
[4] E. Asarin, O. Bournez, T. Dang, O. Maler, A. Pnueli, Effective synthesis of switching controllers for linear systems, Proc. IEEE 88 (7) (2000) 1011–1025.
[5] J.R. Büchi, On a decision method in restricted second-order arithmetic., in: Z. Math. Logik Grundlag. Math., 1960, pp. 66–92.
[6] Fabrice Chevalier, Deepak D'Souza, Pavithra Prabakhar, On continuous timed automata with input-determined guards, in: Naveen Garg, S. Arun-Kumar (Eds.), FSTTCS, Kolkata, India, 2006, in: Lecture Notes in Computer Science, vol. 4337, Springer, 2006, pp. 369–380.
[7] Fabrice Chevalier, Deepak D'Souza, Pavithra Prabhakar, Counter-free input-determined timed automata, in: Jean-François Raskin, P.S. Thiagarajan (Eds.), FORMATS, in: Lecture Notes in Computer Science, vol. 4763, Springer, 2007, pp. 82–97.
[8] Yoram Hirshfeld, Alexander Moshe Rabinovich, Timer formulas and decidable metric temporal logic, Inf. Comput. 198 (2) (2005) 148–178.
[9] Ian Hodkinson, Temporal logics and automata, in: D.M. Gabbay, M. Finger, M.A. Reynolds (Eds.), Temporal Logic: Mathematical Foundations and Computational Aspects, 2, Clarendon Press, 2000, pp. 30–72.
[10] Johan Anthony Willem Kamp, Tense logic and the theory of linear order, Ph.D. Thesis, University of California, Los Angeles, California, 1968.
[11] Ron Koymans, Specifying real-time properties with metric temporal logic, Real-Time Syst. 2 (4) (1990) 255–299.
[12] Robert McNaughton, Seymour Papert, Counter-Free Automata, MIT Press, Cambridge, MA, 1971.
[13] M.O. Rabin, Decidability of second order theories and automata on infinite trees, Trans. American Math. Monthly 141 (1969) 1–35.
[14] Alexander Rabinovich, Star free expressions over the reals, Theoret. Comput. Sci. 233 (1–2) (2000) 233–235.
[15] Alexander Rabinovich, Finite variability interpretation of monadic logic of order, Theoret. Comput. Sci. 275 (1–2) (2002) 111–125.
[16] Alexander Rabinovich, Boris A. Trakhtenbrot, From finite automata toward hybrid systems (extended abstract), in: Bogdan S. Chlebus, Ludwik Czaja (Eds.), FCT, in: Lecture Notes in Computer Science, vol. 1279, Springer, 1997, pp. 411–422.
[17] M.P. Schutzenberger, On finite monoids having only trivial subgroups, Inform. Control 8 (1965) 190–194.
[18] M. Shelah, The monadic theory of order, Ann. Math. 102 (1975) 379–419.
[19] Wolfgang Thomas, Automata on infinite objects, in: Handbook of Theoretical Computer Science, Volume B: Formal Models and Semantics, Elsevier, 1990, pp. 133–192.
[20] Thomas Wilke, Classifying discrete temporal properties, in: Christoph Meinel, Sophie Tison (Eds.), STACS, in: Lecture Notes in Computer Science, vol. 1563, Springer, 1999, pp. 32–46.