# Chapter 1

# Automata and logics over signals

Fabrice Chevalier
*Laboratory for Specification and Verification,*
*ENS de Cachan, France.*

Deepak D'Souza
*Department of Computer Science and Automation,*
*Indian Institute of Science, Bangalore, India.*

M. Raj Mohan
*Department of Computer Science and Automation,*
*Indian Institute of Science, Bangalore, India.*

Pavithra Prabhakar
*Department of Computer Science,*
*University of Illinois at Urbana-Champaign, USA.*

We extend some of the classical connections between automata and logic due to Büchi [1] and McNaughton and Papert [2], 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 [3]. We also identify a "counter-free" subclass of ST-NFA's which characterize 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) [4, 5].

## 1.1. 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 [1]. Kamp [6] 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 [2] 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 [7] 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 [8–10]. For example, non-zeno timed words [8] are special kinds of signals, as are the piece-wise constant behaviours of [10].

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 [3], and called here $\mathrm{MSO}^s$, 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 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 [3] 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 $\mathrm{FO}^s$ of $\mathrm{MSO}^s$. 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 afore-mentioned 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 $\theta$ interpreted over word models, into one interpreted over signals which accepts precisely the timings of the word models of $\theta$. As in the classical case for words, we show that this characterisation allows us to decide whether a given $\mathrm{MSO}^s$ sentence is $\mathrm{FO}^s$-definable.

As a minor by-product we re-prove the expressive completeness of LTL interpreted over signals (i.e. its expressiveness coincides with $\mathrm{FO}^s$ over signals). This result also follows from Kamp's result showing the expressive completeness of LTL over arbitrary functions over the reals [6]. Nonetheless, our proof gives a more ac-

cessible proof of this result, since it uses only Kamp's result for classical words, for which there are simpler proofs in the literature (see [11, 12]).

Turning now to more details on related work, as already mentioned this paper builds on the classical results due to Büchi [1], Schutzenberger [7], McNaughton and Papert [2], and Kamp [6] for word languages. The work of Rabinovich contains many relevant results on signal languages. Rabinovich and Trakhtenbrot [13] introduce automata similar to ST-NFA's called signal acceptors. In [3] Rabinovich shows how to translate an MSO sentence $\varphi$ to a $\text{MSO}^s$ sentence that accepts precisely the timings of $\varphi$, and vice versa. This leads to a proof of the claim in [13] that signal languages definable by signal acceptors and $\text{MSO}^s$ coincide. In contrast our equivalence of ST-NFA's and $\text{MSO}^s$ uses an automata-theoretic argument similar to the proof of Büchi's result (see [14] and Chapter **??** in this volume), and helps us identify the counter-free fragment. In [15] Rabinovich also shows a star-free regular expression characterisation of $\text{FO}^s$-definable signal languages, along the lines of McNaughton and Papert [2].

Though we are mainly concerned with expressiveness in this work, there are a number of related decidability results in the literature. Rabin [16] shows that MSO over reals, with second-order quantification over subsets which are essentially countable unions of closed sets, is decidable. The decidability of $\text{MSO}^s$ follows from this result. Shelah [17] 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 [4, 5] 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 [18, 19], in their continuous semantics.

### 1.2. 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. 1.1 shows a signal $\sigma$ over the alphabet $\{a, b, c\}$

defined on $[0, 4]$ as follows:

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

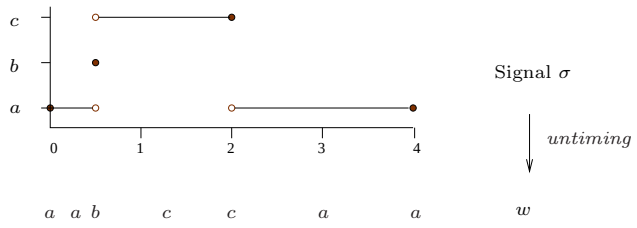



Fig. 1.1. A signal and its untiming

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 $i$, $\sigma$ is constant and equal to $a_i$ in the interval $I_i$. We can obtain a *canonical* interval representation for $\sigma$ by putting each point of discontinuity in a singular interval by itself. Thus the above interval representation for $\sigma$ is 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. 1.1 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 $\sigma$, then we define $\mathit{untiming}(\sigma)$ 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. 1.1 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 $\mathit{Prop}(A)$.

A canonical word $w$ can be "timed" to get a signal in a natural way: thus a signal $\sigma$ is in $\mathit{timing}(w)$ if $\mathit{untiming}(\sigma) = w$. 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 $r \in \mathbb{R}_{\geq 0}$.

### 1.3. 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 out-going transition of the form $(p, a, q)$ in $\delta$.

A *state-transition-labeled* 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 $i \in \{0, \cdots, n\}$, 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. 1.2 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.

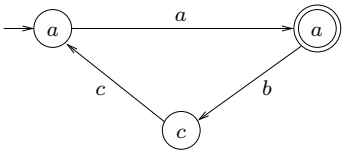



Fig. 1.2. Example ST-NFA.

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 $\mathit{timing}(L(\mathcal{A}))$, and will be denoted by $S(\mathcal{A})$. The signal shown in Fig. 1.1 is accepted by the ST-NFA shown in Fig. 1.2.

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 the sequel.

**Proposition 1.1.** *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 Figure 1.3 below.

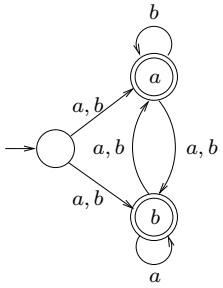



Fig. 1.3. ST-DFA $\mathcal{A}_{Prop}$ accepting the word language $Prop(\{a, b\})$.

□

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 $\mathcal{A} = (Q, S, \rightarrow, F, l)$ 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. 1.2 is proper. Clearly for a proper ST-NFA $\mathcal{A}$ over $A$ we have $L(\mathcal{A}) \subseteq Prop(A)$.

**Lemma 1.1.** *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. $S(\mathcal{A}') = S(\mathcal{A})$).*

***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 $\sigma$ 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 $q \in Q$. Figure 1.4 shows the translation applied to the example ST-NFA of Figure 1.2.

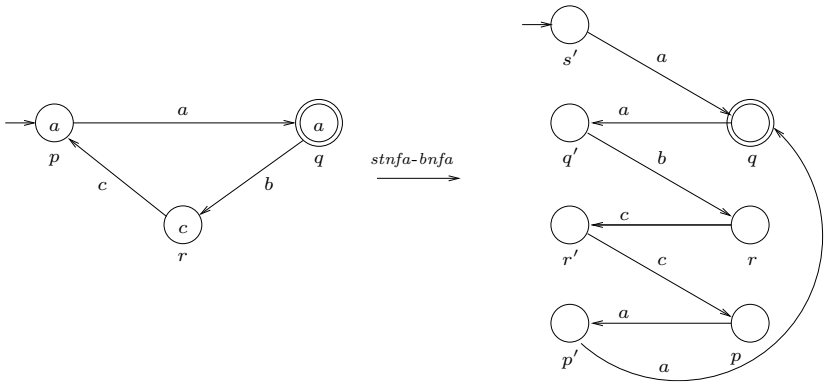



Fig. 1.4. The translation *stnfa-bnfa*.

.

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 $a$ in $\mathcal{A}$ if there is a transition $(q, a, r)$ in $\mathcal{B}$. There is also a transition $(s, a, e_2)$ in $\mathcal{A}$ if there is a transition $(p, a, q)$ out of an initial state of $\mathcal{B}$ and $e_2$ is a transition out of $q$ in $\mathcal{B}$. Finally we also have a transition from $(p, a, q)$ to $f$ labelled $b$, if $(q, b, r)$ is a transition in $\mathcal{B}$. It is not difficult to see that the word languages of $\mathcal{B}$ and $\mathcal{A}$ are the same. Figure 1.5 shows the translation *bnfa-stnfa* applied to the B-NFA on the right.

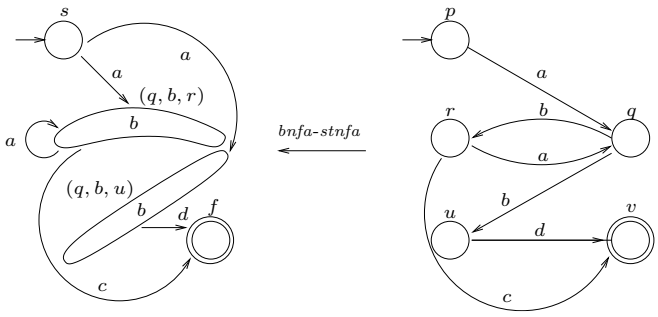



Fig. 1.5. The translation *bnfa-stnfa*.

.

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 1.2.** *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 $\mathcal{B}$, we can apply the standard subset construction to determinize it to get a language-equivalent DFA $\mathcal{B}'$. The subset construction preserves the bipartite structure of $\mathcal{B}$, and hence $\mathcal{B}'$ 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 $\mathcal{B}$ we can first determinize it to get a language-equivalent B-DFA $\mathcal{B}'$. Let the partition on states of $\mathcal{B}'$ be $Q_1$

and $Q_2$. We can now "complete" $\mathcal{B}'$ 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 $\mathcal{B}'$. The resulting automaton is a B-DFA which accepts the complement (with respect to $A(AA)^*$) of the language accepted by $\mathcal{B}$. □

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 1.2.** *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 $\mathcal{A}$ be an ST-NFA over $A$. We can determinize $\mathcal{A}$ as follows: We first go over to the word language equivalent B-NFA $\mathcal{B}$ by applying the translation *stnfa-bnfa* to $\mathcal{A}$. We now determinize $\mathcal{B}$ to get a B-DFA $\mathcal{B}'$. Finally we apply the translation *bnfa-stnfa* to $\mathcal{B}'$ to obtain an ST-NFA $\mathcal{A}'$. In fact $\mathcal{A}'$ 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 $\mathcal{A}'$, 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 $\mathcal{B}'$ was deterministic. But then we have transitions $(q, a, p_1)$ $(q, a, p_2)$ in $\mathcal{B}'$ with $p_1 \neq p_2$, which contradicts the fact that $\mathcal{B}'$ is a B-DFA.

For the second part, closure under union is immediate. For closure under complementation, let $\mathcal{A}$ be an ST-NFA over $A$. We first make $\mathcal{A}$ proper, to get a signal-equivalent proper ST-NFA $\mathcal{A}'$. Then $Sig(A) - S(\mathcal{A}) = Sig(A) - S(\mathcal{A}') = timing(Prop(A) - L(\mathcal{A}'))$ (using Proposition 1.1) $= timing(L(\mathcal{A}''))$ for some ST-NFA $\mathcal{A}''$ (using closure properties of ST-NFA-definable word languages) $= S(\mathcal{A}'')$. The closure under intersection follows from that of union and complementation. □

### 1.4. 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 $\varphi$ of the logic over a signal $\sigma$ in $Sig(A)$, along with an interpretation $\mathbb{I}$ with respect to $\sigma$, 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 $\sigma$ to the variables in $\varphi$, the satisfaction relation $\sigma, \mathbb{I} \models \varphi$, is defined inductively as:

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

For a sentence $\varphi$ (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 $\varphi$ to be $S(\varphi) = \{\sigma \in Sig(A) \mid \sigma \models \varphi\}$.

As an example, the formula

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

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 1.1.** *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 characterization of classical automata [1] (see [14]).

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 $\varphi$ 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 $\sigma$ and an interpretation $\mathbb{I}$ as a signal $\sigma_{\mathbb{I}}^{U,V} : [0, dur(\sigma)] \to A \times \{0,1\}^{m+n}$ 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 $\varphi$ with free variables in $(U, V)$ we have the notion of a $(U, V)$-model of $\varphi$.

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 1.3.** *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. Figure 1.6 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 $\mathcal{A}_{valid}^{U,V}$.

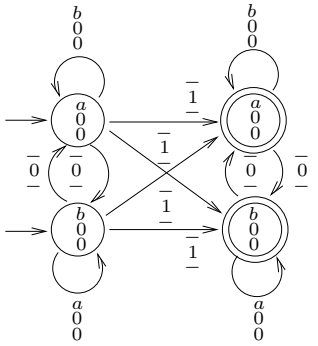


Fig. 1.6. ST-NFA accepting $x$-valid $(\{x, y\}, \emptyset)$-models over the alphabet $\{a, b\}$

.

□

**Proposition 1.4.** *Let $\varphi$ be an* 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 $\varphi$. 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 $\varphi$.*

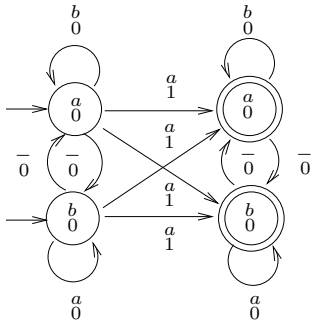
***Proof.*** Let us consider the case when $U' = U \cup \{x\}$ and $V' = V$. Consider ST-NFA $\mathcal{A}''$ which has the same set of states as $\mathcal{A}$, with the same initial and final states. The labeling of states in $\mathcal{A}''$ is same as that of $\mathcal{A}$ except that they are extended with a 0 corresponding to $x$. For every transition in $\mathcal{A}$, there are two transitions in $\mathcal{A}''$ with the same start and target states, and the label extended with 0 in one and 1 in the other. The automaton $\mathcal{A}'$ is then obtained by taking the intersection of $\mathcal{A}''$ with $\mathcal{A}^{U',V}_{valid}$.

For the case when $U' = U$ and $V' = V \cup \{X\}$, we construct $\mathcal{A}''$ similar to the above case, except that in addition now we replace each state in $\mathcal{A}$ by two states, one labeled with an extension of the original label with a 1 corresponding to $X$, and the other labeled with an extension by 0 corresponding to $X$. The new states inherit the incoming and outgoing (extended) transitions from the corresponding original state. In this case we can avoid the intersection with $\mathcal{A}^{U,V'}_{valid}$. This construction easily extends to any $U'$ and $V'$. □

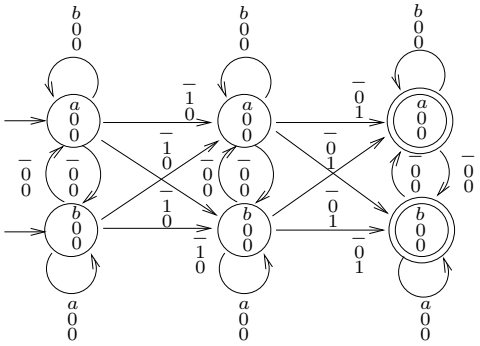
**Lemma 1.3.** *Let $\varphi$ be an* MSO$^s(A)$ *formula and let $(U, V)$ be the set of free variables in it. Then we can construct an* ST-NFA *$\mathcal{A}^{U,V}_{\varphi}$ which accepts precisely the $(U, V)$-models of $\varphi$.*

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

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




(2) $\varphi = x < y$: The automaton $\mathcal{A}^{\{x,y\},\emptyset}_{\varphi}$ (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 $\psi$, where $(U, V)$ is the set of free variables in $\psi$. Then $\mathcal{A}_{\varphi}^{U,V}$ is the intersection of $\mathcal{A}_{valid}^{U,V}$ with the ST-NFA that recognizes the complement of the signal language of $\mathcal{A}_{\psi}^{U,V}$ (cf. Lemma 1.2).

(5) $\varphi = \psi \vee \nu$: Let $\mathcal{A}_{\psi}^{U_1,V_1}$ be the ST-NFA for $\psi$, where $(U_1, V_1)$ is the set of free variables in $\psi$, and let $\mathcal{A}_{\nu}^{U_2,V_2}$ be the ST-NFA for $\nu$, where $(U_2, V_2)$ is the set of free variables in $\nu$. Let $U = U_1 \cup U_2$ and $V = (V_1 \cup V_2)$. By Prop. 1.4 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 $\psi$, so that $U = U' - \{x\}$ and $V = V'$. Let $\mathcal{A}_{\psi}^{U',V'}$ be an ST-NFA for $\psi$. 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 $\psi$, so that $U = U'$ and $V = V' - \{X\}$. Let $\mathcal{A}_{\psi}^{U',V'}$ be an ST-NFA for $\psi$. 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 $\varphi \in \mathrm{MSO}^s(A)$ 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 1.1. Let $\mathcal{A} = (Q, S, \to, F, l)$ 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 $\{e_i = p_i \xrightarrow{a_i} q_i \mid i = 1, \ldots, m\}$ 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 $\mathit{consec}(x, y, X)$ to mean that $x$ and $y$ are "consecutive" points in the set $X$, and define it to be:

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

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 $0, \ldots, m$.

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

This completes the proof of Theorem 1.1.

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 1.1.** *The class of signal languages definable in* MSO$^s(A)$ *and weak* MSO$^s(A)$ *coincide.*

### 1.5. 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 characterize 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 labeled $u$ 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 $q_0 \xrightarrow{a_0} q_1 \xrightarrow{a_1} \cdots q_n \xrightarrow{a_n} q_{n+1}$ is $a_0 l(q_1) a_1 \cdots l(q_n) a_n l(q_{n+1})$ (we ignore the label of the first state in the path, but count the label of the last state in the path). 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 labeled $u$ 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 proper-ness 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. Figure 1.7 below shows a non-proper counter-free ST-NFA and its equivalent proper version which contains a counter.

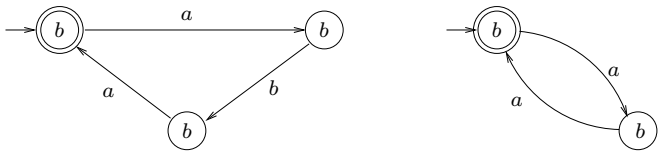



Fig. 1.7. Example showing how proper conversion may introduce counters.

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 1.4.** *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 $\mathcal{C}$.*

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

Let $S_0$, $S_1$, $\cdots$, $S_{n-1}$ 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 $q_0, q_1, \cdots$ 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 $q_0, q_n, q_{2n} \cdots$ 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

$q_{nl}, \cdots, q_{nk}$ contains a counter on $w$ and we are done.

Suppose this is not the case, i.e. for all $i \in \{nl, \cdots, nk\}$ $q_i$ is equal to say $p$. Then from the construction of the sequence one can easily see that for all $i \in \{0, \cdots, n-1\}$ $p$ 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 $S_i$.

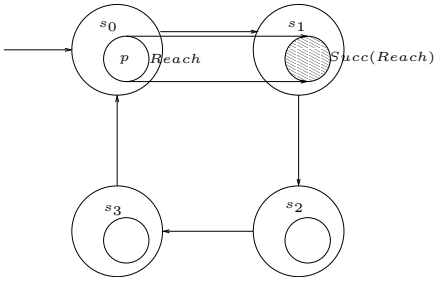



Fig. 1.8. Example depicting the set *Reach* and $Succ(Reach)$.

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 $X_0, \cdots, X_{n-1}$ 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 1.5.** *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 1.3 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 1.5.** *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 1.4 $\mathcal{A}'$ 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 1.2, the resulting automaton is a B-DFA. Furthermore, by lemma 1.4, it does not contain an even-counter. Once again, we "complete" the automaton as in the proof of lemma 1.2. 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 $A(AA)^*$. □

Using the preceding lemmas we can now argue that:

**Lemma 1.6.** *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 1.1, we can convince ourselves that $F_1 \cup F_2 = \mathit{timing}(L_1 \cup L_2)$, $F_1 \cap F_2 = \mathit{timing}(L_1 \cap L_2)$, and $\mathit{Sig}(A) - F_1 = \mathit{timing}(\mathit{Prop}(A) - L_1) = \mathit{timing}((A(AA)^* - L_1) \cap \mathit{Prop}(A))$.

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

### 1.6. Counter-free characterisation of FO signal languages

In this section our aim is to show that $\mathrm{FO}^s$-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 $\mathrm{LTL}(A)$ 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 the 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 [6]. Here we restrict the models to finitely-varying functions in $Sig(A)$, and we denote this logic by $\mathrm{LTL}^s(A)$. Given a signal $\sigma \in Sig(A)$, $t \in [0, dur(\sigma)]$ and $\theta \in \mathrm{LTL}^s(A)$, the satisfaction relation $\sigma, t \models \theta$ is defined as follows:

$\sigma, t \models a$ iff $\sigma(t) = a$.
$\sigma, t \models \theta U \eta$ iff $\exists t' : t < t' \leq dur(\sigma)$, $\sigma, t' \models \eta$, and $\forall t'' : t < t'' < t', \sigma, t'' \models \theta$.
$\sigma, t \models \theta S \eta$ iff $\exists t' : 0 \leq t' < t$, $\sigma, t' \models \eta, \forall t'' : t' < t'' < t$, $\sigma, t'' \models \theta$.

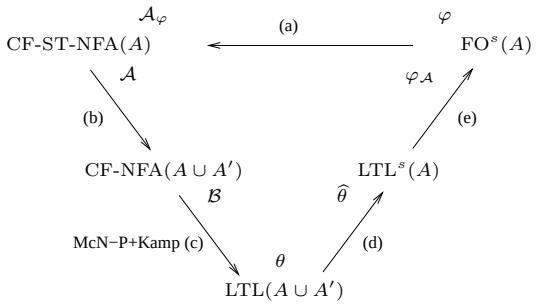
The boolean operators are interpreted in the expected way. We set $S(\theta) = \{\sigma \in Sig(A) \mid \sigma, 0 \models \theta\}$.

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

**Theorem 1.2.** *Let $A$ be a alphabet, and let $F$ be a signal language over $A$. Then the following statements are equivalent:*

*(1) $F$ is definable by an $\mathrm{FO}^s(A)$-sentence.*
*(2) $F$ is definable by a counter-free proper* ST-NFA *over $A$.*
*(3) $F$ is definable by an $\mathrm{LTL}^s(A)$-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 1.4 for Theorem 1.1 produces a counter-free proper ST-NFA at each step. This is true for the base cases $Q_a(x)$ and $x < y$. For the boolean operators, it follows by the closure properties of counter-free signal languages (cf. Lemma 1.6).

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 $\psi$. 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 a 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 1.3. 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 $\mathrm{FO}^s(A)$-sentence $\varphi_\mathcal{A}$.

**Step (b):** Let us denote by $A'$ the alphabet $\{a' \mid a \in A\}$. For a proper word $w = a_0 \dots a_{2n} \in \mathit{Prop}(A)$, we define $\mathit{ann}(w)$ to be the word $a'_0 a_1 a'_2 \dots a_{2n-1} a'_{2n}$ in $(A \cup A')^*$, and apply it to work on subsets of $\mathit{Prop}(A)$ 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 1.6, 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 $\mathit{ann}(L(A))$. Let us call this NFA over $(A \cup A')$ as $\mathcal{B}$. Thus $L(\mathcal{B}) = \mathit{ann}(L(\mathcal{A}))$ and is a classical counter-free word language.

**Step (c):** By the results due to Schutzenberger [7], McNaughton-Papert [2], and Kamp [6] 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 $\mathrm{LTL}(A \cup A')$ formula $\theta$ which defines the same word language as $\mathcal{B}$. Thus $L(\mathcal{B}) = L(\theta)$.

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

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 $\mathit{ltl\text{-}ltls}$ is defined as follows (we use $\widehat{\eta}$ for $\mathit{ltl\text{-}ltls}(\eta)$ in some places for brevity):

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

**Lemma 1.7.** *Let $\theta$ 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 $w', i \models \theta \iff \sigma, t \models ltl\text{-}ltls(\theta)$.*

***Proof.*** We do the proof by induction on $\theta$. For the base case suppose $\theta = a$ with $a \in A$. Then $w', i \models a$ iff $i$ is odd and $w'(i) = a$. This is true iff $t$ is a point of continuity in $\sigma$ and $\sigma(t) = a$. In turn this is true iff $\sigma, t \models a \wedge \theta_{cont}$.

The other base case and induction step for boolean operators are similar. We now show the induction step for $\theta = \theta_1 U \theta_2$ and omit the similar case of $\theta = \theta_1 S \theta_2$.

**Left to right implication.** Let $t \in I_i$ and suppose $w', i \models \theta_1 U \theta_2$. Then $\exists j > i$ such that $w', j \models \theta_2$ and $\forall i < i' < j \;\; w', i' \models \theta_1$. We note that by induction hypothesis we have that $\forall t'' \in I_j \;\; \sigma, t'' \models \widehat{\theta_2}$ and $\forall t'' \in \cup_{\{i' | i < i' < j\}} I_{i'} \;\; \sigma, t'' \models \widehat{\theta_1}$.

We distinguish two cases:

- $i$ is even: then $\sigma, t \models \theta_{disc}$ and we have to show that $\sigma, t \models \widehat{\theta_1} U_d \widehat{\theta_2}$.
  If $j = i + 1$ then $\sigma, t \models \widehat{\theta_2} U \widehat{\theta_2}$. Otherwise, let $k$ be the greatest even integer smaller or equal to $j$ (this is the index corresponding to the last point of discontinuity before $j$). Let $t_k > t$ such that $I_k = \{t_k\}$, note that $\sigma, t_k \models \theta_{disc}$ and $\forall t < t'' < t_k \;\; \sigma, t'' \models \widehat{\theta_1}$. If $k = j$ then $\sigma, t_k \models \widehat{\theta_2}$. Otherwise $k = j - 1$ and $\sigma, t_k \models \widehat{\theta_1} \wedge (\widehat{\theta_2} U \widehat{\theta_2})$. We have thus shown that in both cases $\sigma, t \models \widehat{\theta_1} U_d \widehat{\theta_2}$.
- $i$ is odd: then $\sigma, t \models \theta_{cont}$ and we have to show that $\sigma, t \models \theta_{cont}\, U\big(\theta_{disc} \wedge (\widehat{\theta_2} \vee (\widehat{\theta_1} \wedge (\widehat{\theta_1} U_d \widehat{\theta_2})))\big)$.
  Let $t_{i+1} > t$ be such that $I_{i+1} = \{t_{i+1}\}$. If $j = i+1$ then $\sigma, t_{i+1} \models \widehat{\theta_2}$. Otherwise $\sigma, t_{i+1} \models \widehat{\theta_1}$ and $w', i + 1 \models \theta_1 U \theta_2$. As $i + 1$ is even, by the previous case we have that $\sigma, t_{i+1} \models \widehat{\theta_1} U_d \widehat{\theta_2}$. We have thus proved that $\sigma, t \models \theta_{cont}\, U\big(\theta_{disc} \wedge (\widehat{\theta_2} \vee (\widehat{\theta_1} \wedge (\widehat{\theta_1} U_d \widehat{\theta_2})))\big)$, $t_{i+1}$ being a witness for the outermost until.

**Right to left implication:** Let $t \in I_i$ and suppose $\sigma, t \models ltl\text{-}ltls(\theta)$. We distinguish two cases:

- $i$ is even: then $\sigma, t \models \theta_{disc}$ so $\sigma, t \models \widehat{\theta_1} U_d \widehat{\theta_2}$. If $\sigma, t \models \widehat{\theta_2} U \widehat{\theta_2}$ then $w', i+1 \models \theta_2$ so $w', i \models \theta_1 U \theta_2$.
  Otherwise $\sigma, t \models \widehat{\theta_1} U\big(\theta_{disc} \wedge (\widehat{\theta_2} \vee \widehat{\theta_1} \wedge (\widehat{\theta_2} U \widehat{\theta_2}))\big)$, so there exists $t' > t$ such that $\sigma, t' \models \theta_{disc} \wedge (\widehat{\theta_2} \vee \widehat{\theta_1} \wedge (\widehat{\theta_2} U \widehat{\theta_2}))$ and $\forall t < t'' < t' \;\; \sigma, t'' \models \widehat{\theta_1}$. As $\sigma, t' \models \theta_{disc}$ there exists $j > i$ such that $I_j = \{t'\}$. We have that $\forall i < i' < j \;\; w', i' \models \theta_1$. If $\sigma, t' \models \widehat{\theta_2}$ then $w', j \models \theta_2$. If $\sigma, t' \models \widehat{\theta_1} \wedge (\widehat{\theta_2} U \widehat{\theta_2})$ then $w', j \models \theta_1$ and $w', j + 1 \models \theta_2$. In both cases we have shown that $w', i \models \theta_1 U \theta_2$.
- $i$ is odd: let $t_{i+1}$ such that $I_{i+1} = \{t_{i+1}\}$. Necessarily $\sigma, t_{i+1} \models \widehat{\theta_2} \vee (\widehat{\theta_1} \wedge (\widehat{\theta_1} U_d \widehat{\theta_2}))$. If $\sigma, t_{i+1} \models \widehat{\theta_2}$ then $w', i + 1 \models \theta_2$ so $w', i \models \theta_1 U \theta_2$. Otherwise $\sigma, t_{i+1} \models \widehat{\theta_1} \wedge (\widehat{\theta_1} U_d \widehat{\theta_2})$ so $w', i+1 \models \theta_1$. As $i+1$ is even and $\sigma, t_{i+1} \models \widehat{\theta_1} U_d \widehat{\theta_2}$, by the previous case, we have that $w', i+1 \models \theta_1 U \theta_2$; it follows that $w', i \models \theta_1 U \theta_2$. □

Using the above lemma we can now show that if $\mathcal{A}$ and $\theta$ are as in the previous steps, then $S(\mathcal{A}) = S(\mathit{ltl\text{-}ltls}(\theta))$. To show $S(\mathcal{A}) \subseteq S(\mathit{ltl\text{-}ltls}(\theta))$, 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 $\sigma \in \mathit{timing}(w)$. Let $w' = \mathit{ann}(w)$. Then $w' \in L(\mathcal{B})$. Hence $w', 0 \models \theta$. By Lemma 1.7 we have that $\sigma, 0 \models \mathit{ltl\text{-}ltls}(\theta)$. Hence $\sigma \in S(\mathit{ltl\text{-}ltls}(\theta))$.

Conversely, suppose $\sigma \in S(\mathit{ltl\text{-}ltls}(\theta))$ with $\sigma = (a_0, I_0)\cdots(a_{2n}, I_{2n})$ being its canonical representation. That is $\sigma, 0 \models \mathit{ltl\text{-}ltls}(\theta)$. By Lemma 1.7 we have that $w', 0 \models \theta$, where $w = a_0 \cdots a_{2n}$ and $w' = \mathit{ann}(w)$. Hence $w' \in L(\mathcal{B})$. Hence $w \in L(\mathcal{A})$, and $\sigma \in S(\mathcal{A})$.

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

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

We can now translate $\theta$ to the $\mathrm{FO}^s(A)$ sentence $\psi = \forall x(\mathit{first}(x) \Rightarrow \mathit{ltl\text{-}fo}(\theta))$, 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 $\mathrm{LTL}(A \cup A')$ formula $\theta$ such that $\mathit{ann}(L(\mathcal{A})) = L(\theta)$. By step (d) we have $\mathrm{LTL}^s(A)$ formula $\widehat{\theta}$ such that $S(\mathcal{A}) = S(\widehat{\theta})$. By step (e) we have an $\mathrm{FO}^s(A)$ formula $\varphi_{\mathcal{A}}$ such that $S(\varphi_{\mathcal{A}}) = S(\widehat{\theta}) = S(\mathcal{A})$. This completes the proof of Theorem 1.2. □

To conclude this section, we show how we can decide whether a given $\mathrm{MSO}^s$ 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 (upto isomorphism).

Now given an MSO$^s$ formula $\varphi$, 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 1.2). 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 minimisation step), and hence the canonical ST-DFA will not have a counter, which is a contradiction.

### 1.7. ST-NFA's and signal regular expressions

In this section we give a regular expression formalism for describing "regular" and "counter-free" signal languages. We will essentially define the underlying word languages via regular expressions, and say that the signal languages defined are simply the timings of the underlying word languages.

A *partial signal regular expression* over an alphabet $A$ is given by the following syntax:

$$p ::= \emptyset \mid ab \mid p + p \mid p \cdot p \mid p^*$$

where $a$ and $b$ belong to $A$. A *signal regular expression* over $A$ is an expression of the form

$$r ::= a \cdot p \mid r + r,$$

where $a \in A$ and $p$ is a partial signal regular expression over $A$.

A partial signal regular expression $p$ over $A$ defines a language $L(p) \subseteq (AA)^*$ which is given inductively as follows:

- $L(\emptyset) = \emptyset$
- $L(ab) = \{ab\}$
- If $p$ and $q$ are partial signal regular expressions then $L(p + q) = L(p) \cup L(q)$, $L(p \cdot q) = L(p) \cdot L(q)$, and $L(p^*) = L(p)^*$.

A signal regular expression $r$ over $A$ defines a subset $L(r)$ of $A(AA)^*$ as follows:

- If $a \in A$ and $p$ is a partial signal regular expression then $L(a \cdot p) = a \cdot L(p)$, and
- If $r$ and $s$ are signal regular expressions then $L(r + s) = L(r) \cup L(s)$.

We define the signal language associated with a signal regular expression $r$, denoted $S(r)$, to be $\mathit{timing}(L(r))$.

We now show that the class of signal languages definable by signal regular expressions is precisely the class of signal languages definable by ST-NFA's. For that

it suffices to show that the word languages definable by ST-NFA's and signal regular expressions coincide.

Let us fix an alphabet $A$. It will be useful to consider the alphabet $\Sigma_{AA} = \{s_{ab} \mid ab \in AA\}$. We note that the homomorphism $h : \Sigma_{AA}^* \to A^*$ given by $h(s_{ab}) = ab$, is injective and onto with respect to $(AA)^*$.

**Proposition 1.6.** *The class of word languages defined by partial signal regular expressions over $A$ is precisely the class of word languages which are regular subsets of $(AA)^*$.*

***Proof.*** Languages defined by partial signal regular expressions can be seen to be regular subsets of $(AA)^*$, by a simple inductive argument. Conversely, if $L$ is a regular subset of $(AA)^*$, then $h^{-1}(L)$ is a regular subset of $\Sigma_{AA}^*$ (using a standard property of homomorphisms), and hence there exists a regular expression $r$ over $\Sigma_{AA}$ with $L(r) = h^{-1}(L)$. Now it is easy to see that if we simply replace each symbol $s_{ab}$ by $ab$ in $r$ (let us denote this partial signal regular expression by $h(r)$) then $L(h(r)) = h(L(r))$ which equals $L$. □

**Theorem 1.3.** *The class of signal languages definable by signal regular expressions over $A$ coincides with the class of signal languages definable by* ST-NFA *'s over $A$.*

***Proof.*** From the definition of signal regular expressions and the preceding Prop. 1.6, it follows that the word language defined by a signal regular expression $r$ is a finite union of languages of the form $a \cdot M$ where $M$ is a regular subset of $(AA)^*$. Thus $r$ defines a regular subset of $A(AA)^*$, and by Proposition 1.2, we have an ST-NFA over $A$ accepting $L(r)$.

In the converse direction, it suffices to show that every regular subset $L$ of $A(AA)^*$ is definable by a signal regular expression. Let $L/a$ denote the language $\{w \in A^* \mid aw \in L\}$. Then clearly $L = \bigcup_{a \in A} a \cdot (L/a)$. Further, $L/a$ is a regular language and a subset of $(AA)^*$. So by Prop. 1.6 each $L/a$ is definable by a partial signal regular expression $p_a$. Thus, the signal regular expression $r$ given by the sum of $a \cdot p_a$'s, describes $L$. □

In the rest of this section we give a characterisation of counter-free ST-NFA's in terms of star-free regular expressions.

We define a *star-free* partial signal regular expression over an alphabet $A$ as follows:

$$p' ::= \emptyset \mid ab \mid p' + p' \mid p' \circ p' \mid \tilde{p'},$$

where $a, b \in A$. The syntax of a *star-free* signal regular expression over $A$ is given by

$$r' ::= a \cdot p' \mid r' + r'$$

where $a \in A$ and $p'$ is a star-free partial signal regular expression over $A$. We define the word language generated by a star-free partial signal regular expression $p'$ similar to a partial signal regular expression, except that

- $L(p' \circ q') = (L(p') \cdot L(q')) \cap \mathit{Prop}_2(A)$,
- $L(\tilde{p'}) = \mathit{Prop}_2(A) - L(p')$,

where $\mathit{Prop}_2(A)$ is set of words in $(AA)^*$ which represent proper words – i.e. $\mathit{Prop}_2(A) = \bigcup_{a \in A} \mathit{Prop}(A)/a$. We note that $\mathit{Prop}_2(A)$ can be defined by (the homomorphic image of) the classical star-free expression $r^A_{Prop}$ which is the complement of the sum, over all $a, b \in A$, of $\bar{\emptyset} \cdot s_{aa} s_{ab} \cdot \bar{\emptyset}$.

As before, we define the signal language associated with a star-free signal regular expression $r'$ to be $S(r') = \mathit{timing}(L(r'))$.

**Lemma 1.8.** *The class of regular languages definable by* CF-ST-NFA*'s and star-free signal regular expressions over an alphabet A coincide.*

***Proof.*** Once again, we prove that the class of word languages defined by the two formalisms coincide. Let $L$ be the word language accepted by a counter-free ST-NFA. Then by the proof of Lemma 1.6 we know that $L$ is a regular subset of $\mathit{Prop}(A)$ and is even-counter free (i.e. it is accepted by an NFA with no even-counters). Once again we can write $L = \bigcup_{a \in A} a \cdot (L/a)$, and observe that each $L/a$ is a regular subset of $(AA)^*$ and is even-counter free. Now if we consider $h^{-1}(L/a)$, we can see that it must be *counter-free*, since a counter here would imply an even-counter in $L/a$. By the classical result due to Schutzenberger, we must have a star-free regular expression $p_a$ over $\Sigma_{AA}$ such that $L(p_a) = h^{-1}(L/a)$. (Recall that a star-free regular expression is built from $\emptyset$, $a$ (for $a \in A$), sum, concatenation, and complement in $A^*$). Once again, if we consider the star-free partial signal regular expression $h(p_a)$ obtained from $p_a$ by replacing each $s_{ab}$ by $ab$, then we can see that $L(h(p_a)) = h(L(p_a)) = L/a$. Hence $L$ which is the union of $a \cdot (L/a)$'s can be defined by the sum of the regular expressions $a \cdot p_a$.

Conversely let $r'$ be a star-free signal regular expression over the alphabet $A$. So $r$ is sum of expressions of the form $a \cdot p'$, where $p'$ is a star-free partial signal regular expression. Now each $p'$ defines a subset of $\mathit{Prop}_2(A)$ that is even-counter free. To see this, we observe that we can get a classical star-free regular expression from $p'$, say $h^{-1}(p')$, by replacing every $ab$ by $s_{ab}$, and the concatenation ($\circ$) and complementation ($\tilde{}$) by usual concatenation and complementation followed by intersection with the star-free expression $r^A_{Prop}$. The star-free expression $h^{-1}(p')$ is such that $h(L(h^{-1}(p')) = L(p')$. Now $h^{-1}(p')$ is star-free and hence its language has *no* counters. It follows that $L(p')$ is even-counter free. Thus $L(r)$ is a finite union of languages of the form $a \cdot M$ where $M$ is a even-counter free subset of $\mathit{Prop}_2(A)$. Hence, by characterisation of counter-free ST-NFA languages in the proof of Lemma 1.6, we have that $L(r)$ is accepted by a proper counter-free ST-NFA. □

### 1.8. 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 $\mathrm{FO}^s$. 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 [20] 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 $\varphi_{disc}$ of Sec. 1.4, 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 that 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:

$$\begin{aligned} decseq(W) = \exists l \exists a_0 ( \; & a_0 \in W \; \wedge l < a_0 \wedge \\ & \forall x ((x \in W \wedge l < x) \Rightarrow \exists y (y \in W \wedge l < y \wedge y < x))). \end{aligned}$$

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 $\neg inf(W)$.

## 1.9. Conclusion

In this chapter we have shown that the theory of automata and logics over signals bears a 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, along the lines of the Myhill-Nerode Theorem for regular word 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 investigate a suitable definition of a "minimum" ST-DFA for a given ST-NFA-definable signal language.

## Acknowledgments

We thank Kamal Lodaya and Wolfgang Thomas for several comments and suggestions which have improved the content and presentation considerably. The authors also acknowledge support from the Indo-French TIMED-DISCOVERI project.

Part of the material in this chapter is reproduced with permission from the following source:

F. Chevalier, D. D'Souza, M. Raj Mohan, P. Prabhakar, Automata and logics over finitely varying functions, Annals of Pure and Applied Logic (2009), doi:10.1016/j.apal.2009.07.007.

## References

[1] J. Büchi. On a decision method in restricted second-order arithmetic. In *Z. Math. Logik Grundlag. Math.*, pp. 66–92, (1960).

[2] R. McNaughton and S. Papert, *Counter-Free Automata*. (MIT Press, Cambridge, MA, 1971).

[3] A. Rabinovich, Finite variability interpretation of monadic logic of order., *Theor. Comput. Sci.* **275**(1-2), 111–125, (2002).
[4] F. Chevalier, D. D'Souza, and P. Prabakhar. On continuous timed automata with input-determined guards. In eds. N. Garg and S. Arun-Kumar, *FSTTCS*, vol. 4337, *Lecture Notes in Computer Science*, pp. 369–380, Kolkata, India (Dec., 2006). Springer.
[5] F. Chevalier, D. D'Souza, and P. Prabhakar. Counter-free input-determined timed automata. In eds. J.-F. Raskin and P. S. Thiagarajan, *FORMATS*, vol. 4763, *Lecture Notes in Computer Science*, pp. 82–97. Springer, (2007). ISBN 978-3-540-75453-4.
[6] J. A. W. Kamp, *Tense Logic and the Theory of Linear Order*. (PhD thesis, University of California, Los Angeles, California, 1968).
[7] M. P. Schtzenberger. On finite monoids having only trivial subgroups. In *Information and Control*, pp. 8:190–194, (1965).
[8] R. Alur and D. L. Dill, A theory of timed automata, *Theoretical Computer Science*. **126**(2), 183–235, (1994).
[9] R. Alur, C. Courcoubetis, N. Halbwachs, T. A. Henzinger, P.-H. Ho, X. Nicollin, A. Olivero, J. Sifakis, and S. Yovine, The algorithmic analysis of hybrid systems, *Theor. Comput. Sci.* **138**(1), 3–34, (1995).
[10] E. Asarin, O. Bournez, T. Dang, O. Maler, and A. Pnueli, Effective synthesis of switching controllers for linear systems, *Proceedings of the IEEE*. **88**(7), 1011–1025, (2000).
[11] T. Wilke. Classifying discrete temporal properties. In eds. C. Meinel and S. Tison, *STACS*, vol. 1563, *Lecture Notes in Computer Science*, pp. 32–46. Springer, (1999).
[12] I. Hodkinson, *Temporal Logics and Automata*, In eds. D. M. Gabbay, M. Finger, and M. A. Reynolds, *Temporal Logic: Mathematical Foundations and Computational Aspects*, vol. 2, pp. 30–72. Clarendon Press, (2000).
[13] A. Rabinovich and B. A. Trakhtenbrot. From finite automata toward hybrid systems (extended abstract). In eds. B. S. Chlebus and L. Czaja, *FCT*, vol. 1279, *Lecture Notes in Computer Science*, pp. 411–422. Springer, (1997). ISBN 3-540-63386-3.
[14] W. Thomas. Automata on infinite objects. In *Handbook of Theoretical Computer Science, Volume B: Formal Models and Semantics*, pp. 133–192. Elsevier, (1990).
[15] A. Rabinovich, Star free expressions over the reals., *Theor. Comput. Sci.* **233**(1-2), 233–235, (2000).
[16] M. O. Rabin, Decidability of second order theories and automata on infinite trees, *Trans. American Math. Monthly*. **141**, 1–35, (1969).
[17] M. Shelah, The monadic theory of order, *Annals of Mathematics*. **102**, 379–419, (1975).
[18] R. Koymans, Specifying Real-Time Properties with Metric Temporal Logic., *Real-Time Systems*. **2**(4), 255–299, (1990).
[19] R. Alur, T. Feder, and T. A. Henzinger, The Benefits of Relaxing Punctuality, *Journal of the ACM*. **43**(1), 116–146, (1996).
[20] Y. Hirshfeld and A. M. Rabinovich, Timer formulas and decidable metric temporal logic, *Inf. Comput.* **198**(2), 148–178, (2005).

# Index