Pebble weighted automata and transitive closure logics

Benedikt Bollig, Paul Gastin,
Benjamin Monmege, and Marc Zeitoun

Laboratoire Spécification & Vérification

Pebble weighted automata and transitive closure logics[*]

Benedikt Bollig, Paul Gastin, Benjamin Monmege, and Marc Zeitoun

LSV, ENS Cachan, CNRS, INRIA
61, Av. du Président Wilson, F-94235 Cachan Cedex, France
`firstname.lastname@lsv.ens-cachan.fr`

Abstract. We introduce new classes of weighted automata on words. Equipped with pebbles and a two-way mechanism, they go beyond the class of recognizable formal power series, but capture a weighted version of first-order logic with bounded transitive closure. In contrast to previous work, this logic allows for unrestricted use of universal quantification. Our main result states that pebble weighted automata, nested weighted automata, and first-order logic with bounded transitive closure are expressively equivalent. We also give new logical characterizations of the recognizable series.

1 Introduction

Connections between logical and state-based formalisms have always been a fascinating research area in theoretical computer science, which produced some fundamental theorems. The line of classical results started with the equivalence of MSO logic and finite automata [Büc60,Elg61,Tra61] and gave rise to natural generalizations to trees in terms of logics for tree automata [Rab69,C+08], tree-walking automata, and pebble tree automata [BSSS06,SS07,Boj08]. Such automata models recently attracted significant interest, for instance in the context of manipulating XML documents and evaluating XPath queries [tCS10].

Other extensions of finite automata are of quantitative nature and include timed automata, probabilistic systems, and transducers, which all come with more or less natural, specialized logical characterizations. A generic concept of adding weights to qualitative systems is provided by the theory of weighted automata [KVD09], first introduced by Schützenberger [Sch61]. The output of a weighted automaton running on a word is no longer a Boolean value discriminating between accepted and rejected behaviors. A word is rather mapped to a weight from a semiring, summing over all possible run weights, each calculated as the product of its transition outcomes. Indeed, probabilistic automata and word transducers appear as instances of that framework, which found its way into numerous application areas such as natural language processing and speech recognition or digital image compression (see [KVD09, Part IV]).

A logical characterization of weighted automata, however, was established only recently [DG07], in terms of a (restricted) weighted MSO logic capturing the recognizable formal power series (*i.e.*, the behaviors of finite weighted automata). The key idea is to interpret existential and universal quantifications as the operations sum and product from a semiring. To make this definition work, however, one has to restrict the universal first-order quantification, which, otherwise, appears to be too powerful and goes beyond the class of recognizable series. In the present paper, we follow a different approach. Instead of restricting the logic, we define an extended automata model that naturally *reflects* it. Indeed, it turns out that universal quantification is essentially captured by a pebble (two-way) mechanism in the automata-theoretic counterpart. Inspired by the theory of

---

two-way and pebble automata on words and trees [EH99,BSSS06,SS07,Boj08], we actually define weighted generalizations that preserve their natural connections with logic.

More precisely, we introduce pebble weighted automata on words and establish expressive equivalence to weighted first-order logic with bounded transitive closure and unrestricted use of quantification, extending the classical Boolean case for words [EH07]. Our equivalence proof makes a detour via another natural concept, named nested weighted automata, which resembles the nested tree-walking automata by ten Cate and Segoufin [tCS10]. The transitive closure logic also yields alternative characterizations of the (classical) recognizable formal power series.

These results are not only of theoretical interest. They also lay the basis for quantitative extensions of database query languages such as XPath, and may provide tracks to evaluate quantitative aspects of XML documents. The framework of weighted automata is natural for answering questions such as "How many nodes are selected by a request?", or "How difficult is it to answer a query?". The navigational mechanism of pebble automata is also well-suited in this context. For these reasons, our work is a first step before tackling similar questions on trees.

Outline. In Section 2, we recall the classical concept of recognizable formal power series and introduce weighted MSO logics. The new bounded transitive closure operator is defined in Section 3. The class of recognizable series is revisited in the light of transitive closure, which yields to new logical characterizations. In Section 4, we go beyond recognizable series and introduce nested weighted automata giving a translation into FO+BTC$^<$, first-order logic with bounded transitive closure. Pebble weighted automata are defined in Section 5 where we also state their expressive equivalence to nested weighted automata and FO+BTC$^<$ logic.

2 Notation and background

In this section we set up the notation and we recall some basic results on weighted automata and weighted logics. We refer the reader to [DG07,KVD09] for details.

Throughout the paper, Σ denotes a finite alphabet and Σ^* (resp. Σ^+) is the free monoid (resp. semigroup) over Σ, i.e., the set of words (resp., nonempty words). The length of $u \in \Sigma^*$ is denoted $|u|$. If $|u| = n \geq 1$, we usually write $u = u_1 \cdots u_n$ with $u_i \in \Sigma$ and we let $\mathrm{Pos}(u) = \{1, \ldots, n\}$. For $1 \leq i \leq j \leq n$, we denote by $u[i..j]$ the factor $u_i u_{i+1} \cdots u_j$ of u. Finally, we let $\Sigma^{\leq k} = \bigcup_{1 \leq i \leq k} \Sigma^i$.

Formal power series. A *semiring* is a structure $\mathbb{K} = (K, \oplus, \odot, \mathbf{0}, \mathbf{1})$ where $(K, \oplus, \mathbf{0})$ is a commutative monoid, $(K, \odot, \mathbf{1})$ is a monoid, $\odot$ distributes over $\oplus$, and $\mathbf{0}$ is an absorbing element for $\odot$. We say that $\mathbb{K}$ is *commutative* if so is $(K, \odot, \mathbf{1})$. We shall refer in the examples to the usual Boolean semiring $\mathbb{B} = (\{\mathbf{0}, \mathbf{1}\}, \vee, \wedge, \mathbf{0}, \mathbf{1})$ and to the semiring $(\mathbb{N}, +, \cdot, 0, 1)$ of natural numbers, denoted $\mathbb{N}$. A *formal power series* (or *series*, for short) is a mapping $f : \Sigma^+ \to \mathbb{K}$. The set of formal power series is denoted $\mathbb{K}\langle\langle \Sigma^+ \rangle\rangle$. We denote again by $\oplus$ and $\odot$ the pointwise addition and multiplication (called the *Hadamard product*) on $\mathbb{K}\langle\langle \Sigma^+ \rangle\rangle$, and by $\mathbf{0}$ and $\mathbf{1}$ the constant series with values $\mathbf{0}$ and $\mathbf{1}$, respectively. Then $(\mathbb{K}\langle\langle \Sigma^+ \rangle\rangle, \oplus, \odot, \mathbf{0}, \mathbf{1})$ is itself a semiring.

Weighted automata. All automata we consider are finite. A *weighted automaton* (wA) over $\mathbb{K} = (K, \oplus, \odot, \mathbf{0}, \mathbf{1})$ and Σ is a tuple $\mathcal{A} = (Q, \mu, \lambda, \gamma)$, where Q is the set of states, $\mu : \Sigma \to K^{Q \times Q}$ is the transition weight function and $\lambda, \gamma : Q \to K$ are weight functions for entering and leaving a state. The function μ gives, for each $a \in \Sigma$ and $p, q \in Q$, the weight $\mu(a)_{p,q}$ of the transition $p \xrightarrow{a} q$. It extends uniquely to a homomorphism μ from Σ^+ to $K^{Q \times Q}$ with matrix multiplication. Viewing μ as a mapping $\mu : Q \times \Sigma \times Q \to K$, we sometimes write $\mu(p, a, q)$ instead of $\mu(a)_{p,q}$. A *run* on a

word $u = u_1 \cdots u_n$ is a sequence of transitions $\rho = p_0 \xrightarrow{u_1} p_1 \xrightarrow{u_2} \cdots \xrightarrow{u_n} p_n$. The *weight* of the run ρ is

$$\text{weight}(\rho) \stackrel{\text{def}}{=} \lambda(p_0) \odot [\bigodot_{i=1}^{n} \mu(p_{i-1}, u_i, p_i)] \odot \gamma(p_n)$$

and the *weight* $[\![\mathcal{A}]\!](u)$ of u is the sum of all weights of runs on u, which can be computed as $[\![\mathcal{A}]\!](u) = \lambda \odot \mu(u) \odot \gamma$, viewing λ, μ, γ as matrices of dimension $1 \times |Q|$, $|Q| \times |Q|$, and $|Q| \times 1$, respectively.

We call $[\![\mathcal{A}]\!] \in \mathbb{K}\langle\!\langle \Sigma^+ \rangle\!\rangle$ the *behavior*, or *semantics* of $\mathcal{A}$. A formal power series $f \in \mathbb{K}\langle\!\langle \Sigma^+ \rangle\!\rangle$ is called *recognizable* if it is the behavior of some weighted automaton. We let $\mathbb{K}^{\text{rec}}\langle\!\langle \Sigma^+ \rangle\!\rangle$ be the collection of all the recognizable formal power series.

Example 1. Consider $(\mathbb{N}, +, \cdot, 0, 1)$ and let $\mathcal{A}$ be the automaton with a single state, $\mu(a) = 2$ for all $a \in \Sigma$, and $\lambda = \gamma = 1$. Then, $[\![\mathcal{A}]\!](u) = 2^{|u|}$ for all $u \in \Sigma^+$.

It is well-known that $\mathbb{K}^{\text{rec}}\langle\!\langle \Sigma^+ \rangle\!\rangle$ is stable under $\oplus$ and, if $\mathbb{K}$ is commutative, also under $\odot$, making $(\mathbb{K}^{\text{rec}}\langle\!\langle \Sigma^+ \rangle\!\rangle, \oplus, \odot, \mathbf{0}, \mathbf{1})$ a subsemiring of $(\mathbb{K}\langle\!\langle \Sigma^+ \rangle\!\rangle, \oplus, \odot, \mathbf{0}, \mathbf{1})$.

Weighted logics. We fix infinite supplies $\text{Var} = \{x, y, z, t, x_1, x_2, \ldots\}$ of first-order variables, and $\text{VAR} = \{X, Y, X_1, X_2, \ldots\}$ of second-order variables. The set of *weighted monadic second-order* formulas over $\mathbb{K}$ and Σ, denoted $\text{MSO}(\mathbb{K}, \Sigma)$ (or, simply, MSO), is given by the grammar

$$\varphi ::= k \mid P_a(x) \mid x \leq y \mid x \in X \mid \neg\varphi \mid \varphi \vee \varphi \mid \varphi \wedge \varphi \mid \exists x \varphi \mid \forall x \varphi \mid \exists X \varphi \mid \forall X \varphi$$

where $k \in K$, $a \in \Sigma$, $x, y \in \text{Var}$ and $X \in \text{VAR}$.

For $\varphi \in \text{MSO}(\mathbb{K}, \Sigma)$, let $\text{Free}(\varphi)$ denote the set of free variables of φ. If $\text{Free}(\varphi) = \emptyset$, then φ is called a *sentence*. For a finite set $\mathcal{V} \subseteq \text{Var} \cup \text{VAR}$ and a word $u \in \Sigma^+$, a $(\mathcal{V}, u)$-*assignment* is a function σ that maps a first-order variable in $\mathcal{V}$ to an element of $\text{Pos}(u)$ and a second-order variable in $\mathcal{V}$ to a subset of $\text{Pos}(u)$. For $x \in \text{Var}$ and $i \in \text{Pos}(u)$, $\sigma[x \mapsto i]$ denotes the $(\mathcal{V} \cup \{x\}, u)$-assignment that maps x to i and, otherwise, coincides with σ. For $X \in \text{VAR}$ and $I \subseteq \text{Pos}(u)$, the $(\mathcal{V} \cup \{X\}, u)$-assignment $\sigma[X \mapsto I]$ is defined similarly.

A pair (u, σ), where σ is a $(\mathcal{V}, u)$-assignment, can be encoded as a word over the extended alphabet $\Sigma_\mathcal{V} \stackrel{\text{def}}{=} \Sigma \times \{0, 1\}^\mathcal{V}$. We write a word $(u_1, \sigma_1) \cdots (u_n, \sigma_n) \in \Sigma_\mathcal{V}^+$ as (u, σ) where $u = u_1 \cdots u_n$ and $\sigma = \sigma_1 \cdots \sigma_n$. We call (u, σ) *valid* if, for each first-order variable $x \in \mathcal{V}$, the x-row of σ contains exactly one 1. If (u, σ) is valid, then σ can be considered as the $(\mathcal{V}, u)$-assignment that maps a first-order variable $x \in \mathcal{V}$ to the unique position carrying 1 in the x-row, and a second-order variable $X \in \mathcal{V}$ to the set of positions carrying 1 in the X-row.

Fix a finite set $\mathcal{V}$ of variables such that $\text{Free}(\varphi) \subseteq \mathcal{V}$. The semantics $[\![\varphi]\!]_\mathcal{V} \in \mathbb{K}\langle\!\langle \Sigma_\mathcal{V}^+ \rangle\!\rangle$ of φ wrt. $\mathcal{V}$ is given as follows: if (u, σ) is not valid, we set $[\![\varphi]\!]_\mathcal{V}(u, \sigma) = \mathbf{0}$, otherwise $[\![\varphi]\!]_\mathcal{V}$ is given in Figure 1. Hereby, the product follows the natural order on $\text{Pos}(u)$ and some fixed order on the power set of $\text{Pos}(u)$.

We simply write $[\![\varphi]\!]$ for $[\![\varphi]\!]_{\text{Free}(\varphi)}$. By $\mathbb{K}^{\text{MSO}}\langle\!\langle \Sigma^+ \rangle\!\rangle$, we denote the collection of all formal power series definable by a sentence from $\text{MSO}(\mathbb{K}, \Sigma)$.

Example 2. For $\mathbb{K} = \mathbb{B}$, recognizable and $\text{MSO}(\mathbb{K}, \Sigma)$-definable series coincide. In contrast, for $\mathbb{K} = (\mathbb{N}, +, \cdot, 0, 1)$, the very definition yields $[\![\forall x \forall y \, 2]\!](u) = 2^{|u|^2}$, which is not recognizable [DG07]. Indeed, let $\mathcal{A} = (Q, \mu, \lambda, \gamma)$. Then, $[\![\mathcal{A}]\!](u) = O((M|Q|)^{|u|+2})$ for $M = \max\{\mu(a)_{p,q}, \lambda(p), \gamma(q) \mid a \in \Sigma, p, q \in Q\}$, since there are $O(|Q|^{|u|+1})$ runs on u, each of weight $O(M^{|u|+2})$. Also observe that the behavior of the automaton of Example 1 is $[\![\forall y \, 2]\!]$. Therefore, recognizable series are not stable under universal first-order quantification.

3

$$\llbracket k \rrbracket_{\mathcal{V}}(u, \sigma) = k, \quad \text{for } k \in K.$$

$$\llbracket \varphi_1 \vee \varphi_2 \rrbracket_{\mathcal{V}}(u, \sigma) = \llbracket \varphi_1 \rrbracket_{\mathcal{V}}(u, \sigma) \oplus \llbracket \varphi_2 \rrbracket_{\mathcal{V}}(u, \sigma).$$

$$\llbracket \varphi_1 \wedge \varphi_2 \rrbracket_{\mathcal{V}}(u, \sigma) = \llbracket \varphi_1 \rrbracket_{\mathcal{V}}(u, \sigma) \odot \llbracket \varphi_2 \rrbracket_{\mathcal{V}}(u, \sigma).$$

$$\llbracket P_a(x) \rrbracket_{\mathcal{V}}(u, \sigma) = \begin{cases} \mathbf{1} & \text{if } u_{\sigma(x)} = a, \\ \mathbf{0} & \text{otherwise.} \end{cases}$$

$$\llbracket \exists x \varphi \rrbracket_{\mathcal{V}}(u, \sigma) = \bigoplus_{i \in \mathrm{Pos}(u)} \llbracket \varphi \rrbracket_{\mathcal{V} \cup \{x\}}(u, \sigma[x \mapsto i]).$$

$$\llbracket x \in X \rrbracket_{\mathcal{V}}(u, \sigma) = \begin{cases} \mathbf{1} & \text{if } \sigma(x) \in \sigma(X), \\ \mathbf{0} & \text{otherwise.} \end{cases}$$

$$\llbracket \exists X \varphi \rrbracket_{\mathcal{V}}(u, \sigma) = \bigoplus_{I \subseteq \mathrm{Pos}(u)} \llbracket \varphi \rrbracket_{\mathcal{V} \cup \{X\}}(u, \sigma[X \mapsto I]).$$

$$\llbracket x \leq y \rrbracket_{\mathcal{V}}(u, \sigma) = \begin{cases} \mathbf{1} & \text{if } \sigma(x) \leq \sigma(y), \\ \mathbf{0} & \text{otherwise.} \end{cases}$$

$$\llbracket \forall x \varphi \rrbracket_{\mathcal{V}}(u, \sigma) = \bigodot_{i \in \mathrm{Pos}(u)} \llbracket \varphi \rrbracket_{\mathcal{V} \cup \{x\}}(u, \sigma[x \mapsto i]).$$

$$\llbracket \neg \varphi \rrbracket_{\mathcal{V}}(u, \sigma) = \begin{cases} \mathbf{1} & \text{if } \llbracket \varphi \rrbracket_{\mathcal{V}}(u, \sigma) = \mathbf{0}, \\ \mathbf{0} & \text{otherwise.} \end{cases}$$

$$\llbracket \forall X \varphi \rrbracket_{\mathcal{V}}(u, \sigma) = \bigodot_{I \subseteq \mathrm{Pos}(u)} \llbracket \varphi \rrbracket_{\mathcal{V} \cup \{X\}}(u, \sigma[X \mapsto I]).$$

Fig. 1. Semantics of weighted MSO

We denote by $\mathrm{bMSO}(\mathbb{K}, \Sigma)$ the syntactic Boolean fragment of $\mathrm{MSO}(\mathbb{K}, \Sigma)$ consisting of MSO formulas where atomic formulas $k \in K \setminus \{\mathbf{0}, \mathbf{1}\}$, disjunction and existential quantifiers are disallowed, *i.e.*, given by the grammar

$$\varphi ::= \mathbf{0} \mid \mathbf{1} \mid P_a(x) \mid x \leq y \mid x \in X \mid \neg \varphi \mid \varphi \wedge \varphi \mid \forall x \varphi \mid \forall X \varphi$$

where $a \in \Sigma$, $x, y \in \mathrm{Var}$ and $X \in \mathrm{VAR}$. One can check, by induction, that the semantics of any bMSO formula over an arbitrary semiring $\mathbb{K}$ assumes values in $\{\mathbf{0}, \mathbf{1}\}$ and coincides with the classical semantics in $\mathbb{B}$.

For the sake of simplicity, we use macros for Boolean disjunction $\varphi \underline{\vee} \psi \stackrel{\text{def}}{=} \neg(\neg \varphi \wedge \neg \psi)$ and Boolean existential quantification $\underline{\exists} x \varphi \stackrel{\text{def}}{=} \neg \forall x \neg \varphi$, and $\underline{\exists} X \varphi \stackrel{\text{def}}{=} \neg \forall X \neg \varphi$. The semantics of $\underline{\vee}$ and $\underline{\exists}$ coincide with the classical semantics of disjunction and existential quantification in the Boolean semiring $\mathbb{B}$. Finally, we define $\varphi \stackrel{+}{\to} \psi \stackrel{\text{def}}{=} \neg \varphi \underline{\vee} (\varphi \wedge \psi)$ so that, if φ is a Boolean formula (i.e., $\llbracket \varphi \rrbracket(\Sigma^+) \subseteq \{\mathbf{0}, \mathbf{1}\}$), $\llbracket \varphi \stackrel{+}{\to} \psi \rrbracket(u, \sigma) = \llbracket \psi \rrbracket(u, \sigma)$ if $\llbracket \varphi \rrbracket(u, \sigma) = \mathbf{1}$, and $\llbracket \varphi \stackrel{+}{\to} \psi \rrbracket(u, \sigma) = \mathbf{1}$ if $\llbracket \varphi \rrbracket(u, \sigma) = \mathbf{0}$.

A common fragment of $\mathrm{MSO}(\mathbb{K}, \Sigma)$ is the weighted first-order logic $\mathrm{FO}(\mathbb{K}, \Sigma)$, where no second-order quantifier appears (note that second order variables may still appear free):

$$\varphi ::= k \mid P_a(x) \mid x \leq y \mid x \in X \mid \neg \varphi \mid \varphi \vee \varphi \mid \varphi \wedge \varphi \mid \exists x \varphi \mid \forall x \varphi.$$

We let $\mathrm{bFO}(\mathbb{K}, \Sigma)$ be the fragment of $\mathrm{bMSO}(\mathbb{K}, \Sigma)$ with no second-order quantifiers:

$$\varphi ::= \mathbf{0} \mid \mathbf{1} \mid P_a(x) \mid x \leq y \mid x \in X \mid \neg \varphi \mid \varphi \wedge \varphi \mid \forall x \varphi.$$

We also let bFO+mod be the fragment of bMSO consisting of bFO augmented with modulo constraints $x \equiv_\ell m$ for constants $1 \leq m \leq \ell$ (since the positions of words start with 1, it is more convenient to compute modulo as a value between 1 and ℓ). The semantics is given by

$$\llbracket x \equiv_\ell m \rrbracket(u, \sigma) = \begin{cases} \mathbf{1} & \text{if } \sigma(x) \equiv m \bmod \ell \\ \mathbf{0} & \text{otherwise} \end{cases}$$

4

and it is bMSO-definable by

$$x \equiv_\ell m \quad \overset{\text{def}}{=} \quad \forall X \Big(\big[(x \in X) \wedge \big(\forall y (y \in X \wedge y > \ell) \overset{+}{\to} y - \ell \in X \big) \big] \overset{+}{\to} m \in X \Big).$$

For $\mathcal{L} \subseteq$ bMSO, an $\mathcal{L}$-*step formula* is a finite disjunction $\bigvee_i (\varphi_i \wedge k_i)$ with $\varphi_i \in \mathcal{L}$ and $k_i \in K$. All bMSO-step formulas are clearly recognizable. By [DG07], $\forall x\, \varphi$ is recognizable for any step formula φ. A restricted fragment RMSO$(\mathbb{K}, \Sigma)$ of MSO$(\mathbb{K}, \Sigma)$ was defined in [DG07] in order to obtain

Theorem 3 ([DG07]). *A formal power series is recognizable if and only if it is definable in* RMSO$(\mathbb{K}, \Sigma)$.

Essentially, RMSO$(\mathbb{K}, \Sigma)$ restricts the use of universal second order quantification to bMSO formulas and of universal first-order quantification to bMSO-step formulas.

3   Transitive closure logic and weighted automata

To ease notation, we write $[\![\varphi(x, y)]\!](u, i, j)$ instead of $[\![\varphi(x, y)]\!](u, [x \mapsto i, y \mapsto j])$. We allow constants, modulo constraints and comparisons, like *e.g.* $x \leq y+2$. We use first and last as abbreviations for the first and last positions of a word. All of these shortcuts can be replaced by suitable bFO-formulas, except "$x \equiv_\ell m$" with $1 \leq m \leq \ell$, which is bMSO-definable.

Transitive closure. For a formula $\varphi(x, y)$ with at least two free variables x and y, we let

$$\varphi^1(x, y) \overset{\text{def}}{=} \varphi(x, y) \wedge (x \leq y),$$
$$\varphi^{n+1}(x, y) \overset{\text{def}}{=} \exists z \big[(x < z < y) \wedge \varphi(x, z) \wedge \varphi^n(z, y) \big], \quad \text{for } n \geq 1.$$

We now define a transitive closure operator $\text{TC}^<_{xy}\varphi$ by

$$\text{TC}^<_{xy}\varphi = \bigvee_{n \geq 1} \varphi^n.$$

This infinite disjunction is well-defined: $[\![\varphi^n(x, y)]\!](u, \sigma) = 0$ if $n \geq \max(2, |u|)$, *i.e.*, on each pair (u, σ), only finitely many disjuncts assume a nonzero value. Intuitively, the $\text{TC}^<_{xy}$ operator generalizes the forward transitive closure operator from the Boolean case: a formula $\varphi(x, y)$ with two free variables defines a binary relation on positions of a word u, namely $\{(i, j) \mid (i < j) \wedge [\![\varphi(x, y)]\!](u, i, j)\}$. The relation defined by $\text{TC}^<_{xy}\varphi$ is the transitive closure of this relation, augmented with all pairs (i, i) satisfying φ.

The fragment FO+TC$^<(\mathbb{K}, \Sigma)$ is then defined by the grammar

$$\varphi ::= k \mid P_a(x) \mid x \leq y \mid \neg\varphi \mid \varphi \vee \varphi \mid \varphi \wedge \varphi \mid \exists x \varphi \mid \forall x \varphi \mid \text{TC}^<_{xy}\varphi$$

with the restriction that one can apply negation only over bFO-formulas, *i.e.*, without atomic formulas $k \in K \setminus \{\mathbf{0}, \mathbf{1}\}$, nor disjunctions, nor existential quantifiers.

5

Bounded transitive closure. We also introduce a *bounded* transitive closure operation. For each integer $N > 0$, the $N\text{-TC}_{xy}^<$ operator is given by

$$N\text{-TC}_{xy}^< \varphi = \text{TC}_{xy}^<\big((y \leq x + N) \wedge \varphi(x,y)\big). \tag{1}$$

Equivalently, $N\text{-TC}_{xy}^< \varphi = \bigvee_{n \geq 1} \varphi^{n,N}$ where $\varphi^{1,N}(x,y) \overset{\text{def}}{=} \varphi(x,y) \wedge (x \leq y \leq x + N)$ and for $n \geq 2$,

$$\varphi^{n,N}(x,y) = \exists z_0 \cdots \exists z_n \big[x = z_0 \wedge y = z_n \wedge \bigwedge_{1 \leq \ell \leq n} (z_{\ell-1} < z_\ell \leq z_{\ell-1} + N) \wedge \varphi(z_{\ell-1}, z_\ell) \big]. \tag{2}$$

All results of the paper concerning the bounded transitive closure operator also hold with alternate choices for its definition: for instance, we may allow a fixed number of unbounded steps in (2), or we may replace $\varphi^{1,N}$ by φ in the second definition of $N\text{-TC}_{xy}^<$. The fragment $\text{FO+BTC}^<(\mathbb{K}, \Sigma)$ is then defined by the grammar

$$\varphi ::= k \mid P_a(x) \mid x \leq y \mid \neg\varphi \mid \varphi \vee \varphi \mid \varphi \wedge \varphi \mid \exists x \varphi \mid \forall x \varphi \mid N\text{-TC}_{xy}^< \varphi,$$

with the restriction that one can apply negation only over bFO-formulas, and with $N \geq 1$: to build formulas of $\text{FO+BTC}^<$, we are allowed to use each of the operators $1\text{-TC}_{xy}^<$, $2\text{-TC}_{xy}^<$, and so on. We denote by $\mathbb{K}^{\text{FO+BTC}^<}\langle\!\langle \Sigma^+ \rangle\!\rangle$ the class of all $\text{FO+BTC}^<(\mathbb{K}, \Sigma)$-definable series.

Example 4. Let $\varphi(x,y) \overset{\text{def}}{=} (y = x + 1) \wedge \forall z\, 2 \wedge (x = 1 \overset{+}{\to} \forall z\, 2)$ on $\mathbb{K} = \mathbb{N}$. Let $u = u_1 \cdots u_n$. We have $[\![N\text{-TC}_{xy}^< \varphi]\!](u, \mathsf{first}, \mathsf{last}) = \bigodot_{i=1}^{n-1} [\![\varphi]\!](u, i, i+1)$ due to the constraint $y = x + 1$ in φ. Now, $[\![\varphi]\!](u, 1, 2) = 2^{2|u|}$ and $[\![\varphi]\!](u, i, i+1) = 2^{|u|}$ if $i > 1$, so $[\![N\text{-TC}_{xy}^< \varphi]\!](u, \mathsf{first}, \mathsf{last}) = 2^{|u|^2}$. This example shows in particular that the class of recognizable series is not closed under $\text{BTC}^<$.

Example 5. Consider the formula $\psi = 2\text{-TC}_{x,y}^<(2 \wedge y = x + 2)$ over Σ and $\mathbb{K} = \mathbb{N}$. We have

$$[\![\psi]\!](u, \mathsf{first}, \mathsf{last}) = \begin{cases} 2^n & \text{if } |u| = 2n + 1 \text{ with } n \geq 1 \\ 0 & \text{otherwise.} \end{cases}$$

Notice that the support of $[\![\psi]\!]$ (*i.e.*, the language of all words mapped to nonzero values) is not bFO-definable. Therefore, $[\![\psi]\!]$ is not FO-definable, since otherwise, we would have $[\![\psi]\!] = [\![\varphi]\!]$ for φ an FO-formula, and the bFO formula obtained from φ by changing all nonzero constants of $\mathbb{N}$ to **1** and replacing $\vee$ and $\exists$ by $\underline{\vee}$ and $\underline{\exists}$ respectively would recognize the support of $[\![\psi]\!]$ (this works because $\mathbb{N}$ is a positive semiring).

Example 6. It is well-known that *modulo* can be expressed in $\text{bFO+BTC}^<$. For $1 \leq m \leq \ell$, the predicate is expressed by

$$x \equiv_\ell m \overset{\text{def}}{=} \ell\text{-TC}_{yz}^<(z = y \vee z = y + \ell)(m, x)$$

while the binary relation can be expressed with

$$x \equiv_\ell y \overset{\text{def}}{=} (y = x) \vee \ell\text{-TC}_{x'y'}^<(y' = x' + \ell)(x, y) \vee \ell\text{-TC}_{x'y'}^<(y' = x' + \ell)(y, x).$$

We now consider syntactical restrictions of $\text{FO+TC}^<$ and $\text{FO+BTC}^<$, inspired by normal form formulas of [NS00] where only one "external" transitive closure is allowed.

Definition 7. *For $\mathcal{L} \subseteq$ bMSO, $\mathrm{BTC}^<_{\mathrm{step}}(\mathcal{L})$ consists of formulas of the form N-$\mathrm{TC}^<_{xy}\varphi$, where $\varphi(x, y)$ is an $\mathcal{L}$-step formula with two free variables x, y. We say that $f \in \mathbb{K}\langle\!\langle \Sigma^+ \rangle\!\rangle$ is $\mathrm{BTC}^<_{\mathrm{step}}(\mathcal{L})$-definable if there exists an $\mathcal{L}$-step formula $\varphi(x, y)$ such that, for all $u \in \Sigma^+$:*

$$f(u) = [\![N\text{-}\mathrm{TC}^<_{xy}\varphi]\!](u, \mathsf{first}, \mathsf{last}).$$

We define $\mathrm{TC}^<_{\mathrm{step}}(\mathcal{L})$ and $\mathrm{TC}^<_{\mathrm{step}}(\mathcal{L})$-definability similarly, by replacing N-$\mathrm{TC}^<_{xy}\varphi$ with $\mathrm{TC}^<_{xy}\varphi$.

We also define fragments of the logic $\mathrm{RMSO}(\mathbb{K}, \Sigma)$ introduced in [DG07].

Definition 8. *For $\mathcal{L} \subseteq$ bMSO, let $\exists\forall_{\mathrm{step}}(\mathcal{L})$ consists of all MSO-formulas of the form $\exists X \forall x\, \varphi(x, X)$ with φ an $\mathcal{L}$-step formula.*

The following result characterizes the expressive power of weighted automata.

Theorem 9. *Let $\mathbb{K}$ be a (possibly noncommutative) semiring and $f \in \mathbb{K}\langle\!\langle \Sigma^+ \rangle\!\rangle$. The following assertions are equivalent over $\mathbb{K}$ and Σ.*
- *(1) f is recognizable.*
- *(2) f is $\mathrm{BTC}^<_{\mathrm{step}}(\mathrm{bFO+mod})$-definable.*
- *(3) f is $\mathrm{BTC}^<_{\mathrm{step}}(\mathrm{bMSO})$-definable.*
- *(4) f is $\mathrm{TC}^<_{\mathrm{step}}(\mathrm{bFO+mod})$-definable.*
- *(5) f is $\mathrm{TC}^<_{\mathrm{step}}(\mathrm{bMSO})$-definable.*
- *(6) f is $\exists\forall_{\mathrm{step}}(\mathrm{bFO})$-definable.*
- *(7) f is $\exists\forall_{\mathrm{step}}(\mathrm{bMSO})$-definable.*

Proof. The implications (2) $\Rightarrow$ (3), (4) $\Rightarrow$ (5) and (6) $\Rightarrow$ (7) are clear since bFO+mod $\subseteq$ bMSO, and (7) $\Rightarrow$ (1) follows from Theorem 3 because $\exists\forall_{\mathrm{step}}(\mathrm{bMSO}) \subseteq \mathrm{RMSO}$. The implications (2) $\Rightarrow$ (4) and (3) $\Rightarrow$ (5) are also obvious since $\mathrm{BTC}^<$ can be defined with $\mathrm{TC}^<$. We prove (1) $\Rightarrow$ (2), (1) $\Rightarrow$ (6), and (5) $\Rightarrow$ (7). For the first two implications, let us fix a weighted automaton $\mathcal{A} = (Q, \mu, \lambda, \gamma)$ with $Q = \{1, \ldots, n\}$.

For $d \geq 1$ and $p, q \in Q$, we use a formula $\psi^d_{p,q}(x, y)$ to compute the weight of the factor of length d located between positions x and y, when $\mathcal{A}$ goes from p to q:

$$\psi^d_{p,q}(x, y) \stackrel{\mathrm{def}}{=} (y = x + d - 1) \wedge \bigvee_{v = v_1 \cdots v_d} \left(\mu(v)_{p,q} \wedge \bigwedge_{1 \leq i \leq d} P_{v_i}(x + i - 1) \right)$$

We easily show that for every word u, and every positions i, j,

$$[\![\psi^d_{p,q}(x, y)]\!](u, i, j) = \begin{cases} \mu(u[i..j])_{p,q} & \text{if } j = i + d - 1, \\ \mathbf{0} & \text{otherwise.} \end{cases}$$

Let us show (1) $\Rightarrow$ (2). We construct a formula $\varphi(x, y)$ of bFO+mod, such that $[\![\mathcal{A}]\!](u) = [\![2n\text{-}\mathrm{TC}^<_{xy}\varphi]\!](u, \mathsf{first}, \mathsf{last})$. The idea, inspired by [Tho82], consists in making the transitive closure pick positions $z_\ell = \ell n + q_\ell$, with $1 \leq q_\ell \leq n$, for successive values of ℓ, to encode runs of $\mathcal{A}$ going through state q_ℓ just before reading the letter at position $\ell n + 1$. To make this still work for $\ell = 0$, one can assume wlog. that $\lambda(1) = \mathbf{1}$ and $\lambda(q) = \mathbf{0}$ for $q \neq 1$, *i.e.*, the only initial state yielding a nonzero value is $q_0 = 1$. Consider slices $[\ell n + 1, (\ell + 1)n]$ of positions in the word where we evaluate the formula (the last slice might be incomplete). Each position y is located in exactly one such slice. We write $\langle y \rangle = \ell n + 1$ for the first position of that slice, as well as $[y] \stackrel{\mathrm{def}}{=} y + 1 - \langle y \rangle \in Q$ for

7

the corresponding "offset". Notice that, for $q \in Q$, $[y] = q$ can be expressed in bFO+mod simply by $y \equiv_n q$. Hence, we will freely use $[y]$ as well as $\langle y \rangle = y + 1 - [y]$ as macros in formulas. Our $\mathrm{TC}^<$-formula picks positions x and y marked $\bullet$ in Figure 2, and computes the weight of the factor of length n between the positions $\langle x \rangle$ and $\langle y \rangle - 1$, assuming states $[x]$ and $[y]$ just before these positions.
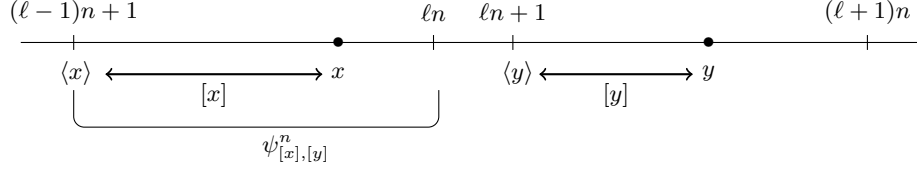
Fig. 2. Positions picked by the $\mathrm{TC}^<$-formula

The formula φ distinguishes the cases where x is far or near from the last position:

$$\varphi(x,y) = (\langle x \rangle + 2n \leq \mathsf{last}) \wedge (\langle y \rangle = \langle x \rangle + n) \wedge \psi^n_{[x],[y]}(\langle x \rangle, \langle y \rangle - 1)$$
$$\vee \, (\langle x \rangle + 2n > \mathsf{last}) \wedge (y = \mathsf{last} \wedge \bigvee_{q \in Q} \psi^{y - \langle x \rangle + 1}_{[x],q}(\langle x \rangle, y) \wedge \gamma(q)$$

This definition implies that for every word u and every positions i, j,

$$[\![\varphi(x,y)]\!](u, i, j) = \begin{cases} \mu(u[\langle i \rangle .. \langle j \rangle - 1])_{[i],[j]} & \text{if } \langle j \rangle = \langle i \rangle + n \wedge \langle i \rangle + 2n \leq \mathsf{last}, \\ \delta^{[i]} \odot \mu(u[\langle i \rangle .. j]) \odot \gamma & \text{if } j = \mathsf{last} \wedge \langle i \rangle + 2n > \mathsf{last}, \\ \mathbf{0} & \text{otherwise}, \end{cases} \tag{3}$$

where δ^ℓ is the row vector with $\mathbf{1}$ in position ℓ and $\mathbf{0}$ elsewhere. Now, by definition of the $2n\text{-}\mathrm{TC}^<_{xy}$ operator,

$$[\![2n\text{-}\mathrm{TC}^<_{xy}\varphi(x,y)]\!](u, \mathsf{first}, \mathsf{last}) = \bigoplus_{m \geq 1} [\![\varphi^{m,2n}(x,y)]\!](u, \mathsf{first}, \mathsf{last}).$$

We show that there is a single nonzero value in this sum which is $[\![\varphi^{m,2n}(x,y)]\!](u, \mathsf{first}, \mathsf{last})$ with $m = \max(1, \lfloor \frac{|u|-1}{n} \rfloor)$.

Let $1 = i_0, \ldots, i_m = |u|$ be a sequence of positions chosen in an assignment of variables $z_0, \ldots, z_m$ in (2), yielding a nonzero value in $[\![\varphi^{m,2n}(x,y)]\!](u, \mathsf{first}, \mathsf{last})$. The first case of (3) implies that $\langle i_\ell \rangle = \langle i_{\ell-1} \rangle + n$ for all $\ell \in \{1, \ldots, m-1\}$, so that $\langle i_\ell \rangle = \ell n + 1$ if $\ell < m$.

If $|u| \leq 2n$, then it follows from (3) that $[\![\varphi^{1,2n}(x,y)]\!](u, \mathsf{first}, \mathsf{last}) = [\![\varphi(x,y)]\!](u, \mathsf{first}, \mathsf{last}) = \lambda \odot \mu(u) \odot \gamma$ (the second case of (3) applies). Further, $[\![\varphi^{m,2n}(x,y)]\!](u, \mathsf{first}, \mathsf{last}) = \mathbf{0}$ for $m \geq 2$, because $1 < i_1 < |u|$ implies that $[\![\varphi(x,y)]\!](u, i_0, i_1) = \mathbf{0}$ (the first case of (3) is excluded because $\langle i_0 \rangle + 2n > |u|$ and the second one because $i_1 \neq |u|$). Therefore, $[\![2n\text{-}\mathrm{TC}^<_{xy}\varphi(x,y)]\!](u, \mathsf{first}, \mathsf{last}) = \lambda \odot \mu(u) \odot \gamma = [\![\mathcal{A}]\!](u)$.

If $|u| > 2n$ then $[\![\varphi(x,y)]\!](u, \mathsf{first}, \mathsf{last}) = 0$. Then $m \geq 2$ and $\langle i_\ell \rangle = \ell n + 1$ for $\ell < m$. By (3), $[\![\varphi^{m,2n}(x,y)]\!](u, \mathsf{first}, \mathsf{last}) \neq \mathbf{0}$ implies $\langle i_{m-2} \rangle + 2n \leq |u| < \langle i_{m-1} \rangle + 2n$, that is $m = \lfloor \frac{|u|-1}{n} \rfloor$. We deduce that $[\![2n\text{-}\mathrm{TC}^<_{xy}\varphi(x,y)]\!](u, \mathsf{first}, \mathsf{last}) = [\![\varphi^{m,2n}(x,y)]\!](u, \mathsf{first}, \mathsf{last})$ for $m = \lfloor \frac{|u|-1}{n} \rfloor$. The assignment $1 = i_0 < i_1 < \ldots < i_m = |u|$ is uniquely defined by the sequence $([i_1], \ldots, [i_{m-1}]) \in Q^{m-1}$, since $i_\ell = \langle i_\ell \rangle - 1 + [i_\ell] = \ell n + [i_\ell]$ for $\ell < m$. The possible choices of this tuple induce

8

the sum in the following evaluation of $[\![\varphi^{m,2n}(x,y)]\!](u, \mathsf{first}, \mathsf{last})$, where we rename the sequence $([i_1], \ldots, [i_{m-1}])$ as a vector $\boldsymbol{q} = (q_1, \ldots, q_{m-1})$ and we omit the variable names (x, y) in the right hand side. Recall that $q_0 = 1$ is the unique initial state giving a nonzero value.

$$[\![\varphi^{m,2n}(x,y)]\!](u, \mathsf{first}, \mathsf{last})$$

$$= \bigoplus_{\boldsymbol{q} \in Q^{m-1}} \left[\bigodot_{\ell=1}^{m-1} [\![\varphi]\!]\big(u, \ (\ell-1)n + q_{\ell-1}, \ \ell n + q_\ell - 1\big) \right] \odot [\![\varphi]\!]\big(u, \ (m-1)n + q_{m-1}, \ |u|\big)$$

$$= \bigoplus_{\boldsymbol{q} \in Q^{m-1}} \left(\left[\bigodot_{\ell=1}^{m-1} \mu\big(u[(\ell-1)n+1..\ell n]\big)_{q_{\ell-1},q_\ell} \right] \odot \big(\delta^{q_{m-1}} \odot \mu(u[(m-1)n+1..|u|]) \odot \gamma\big) \right)$$

$$= \delta^{q_0} \odot \mu\big(u[1..(m-1)n]\big) \odot \mu(u[(m-1)n+1..|u|]) \odot \gamma$$

$$= \lambda \ \odot \mu(u) \odot \gamma.$$

Let us now outline the proof of (1) $\Rightarrow$ (6), which relies on a similar technique. To encode runs of $\mathcal{A}$ by a formula $\exists X \forall x \, \varphi(x, X)$ with φ a bFO-step formula, we write an $\exists \forall_{\mathsf{step}}(\mathrm{bFO})$ formula forcing X to represent positions $x_0 < t_0 < x_1 < t_1 < \ldots$ with $x_\ell = (2n+1)\ell + 1$, where the distance $t_\ell - x_\ell$ encodes the state of $\mathcal{A}$ reached just before position x_ℓ. For instance, since we assume as above that the only initial state yielding a nonzero value is $q_0 = 1$, we enforce $x_0 = 1$ and $t_0 = 2$, so that $t_0 - x_0 = 1$ encodes that state. In order to distinguish the x_ℓ's from the t_ℓ's, we enforce $1 \leq t_\ell - x_\ell \leq n$ (so that $t_\ell - x_\ell \in Q$) but we make a gap between t_ℓ and $x_{\ell+1}$, namely $x_{\ell+1} - t_\ell > n$. We use shortcuts like $\{z, t\} \subseteq X$, or $X \cap \,]z, t] = \emptyset$, where $]z, t] = \{z' \mid z < z' \leq t\}$, which are easily
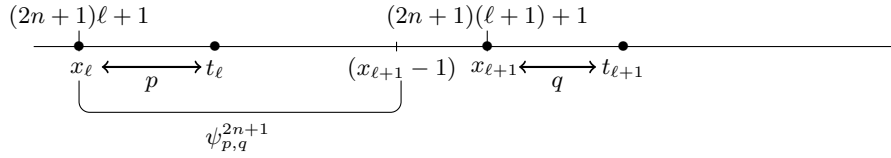
Fig. 3. Positions represented by X, marked $\bullet$

bFO-definable. The variables x and y appearing in φ_{far} and φ_{near} represent two consecutive x_ℓ's. We write

$$\varphi = \left[\mathsf{last} > n \wedge \{1, 2\} \subseteq X \wedge (\varphi_{\mathrm{far}} \vee \varphi_{\mathrm{near}}) \right]$$
$$\vee \left[\mathsf{last} \leq n \wedge X = \emptyset \wedge \left[x = \mathsf{first} \xrightarrow{+} \bigvee_{p,q \in Q} \lambda(p) \wedge \psi_{p,q}^{\mathsf{last}}(1, \mathsf{last}) \wedge \gamma(q) \right] \right].$$

The outermost disjunction distinguishes the cases of long or short words, while the disjunction $[\varphi_{\mathrm{far}} \vee \varphi_{\mathrm{near}}](x, X)$ discriminates between positions $x \in X$ located far/near from the last position. For the case of short words, the product resulting from the quantification $\forall x$ has value $[\![\mathcal{A}]\!](u)$ thanks to the premise $x = \mathsf{first}$ of the implication $\xrightarrow{+}$ (without it, we would compute $[\![\mathcal{A}]\!](u)^{|u|}$). Moreover, the sum resulting from $\exists X$ has a single nonzero value thanks to $X = \emptyset$. Therefore, the

overall semantics is indeed $[\![\mathcal{A}]\!](u)$. We define φ_{far} as

$$\varphi_{\text{far}}(x, X) = (x + 3n + 1 \leq \mathsf{last}) \wedge \Big((x \in X \wedge X \cap\,]x, x + n] \neq \emptyset) \overset{+}{\to}$$

$$\exists y \bigvee_{p,q \in Q} X \cap\,]x, x + 3n + 1] = \{x + p, y, y + q\} \wedge \psi_{p,q}^{2n+1}(x, y - 1)\Big).$$

Note that the only position y for which $[\![\psi_{p,q}^{2n+1}]\!](x, y - 1) \neq \mathbf{0}$ is $x + 2n + 1$, by definition of $\psi_{p,q}^{2n+1}$. To produce a nonzero value, we must also have $x + 3n + 1 \leq \mathsf{last}$, so that $y + n \leq \mathsf{last}$. Further, $y \in X \wedge X \cap\,]y, y + n] \neq \emptyset$, so that the pattern repeats. We define φ_{near} as

$$\varphi_{\text{near}}(x, X) = (x + 3n + 1 > \mathsf{last}) \wedge \Big((x \in X \wedge X \cap\,]x, x + n] \neq \emptyset) \overset{+}{\to}$$

$$\bigvee_{p,q \in Q} X \cap\,]x, \mathsf{last}] = \{x + p\} \wedge \psi_{p,q}^{\mathsf{last} - x + 1}(x, \mathsf{last}) \wedge \gamma(q)\Big).$$

Then one can check that $\varphi(x, X)$ is a bFO-step formula and $[\![\exists X \forall x \varphi(x, X)]\!](u, \mathsf{first}, \mathsf{last}) = [\![\mathcal{A}]\!](u)$. As in [Tho82], we could use a more compact formula by encoding states in binary.

We finally prove (5) $\Rightarrow$ (7). Let $\psi(z, t) = [\mathrm{TC}_{xy}^{<}\big(\bigvee_{i \in I} \varphi_i(x, y) \wedge k_i\big)](z, t)$ be a $\mathrm{TC}_{\text{step}}^{<}(\text{bMSO})$-formula: we note $\varphi(x, y) = \bigvee_{i \in I} \varphi_i(x, y) \wedge k_i$. Let $u \in \Sigma^+$ be a word, $j \in \mathrm{Pos}(u)$ and $J \subseteq \mathrm{Pos}(u)$. If $j < \max(J)$ then we let $\mathrm{next}(j, J) = \inf\{\ell \in J \mid j < \ell\}$. We define the bMSO-step formula

$$\xi(x, X) \overset{\text{def}}{=} \bigvee_{i \in I} k_i \wedge \exists y (x < y \wedge X \cap\,]x, y] = \{y\} \wedge \varphi_i(x, y))$$

and we can easily check that

$$[\![\xi]\!](u, j, J) = \begin{cases} [\![\varphi]\!](u, j, \mathrm{next}(j, J)) & \text{if } j < \max(J) \\ \mathbf{0} & \text{otherwise.} \end{cases}$$

The $\exists\forall_{\text{step}}(\text{bMSO})$-formula equivalent to ψ is then

$$\psi'(z, t) = \exists X \forall x \left(\begin{array}{c} z = t \wedge X = \emptyset \wedge (x = z \overset{+}{\to} \varphi(z, t)) \\ \vee \\ z \neq t \wedge \{z, t\} \subseteq X \subseteq [z, t] \wedge ((x \in X \wedge x < \max(X)) \overset{+}{\to} \xi(x, X)) \end{array} \right).$$

Note that bMSO-step formulas are closed under positive Boolean combinations so the formula ψ' above is indeed an $\exists\forall_{\text{step}}(\text{bMSO})$-formula. Let $u \in \Sigma^+$ and $j, \ell \in \mathrm{Pos}(u)$. It is easy to check that $[\![\psi]\!]'(u, j, j) = [\![\psi]\!](u, j, j)$. Assume now that $j < \ell$ and let $j = j_0 < j_1 < \cdots < j_m = \ell$ for some $m > 0$. For $J = \{j_0, j_1, \ldots, j_m\}$, the formula $\alpha(X) = \forall x((x \in X \wedge x < \max(X)) \overset{+}{\to} \xi(x, X))$ computes $[\![\alpha]\!](u, J) = \bigodot_{n < m} [\![\varphi]\!](u, j_n, j_{n+1})$. Therefore,

$$[\![\psi]\!]'(u, j, \ell) = \bigoplus_{m > 0} \bigoplus_{j = j_0 < j_1 < \cdots < j_m = \ell} \bigodot_{n < m} [\![\varphi]\!](u, j_n, j_{n+1}) = \bigoplus_{m > 0} [\![\varphi^m]\!](u, j, \ell) = [\![\psi(z, t)]\!](u, j, \ell)$$

where the last equality comes from the definition of the transitive closure operator. $\qquad\square$

4 Weighted nested automata

Example 2 shows that weighted automata lack closure properties to capture FO+BTC$^<$. We introduce a notion of nested automata making up for this gap.

For $r \geq 0$, the class r-nwA(Σ) (r-nwA if Σ is understood) of *r-nested weighted automata* over Σ (and $\mathbb{K}$) consists of all tuples $(Q, \mu, \lambda, \gamma)$ where Q is the set of states, $\lambda, \gamma : Q \to K$, and $\mu : Q \times \Sigma \times Q \to (r-1)$-nwA$(\Sigma \times \{0,1\})$. Here, we agree that (-1)-nwA $= K$. In particular, a 0-nwA(Σ) is a weighted automaton over Σ. Intuitively, the weight of a transition is computed by an automaton of the preceding level running on the whole word, where the additional $\{0,1\}$ component marks the letter of the transition whose weight is to be computed.

Let us formally define the behavior $[\![\mathcal{A}]\!] \in \mathbb{K}\langle\!\langle \Sigma^+ \rangle\!\rangle$ of $\mathcal{A} = (Q, \mu, \lambda, \gamma) \in r$-nwA$(\Sigma)$. If $r = 0$, then $[\![\mathcal{A}]\!]$ is the behavior of $\mathcal{A}$ considered as wA over Σ. For $r \geq 1$, the weight of a *run* $\rho = q_0 \xrightarrow{u_1} q_1 \xrightarrow{u_2} \cdots \xrightarrow{u_n} q_n$ of $\mathcal{A}$ on $u = u_1 \cdots u_n \in \Sigma^+$ is

$$\text{weight}(\rho) \stackrel{\text{def}}{=} \lambda(q_0) \odot \Big[\bigodot_{i=1}^n [\![\mu(q_{i-1}, u_i, q_i)]\!](u, i) \Big] \odot \gamma(q_n)$$

where $(u, i) \in (\Sigma \times \{0,1\})^+$ is the word $v = v_1 \cdots v_n$ with $v_i = (u_i, 1)$ and $v_j = (u_j, 0)$ if $j \neq i$. As usual, $[\![\mathcal{A}]\!](u)$ is the sum of the weights of all runs of $\mathcal{A}$ on u. Note that, unlike the nested automata of [tCS10], the values given by lower automata do not explicitly influence the applicable transitions.

Set nwA $= \bigcup_{r \geq 0} r$-nwA. A series $f \in \mathbb{K}\langle\!\langle \Sigma^+ \rangle\!\rangle$ is *r-nwA-recognizable* if $f = [\![\mathcal{A}]\!]$ for some r-nwA $\mathcal{A}$. It is *nwA-recognizable* if it is r-nwA-recognizable for some r. We let $\mathbb{K}^{r\text{-nwA}}\langle\!\langle \Sigma^+ \rangle\!\rangle$ (resp., $\mathbb{K}^{\text{nwA}}\langle\!\langle \Sigma^+ \rangle\!\rangle$) be the class of r-nwA-recognizable (resp., nwA-recognizable) series over $\mathbb{K}$ and Σ.

Example 10. A 1-nwA recognizing the series $u \mapsto 2^{|u|^2}$ over $\mathbb{N}$ is $\mathcal{A} = (\{p\}, \mu, 1, 1)$ where, for every $a \in \Sigma$, $\mu(p, a, p)$ is the weighted automaton of Example 1.
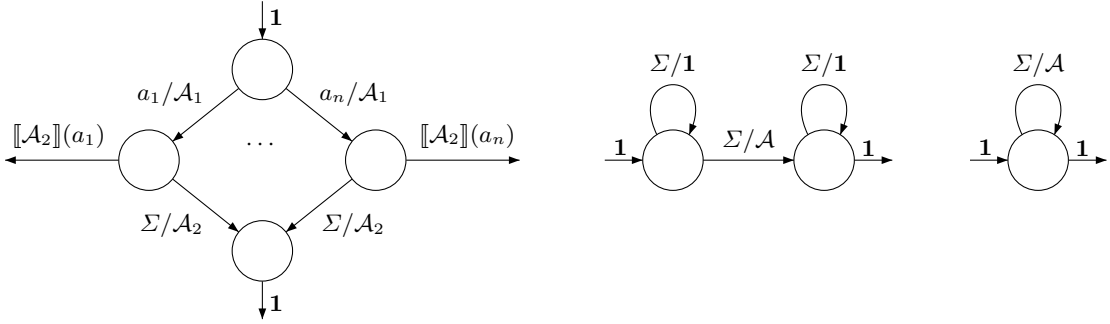
Proposition 11. *Every* nwA*-recognizable series is* FO+BTC$^<$*-definable.*

Proof. Starting from an r-nwA, we build an FO+BTC$^<$-formula by induction on r. The case $r = 0$ was already done in Theorem 9 since *modulo* is definable in FO+BTC$^<$. The induction step is shown like in the proof of the implication (1) $\Rightarrow$ (2) of Theorem 9. The main difference is in the definition of the formulas $\psi_{p,q}^\ell$, which use now the induction hypothesis. So let $\xi_{p,a,q}$ be the FO+BTC$^<$-formula describing the series denoted by the automaton $\mu(p, a, q)$ over the alphabet $\Sigma \times \{0,1\}$. Since the automaton $\mu(p, a, q)$ is only called on words of the form (u, i) with $i \in \text{Pos}(u)$, we may assume that $\xi_{p,a,q}$ has one free first-oder variable z for the position marked by i in u. We can then define

$$\psi_{p_0, p_\ell}^\ell(x, y) \stackrel{\text{def}}{=} (y = x + \ell - 1) \wedge \bigvee_{p_1, \ldots, p_{\ell-1} \in Q} \bigwedge_{0 \leq i < \ell} \bigvee_{a \in \Sigma} P_a(x + i) \wedge \xi_{p_i, a, p_{i+1}}(x + i) .$$

We have to guess non-deterministically the states $p_1, \ldots, p_{\ell-1}$ that the nested automaton will visit between the positions x and y, because each of them is useful to compute the runs of lower automata. We obtain for $u \in \Sigma^+$ and $1 \leq j \leq j + \ell - 1 \leq |u|$

$$[\![\psi_{p_0, p_\ell}^\ell]\!](u, j, j + \ell - 1) = \bigoplus_{p_1, \ldots, p_{\ell-1} \in Q} \bigodot_{0 \leq i < \ell} [\![\xi_{p_i, u_{j+i}, p_{i+1}}]\!](u, j + i) .$$

11

Fig. 4. nwA for conjunction, existential and universal quantifications

Then, we use exactly the same formula as in Theorem 9:

$$\varphi(x, y) = (\langle x \rangle + 2n \leq \mathsf{last}) \wedge (\langle y \rangle = \langle x \rangle + n) \wedge \psi^n_{[x],[y]}(\langle x \rangle, \langle y \rangle - 1)$$

$$\vee\, (\langle x \rangle + 2n > \mathsf{last}) \wedge (y = \mathsf{last}) \wedge \bigvee_{q \in Q} \psi^{y - \langle x \rangle + 1}_{[x], q}(\langle x \rangle, y) \wedge \gamma(q)$$

The macros $\langle x \rangle$ and $[x]$ are now expressed in FO+BTC$^<$ using the definition of modulo given in Example 6.

We remark that this construction yields an FO+BTC$^<$-formula using only $r + 2$ nested BTC$^<$-operators: $r + 1$ nested BTC$^<$-operators coming from the induction of the proof of Theorem 9, and the last one coming from the definition of the modulo operation. □

We will see in Section 5 that the opposite direction of Proposition 11 holds, too. We just prove for now some closure properties. The proof will be completed in Section 5. We say that a class $\mathcal{C}$ of MSO-definable series is closed under disjunction conjunction, first-order quantifications if, when $\llbracket \varphi \rrbracket, \llbracket \psi \rrbracket \in \mathcal{C}$, we have $\llbracket \varphi \vee \psi \rrbracket, \llbracket \varphi \wedge \psi \rrbracket, \llbracket \exists x \varphi \rrbracket, \llbracket \forall x \varphi \rrbracket \in \mathcal{C}$, respectively.

Proposition 12. *The class $\mathbb{K}^{\mathrm{nwA}}\langle\langle \Sigma^+ \rangle\rangle$ over a (possibly noncommutative) semiring $\mathbb{K}$ is stable under disjunction, conjunction, and first-order quantifications.*

Proof. Notice that an r-nwA-recognizable series is also $(r + 1)$-nwA-recognizable. This is trivial for $r = -1$, and an easy induction shows it for all $r \geq 0$. Therefore, given two nwA, one can assume that they are r-nwA for the same r. Closure under $\wedge$ is then obtained as follows. Let $\mathcal{A}_1, \mathcal{A}_2$ be two r-nwA over Σ. We still write $\mathcal{A}_1$ and $\mathcal{A}_2$ for the corresponding automata on $\Sigma \times \{0, 1\}$ that ignore the extra component. An automaton for the conjunction of $\mathcal{A}_1$ and $\mathcal{A}_2$ is depicted on the left of Figure 4. Formally, it is given by $(\{q_a \mid a \in \Sigma\} \uplus \{q_0, q_1\}, \mu, \lambda, \gamma)$ where $\lambda(q) = \mathbf{1}$ if $q = q_0$ and $\lambda(q) = \mathbf{0}$ otherwise, and for $a, b \in \Sigma$, $\mu(q_0, a, q_a) = \mathcal{A}_1$, $\mu(q_a, b, q_1) = \mathcal{A}_2$, $\mu(q_1, b, q_1) = \mathbf{1}$, and $\mu(p, a, q) = \mathbf{0}$ elsewhere, and finally $\gamma(q_0) = \mathbf{0}$, $\gamma(q_a) = \llbracket \mathcal{A}_2 \rrbracket(a)$ and $\gamma(q_1) = \mathbf{1}$. The idea is that $\llbracket \mathcal{A}_1 \rrbracket(u)$ is computed when reading the first letter, and $\llbracket \mathcal{A}_2 \rrbracket(u)$ is computed when reading the second letter if $|u| \geq 2$, or thanks to γ if $|u| = 1$.

Closure under $\vee$ is obtained using the disjoint union of the automata $\mathcal{A}_1$ and $\mathcal{A}_2$.

For first-order quantifiers, we use the extra nesting. For an r-nwA $\mathcal{A}$ over $\Sigma \times \{0, 1\}$, the $(r + 1)$-nwA for existential and universal first-order quantifications are depicted in Figure 4 (middle and right). □

Remark 13. If $f = \llbracket \mathcal{A} \rrbracket$ where $\mathcal{A} = (Q, \mu, \lambda, \gamma)$ is an r-nwA, then, for all $k \in K$, the series $k \odot f$ and $f \odot k$ are respectively recognized by the r-nwA $k \odot \mathcal{A} = (Q, \mu, k \odot \lambda, \gamma)$ and $\mathcal{A} \odot k = (Q, \mu, \lambda, \gamma \odot k)$.

5 Pebble weighted automata

We now consider *pebble weighted automata (*pwA*)*. A pwA has a read-only tape. At each step, it can move its head one position to the left or to the right (within the boundaries of the input tape), or either drop or lift a pebble at the current head position. Applicable transitions and weights depend on the current letter, current state, and the pebbles carried by the current position. Pebbles are handled using a stack policy: if the automaton has r pebbles and pebbles $r, r-1, \ldots, i$ have already been dropped, it can either lift pebble i (if $i \leq r$), drop pebble $i-1$ (if $i \geq 2$), or move. A pebble can only be dropped or lifted at the current head position.

As these automata can go in either direction, we add two fresh symbols $\triangleright$ and $\triangleleft$ to mark the beginning and the end of an input word. Let $\widetilde{\Sigma} = \Sigma \uplus \{\triangleright, \triangleleft\}$. To compute the value of $w = a_1 \cdots a_n \in \Sigma^+$, a pwA will work on a tape holding $\widetilde{w} = \triangleright w \triangleleft$. For convenience, we number the letters of $\widetilde{w}$ from 0, setting $\widetilde{w}_0 = \triangleright$, $\widetilde{w}_{n+1} = \triangleleft$, and $\widetilde{w}_i = a_i$ for $1 \leq i \leq n$.

Definition 14. *Let $r \geq 0$. An r-pebble weighted automaton (r-pwA) over $\mathbb{K}$ and Σ is a pair $\mathcal{A} = (Q, \mu)$ where Q is a finite set of states and $\mu : Q \times \widetilde{\Sigma} \times 2^r \times D \times Q \to K$ is the transition weight function, with $D = \{\leftarrow, \rightarrow, \mathrm{drop}, \mathrm{lift}\}$.*

A *configuration* of $\mathcal{A}$ on a word $w \in \Sigma^+$ of length n is a triple $(p, i, u) \in Q \times \{0, \ldots, n+2\} \times \{1, \ldots, n\}^{\leq r}$. The word w itself will be understood. Informally, p denotes the current state of $\mathcal{A}$ and i is the head position in $\widetilde{w}$, i.e, positions 0 and $n+1$ point to $\triangleright$ and $\triangleleft$, respectively, position $1 \leq i \leq n$ points to $\widetilde{w}_i \in \Sigma$, and position $n+2$ is outside $\widetilde{w}$. Finally, $u = u_j \cdots u_r$ with $1 \leq j \leq r+1$ encodes the locations of pebbles $j, \ldots, r$ ($u_\ell \in \{1, \ldots, n\}$ is the position of pebble ℓ) while pebbles $1, \ldots, j-1$ are currently not on the tape. For $m \in \{0, \ldots, n+1\}$, we set $u^{-1}(m) = \{\ell \in \{j, \ldots, r\} \mid u_\ell = m\}$ (viewing u as a partial function $u : \{1, \ldots, r\} \rightharpoonup \{0, \ldots, n+1\}$). Note that $u^{-1}(0) = u^{-1}(n+1) = \emptyset$.

There is a *step* of weight k from configuration (p, i, u) to configuration (q, j, v) if $i \leq n+1$, $k = \mu(p, \widetilde{w}_i, u^{-1}(i), d, q)$, and

$$\begin{cases} j = i-1 & \text{if } d = \leftarrow \\ j = i+1 & \text{if } d = \rightarrow \\ j = i & \text{otherwise} \end{cases} \quad \text{and} \quad \begin{cases} v = iu & \text{if } d = \mathrm{drop} \\ u = iv & \text{if } d = \mathrm{lift} \\ v = u & \text{otherwise.} \end{cases}$$

A *run* of $\mathcal{A}$ on w is a sequence of steps from a configuration $(p, 0, \varepsilon)$ to a configuration $(q, n+2, \varepsilon)$ (at the end, no pebble is left on the tape). We denote by $\mathrm{weight}(\rho)$ the product of the weights of the steps in the run. Run ρ is *simple* if whenever two configurations α and β appear in ρ, we have $\alpha \neq \beta$.

The series $\llbracket \mathcal{A} \rrbracket \in \mathbb{K}\langle\!\langle \Sigma^+ \rangle\!\rangle$ is defined by $\llbracket \mathcal{A} \rrbracket(w) = \bigoplus_{\rho \text{ simple run on } w} \mathrm{weight}(\rho)$. We denote by $\mathbb{K}^{r\text{-pwA}}\langle\!\langle \Sigma^+ \rangle\!\rangle$ the collection of formal power series defined by r-pwA, and we let $\mathbb{K}^{\mathrm{pwA}}\langle\!\langle \Sigma^+ \rangle\!\rangle = \bigcup_{r \geq 0} \mathbb{K}^{r\text{-pwA}}\langle\!\langle \Sigma^+ \rangle\!\rangle$. Note that a 0-pwA is in fact a 2-way weighted automaton. It follows from Proposition 17 that 2-way wA have the same expressive power as classical (1-way) wA.

Example 15. Let us sketch a 1-pwA $\mathcal{A}$ recognizing the series $u \mapsto 2^{|u|^2}$ over $\mathbb{N}$. The idea is that $\mathcal{A}$ drops its pebble successively on every position of the input word. Transitions for reallocating the pebble have weight **1**. When a pebble is dropped, $\mathcal{A}$ scans the whole word from left to right where every transition has weight 2. As this scan happens $|u|$ times, we obtain $\llbracket \mathcal{A} \rrbracket(u) = 2^{|u|^2}$.

The following theorem summarizes the main results of Sections 4 and 5.

Theorem 16. *For every commutative semiring $\mathbb{K}$, we have*

$$\mathbb{K}^{\mathrm{FO+BTC}^<}\langle\!\langle \Sigma^+ \rangle\!\rangle = \mathbb{K}^{\mathrm{pwA}}\langle\!\langle \Sigma^+ \rangle\!\rangle = \mathbb{K}^{\mathrm{nwA}}\langle\!\langle \Sigma^+ \rangle\!\rangle \,.$$

Propositions 17 and 18 will reveal that $\mathbb{K}^{\mathrm{FO+BTC}^<}\langle\!\langle \Sigma^+ \rangle\!\rangle \subseteq \mathbb{K}^{\mathrm{pwA}}\langle\!\langle \Sigma^+ \rangle\!\rangle \subseteq \mathbb{K}^{\mathrm{nwA}}\langle\!\langle \Sigma^+ \rangle\!\rangle$. The theorem follows, using Proposition 11 for the remaining inclusion. While the first inclusion is easy, the following result is more difficult.

Proposition 17. *For every commutative semiring $\mathbb{K}$ and every $r \geq 0$, we have*

$$\mathbb{K}^{r\text{-pwA}}\langle\!\langle \Sigma^+ \rangle\!\rangle \subseteq \mathbb{K}^{r\text{-nwA}}\langle\!\langle \Sigma^+ \rangle\!\rangle \,.$$

Proof. We provide a translation of a generalized version of r-pwA to r-nwA. That generalized notion equips an r-pwA $\mathcal{A} = (P, \mu)$ with an equivalence relation $\sim \,\subseteq P \times P$, which is canonically extended to configurations of $\mathcal{A}$: we write $(p, i, u) \sim (p', i', u')$ if $p \sim p'$, $i = i'$, and $u = u'$. The semantics of $[\![\mathcal{A}]\!]_\sim$ is then defined by replacing equality of configurations with $\sim$-equivalence of configurations in the definition of a simple run. To stress this fact, we henceforth say that a run is $\sim$-*simple*. We will explain below why it is useful to consider such an equivalence relation in the inductive process.

So let $r \geq 0$, let $\mathcal{A} = (P, \mu)$ be an r-pwA over $\mathbb{K}$ and Σ, and let $\sim \,\subseteq P \times P$ be an equivalence relation. For technical reasons, we assume wlog. that all runs of $\mathcal{A}$ in which a pebble is dropped and immediately lifted have weight $\mathbf{0}$. We will build an r-nwA $\langle\mathcal{A}\rangle_\sim = (Q, \nu, \lambda, \gamma)$ over Σ such that $[\![\langle\mathcal{A}\rangle_\sim]\!] = [\![\mathcal{A}]\!]_\sim$.

The construction of $\langle\mathcal{A}\rangle_\sim$ proceeds inductively, on the number of pebbles r. It involves two alternating transformations, which are illustrated in Figure 5. The left-hand side depicts a $\sim$-simple run of $\mathcal{A}$ on some word w with factor ab. To simulate such a run, $\langle\mathcal{A}\rangle_\sim$ scans w from left to right and guesses, at each position z, the sequence of those states and directions that are encountered at z while pebble r has not been dropped. The state of $\langle\mathcal{A}\rangle_\sim$ taken before reading the a at position z is the sequence $q = p_0 \to p_3 \leftarrow p_4 \,\mathrm{drop}\, p_5 \,\hat{p}_5 \,\mathrm{lift}\, p_6 \leftarrow p_7 \,\mathrm{drop}\, p_8 \,\hat{p}_8 \,\mathrm{lift}\, p_9 \to$ (which is enriched by the guessed input letter a, as explained below). As the micro-states $p_0, p_3, \ldots$ form a segment of a $\sim$-simple run, $p_0, p_3, p_4, p_6, p_7, p_9$ are pairwise distinct (wrt. $\sim$) and so are $p_5, \hat{p}_5, p_8, \hat{p}_8$. Recall that q actually restricts to run segments in which pebble r has not, or just, been dropped (shown in solid lines in Fig 5(a)). Missing segments are deferred to an $(r-1)$-nwA $\mathcal{B}_q$, which is called at position z and computes the run segments from p_5 to $\hat{p}_5$ and from p_8 to $\hat{p}_8$ that both start in z (shown in dashed lines in Fig. 5(a)). To this aim, $\mathcal{B}_q$ works on an extension of w where position z is marked, indicating that pebble r is considered to be at z.

Before we go into the construction of $\mathcal{B}_q$, let us formalize the r-nwA $\langle\mathcal{A}\rangle_\sim$. We set $\Pi = (P\{\to, \leftarrow\} \cup P\{\mathrm{drop}\}PP\{\mathrm{lift}\})^* P\{\to\}$. Sequences from Π may keep track of states and directions that are taken at one given position. As aforementioned, they must meet the requirements of $\sim$-simple runs so that only some of them can be considered as states. To determine the set $Q \subseteq (\widetilde{\Sigma} \uplus \{\Box\}) \times \Pi$ of states of $\langle\mathcal{A}\rangle_\sim$, we therefore define, given $\pi \in \Pi$, projections $\mathrm{proj}_1(\pi) \in P^+$ and $\mathrm{proj}_2(\pi) \in (PP)^*$ inductively as follows:

$$\mathrm{proj}_1(p \to \pi) = \mathrm{proj}_1(p \leftarrow \pi) = p \,\mathrm{proj}_1(\pi) \qquad \mathrm{proj}_2(p \to \pi) = \mathrm{proj}_2(p \leftarrow \pi) = \mathrm{proj}_2(\pi)$$
$$\mathrm{proj}_1(p \,\mathrm{drop}\, p_1 \,\hat{p}_1 \,\mathrm{lift}\, \pi) = p \,\mathrm{proj}_1(\pi) \qquad \mathrm{proj}_2(p \,\mathrm{drop}\, p_1 \,\hat{p}_1 \,\mathrm{lift}\, \pi) = p_1 \,\hat{p}_1 \,\mathrm{proj}_2(\pi)$$
$$\mathrm{proj}_1(\varepsilon) = \varepsilon \qquad\qquad\qquad\qquad\qquad \mathrm{proj}_2(\varepsilon) = \varepsilon.$$

With this, we set Q to be the set of pairs $(a, \pi) \in (\widetilde{\Sigma} \uplus \{\Box\}) \times \Pi$ such that $\mathrm{proj}_1(\pi)$ consists of pairwise distinct states wrt. $\sim$ and so does $\mathrm{proj}_2(\pi)$ (there might be states that occur in both
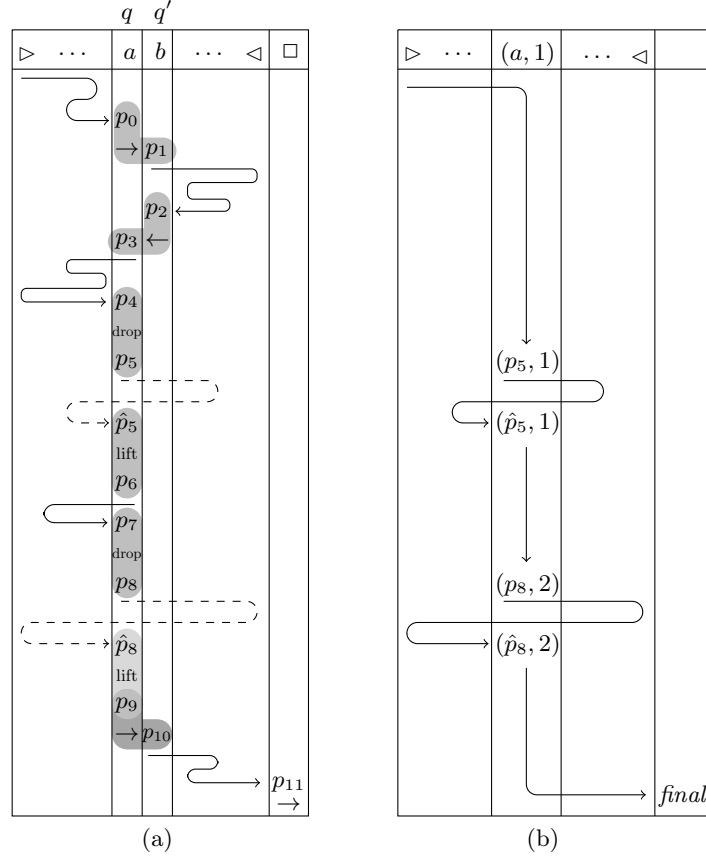
Fig. 5. (a) Runs of r-pwA $\mathcal{A}$ and r-nwA $\langle\mathcal{A}\rangle_\sim$; (b) run of $(r-1)$-pwA $\mathcal{A}_q$

$\mathrm{proj}_1(\pi)$ and $\mathrm{proj}_2(\pi)$). The letter $a \in \widetilde{\Sigma} \uplus \{\Box\}$ of a state $(a, \pi) \in Q$ will denote the symbol that is to be read next. Symbol $\Box$ means that there is no letter left so that the automaton is beyond the scope of $\triangleright w \triangleleft$ when w is the input word (we could chose, e.g., $\triangleleft$ instead of $\Box$, but the presentation will be clearer with a distinguished symbol). Next, we explain how the weight of the run segments of $\mathcal{A}$ with lifted pebble r is computed in $\langle\mathcal{A}\rangle_\sim$. Two neighboring states of $\langle\mathcal{A}\rangle_\sim$ need to complement each other, which can be checked locally by means of transitions. To determine a corresponding weight, we first count weights of those transitions that move from the current position z to $z+1$ or from $z+1$ to z. This is the reason why a state of $\langle\mathcal{A}\rangle_\sim$ also maintains the letter that is to be read next. In Figure 5(a), the 7 micro-transitions that we count in the step from q to q' are highlighted in gray. Assuming $q = (a, \pi_0)$ and $q' = (b, \pi_1)$, we obtain a value $\mathrm{weight}_{a,b}(\pi_0 \,|\, \pi_1)$ as the following product:

$$
\begin{aligned}
& \mu(p_0, a, \emptyset, \rightarrow, p_1) \\
\odot\; & \mu(p_2, b, \emptyset, \leftarrow, p_3) \\
\odot\; & \mu(p_4, a, \emptyset, \mathrm{drop}, p_5) \\
\odot\; & \mu(\hat{p}_5, a, \{r\}, \mathrm{lift}, p_6) \\
\odot\; & \mu(p_7, a, \emptyset, \mathrm{drop}, p_8) \\
\odot\; & \mu(\hat{p}_8, a, \{r\}, \mathrm{lift}, p_9) \\
\odot\; & \mu(p_9, a, \emptyset, \rightarrow, p_{10})
\end{aligned}
$$

15

The formal definition of $\text{weight}_{a,b}(\pi_0 \,|\, \pi_1) \in K$ is given in Figure 6 (there and in the following, whenever an argument is not specified, we suppose the result to be $\mathbf{0}$).

$$\text{weight}_{a,b}(\pi_0 \,|\, \pi_1) = \text{weight}^0_{a,b}(\pi_0 \,|\, \pi_1)$$

$$\text{weight}^0_{a,b}(p_0 \to \pi_0 \,|\, p_1\,\pi_1) = \begin{pmatrix} \mu(p_0, a, \emptyset, \to, p_1) \\ \odot\, \text{weight}^1_{a,b}(\pi_0 \,|\, p_1\,\pi_1) \end{pmatrix} \text{ if } \begin{cases} a \neq \vartriangleleft \\ \text{or } \pi_0 = \varepsilon, b = \square, \pi_1 = \to \end{cases}$$

$$\text{weight}^1_{a,b}(\pi_0 \,|\, p_1 \to \pi_1) \;\;= \text{weight}^1_{a,b}(\pi_0 \,|\, \pi_1) \text{ if } b \notin \{\vartriangleleft, \square\} \text{ or } \pi_0 = \pi_1 = \varepsilon$$

$$\text{weight}^0_{a,b}(p_0 \leftarrow \pi_0 \,|\, \pi_1) \;\;= \text{weight}^0_{a,b}(\pi_0 \,|\, \pi_1) \text{ if } a \neq \vartriangleright$$

$$\text{weight}^1_{a,b}(p_0\,\pi_0 \,|\, p_1 \leftarrow \pi_1) = \mu(p_1, b, \emptyset, \leftarrow, p_0) \odot \text{weight}^0_{a,b}(p_0\,\pi_0 \,|\, \pi_1) \text{ if } b \neq \square$$

$$\text{weight}^0_{a,b}(p_0 \,\text{drop}\, p\, \hat{p}\, \text{lift}\, p'_0\, \pi_0 \,|\, \pi_1) = \begin{pmatrix} \mu(p_0, a, \emptyset, \text{drop}, p) \\ \odot\, \mu(\hat{p}, a, \{r\}, \text{lift}, p'_0) \\ \odot\, \text{weight}^0_{a,b}(p'_0\, \pi_0 \,|\, \pi_1) \end{pmatrix} \text{ if } a \notin \{\vartriangleright, \vartriangleleft\}$$

$$\text{weight}^1_{a,b}(\pi_0 \,|\, p_1 \,\text{drop}\, p\, \hat{p}\, \text{lift}\, \pi_1) = \text{weight}^1_{a,b}(\pi_0 \,|\, \pi_1) \text{ if } b \notin \{\vartriangleleft, \square\}$$

$$\text{weight}^1_{a,b}(\varepsilon \,|\, \varepsilon) = \mathbf{1}$$

Fig. 6. Transition weights for $\langle\mathcal{A}\rangle_\sim$

We are now prepared to define the components ν, λ, γ of $\langle\mathcal{A}\rangle_\sim$. Let $(a, \pi) \in Q$. We set

$$\lambda(a, \pi) = \sum_{(\vartriangleright, \pi') \in Q} \text{weight}_{\vartriangleright, a}(\pi' \,|\, \pi)$$

$$\gamma(a, \pi) = \begin{cases} \sum_{p \in P} \text{weight}_{\vartriangleleft, \square}(\pi \,|\, p \to) & \text{if } a = \vartriangleleft \\ \mathbf{0} & \text{if } a \neq \vartriangleleft \end{cases}$$

Moreover, for $a \in \Sigma$ and states $q_0 = (a_0, \pi_0)$ and $q_1 = (a_1, \pi_1)$ from Q, we define

$$\nu(a)_{q_0, q_1} = \begin{cases} \text{weight}_{a_0, a_1}(\pi_0 \,|\, \pi_1) \odot \mathcal{B}_{q_0} & \text{if } a = a_0 \\ \mathbf{0} & \text{if } a \neq a_0 \end{cases}$$

Here, $\mathcal{B}_q$ is the constant $\mathbf{1}$ if $r = 0$. Otherwise, $\mathcal{B}_q$ is the $(r - 1)$-nwA over $\Delta = \Sigma \times \{0, 1\}$ that is determined inductively as follows: We first define an $(r - 1)$-pwA $\mathcal{A}_q$ over Δ together with an equivalence relation $\sim_q$. Then, we set $\mathcal{B}_q = \langle\mathcal{A}_q\rangle_{\sim_q}$, using the above translation. Note that the notation $k \odot \mathcal{B}_{q_0}$ in the definition of $\nu(a)_{q_0, q_1}$ is justified due to Remark 13.

Let us define the $(r - 1)$-pwA $\mathcal{A}_q = (P', \mu')$ over Δ as well as $\sim_q \subseteq P' \times P'$. Suppose $q = (a, \pi)$ and $\text{proj}_2(\pi) = p_1\, \hat{p}_1 \cdots p_n\, \hat{p}_n$. The behavior of $\mathcal{A}_q$ is split into $n + 2$ phases. In phase 0, it scans the input word from left to right until it finds the (unique) letter of the form $(a, 1)$ with $a \in \Sigma$. At that position, call it z, $\mathcal{A}_q$ enters a state $(p_1, 1)$ (the second component indicating the current phase). All these transitions are performed with weight $\mathbf{1}$. Then, $\mathcal{A}_q$ starts simulating $\mathcal{A}$, considering pebble r at position z. Back at position z in state $(\hat{p}_1, 1)$, weight-$\mathbf{1}$ transitions will allow $\mathcal{A}_q$ to enter the next phase, starting in $(p_2, 2)$ and again considering pebble r at position z. The simulation of $\mathcal{A}$ ends when $(\hat{p}_n, n)$ is reached in position z. In the final phase, $n + 1$, a sequel of weight-$\mathbf{1}$ transitions guides $\mathcal{A}_q$ to the end of the tape where it stops.

16

The relation $\sim_q$ will consider states (p, i) and (p', i') equivalent iff $p \sim p'$, i.e., it ignores the phase number. This explains the purpose of the equivalence relation: in order for the automaton $\mathcal{A}_q$ to simulate the dashed part of the run of $\mathcal{A}$, we use phase numbers, so that, during the simulation two different states (p, i) and (p, i') of $\mathcal{A}_q$ may correspond to the same original state p of $\mathcal{A}$. Now, only simple runs are considered to compute $[\![\mathcal{A}]\!]$. Therefore, for the simulation to be faithful, we want to rule out runs of $\mathcal{A}_q$ containing two configurations which only differ by the phase number, that is, containing two $\sim_q$-equivalent configurations. This is why we only keep $\sim_q$-simple runs.

A run of $\mathcal{A}_q$ is illustrated in Figure 5(b) (with $n = 2$). Note that, if $n = 0$, then $\mathcal{A}_q$ simply scans the word from left to right, outputting weight $\mathbf{1}$. For $n \geq 1$, $\mathcal{A}_q$ is formally given as follows: We set $P' = (P \times \{1, \ldots, n\}) \cup \{\text{init}, \text{final}\} \cup (\{\text{switch}\} \times \{1, \ldots, n\})$. The following transitions describe the part for simulating $\mathcal{A}$. For $p, p' \in P$, $i \in \{1, \ldots, n\}$, $\alpha \in \widetilde{\Delta} = \Delta \cup \{\triangleright, \triangleleft\}$, $J \subseteq 2^{r-1}$, and $d \in D$, let

$$\mu'((p, i), \alpha, J, d, (p', i)) = \begin{cases} \mu(p, a, J \cup \{r\}, d, p') & \text{if } \alpha = (a, 1) \text{ (with } a \in \Sigma) \\ \mu(p, a, J, d, p') & \text{if } \alpha = (a, 0) \text{ or } \alpha = a \in \{\triangleright, \triangleleft\}) \end{cases}$$

Next, the following transitions have weight $\mathbf{1}$ (and all non-specified transitions have weight $\mathbf{0}$):

$$
\begin{array}{ll}
(\text{init}, \triangleright, \emptyset, \rightarrow, \text{init}) & \\
(\text{init}, (a, 0), \emptyset, \rightarrow, \text{init}) & \text{for all } a \in \Sigma \\
(\text{init}, (a, 1), \emptyset, \rightarrow, (\text{switch}, 1)) & \text{for all } a \in \Sigma \\
(\text{switch}, i), \alpha, \emptyset, \leftarrow, (p_i, i)) & \text{for all } i \in \{1, \ldots, n\} \text{ and } \alpha \in \widetilde{\Delta} \\
((\hat{p}_i, i), (a, 1), \emptyset, \rightarrow, (\text{switch}, i+1)) & \text{for all } i \in \{1, \ldots, n-1\} \text{ and } a \in \Sigma \\
((\hat{p}_n, n), (a, 1), \emptyset, \rightarrow, \text{final}) & \text{for all } a \in \Sigma \\
(\text{final}, \alpha, \emptyset, \rightarrow, \text{final}) & \text{for all } \alpha \in \widetilde{\Delta}
\end{array}
$$

Finally, we let $\sim_q = \text{id}_R \cup \{((p, i), (p', i')) \in (P \times \{1, \ldots, n\})^2 \mid p \sim p'\}$ where $R = \{\text{init}, \text{final}\} \cup (\{\text{switch}\} \times \{1, \ldots, n\})$.

We will now argue for correctness of the construction of $\langle \mathcal{A} \rangle_\sim$, proceeding by induction on r. Fix a word $w = a_1 \cdots a_n \in \Sigma^+$ of length $n \geq 1$. We suppose that a $\sim$-simple run ρ of $\mathcal{A}$ on w consists of n^ρ steps and is of the form $\rho = \rho_0 \rho_1 \cdots \rho_{n^\rho}$. For $i \in \{0, \ldots, n+2\}$, let $q_i^\rho = (a_i, \pi_i^\rho) \in Q$ denote the state associated with position i according to the construction of $\langle \mathcal{A} \rangle_\sim$ which is illustrated in Figure 5(a). In particular, $a_0 = \triangleright$ and $\pi_0^\rho \in (P\{\rightarrow\})^+$, $a_{n+1} = \triangleleft$ and $\pi_{n+1}^\rho \in (P\{\leftarrow\})^* P\{\rightarrow\}$, $a_{n+2} = \square$ and $\pi_{n+2}^\rho \in P\{\rightarrow\}$. For each i, the sequence π_i^ρ is computed from ρ by the transducer of Figure 7. Note that there are no transitions from state 3 to state 2 since we have assumed that a drop is never immediately followed by a lift. Finally, for $1 \leq j \leq n^\rho$, let k_j^ρ denote the weight of the j-th step $\rho_{j-1}\rho_j$ in ρ.

Suppose $r = 0$. Then, we only use states 1 and 2 from the transducer of Figure 7. We call $\rho_{j-1}\rho_j$ an i-step if either $\rho_{j-1}\rho_j = (p, i, \varepsilon)(q, i+1, \varepsilon)$ or $\rho_{j-1}\rho_j = (p, i+1, \varepsilon)(q, i, \varepsilon)$. Then, we have

$$\text{weight}_{a_i, a_{i+1}}(\pi_i^\rho \mid \pi_{i+1}^\rho) = \bigodot_{1 \leq j \leq n^\rho \mid \rho_{j-1}\rho_j \ i\text{-step}} k_j^\rho \,. \tag{4}$$

17
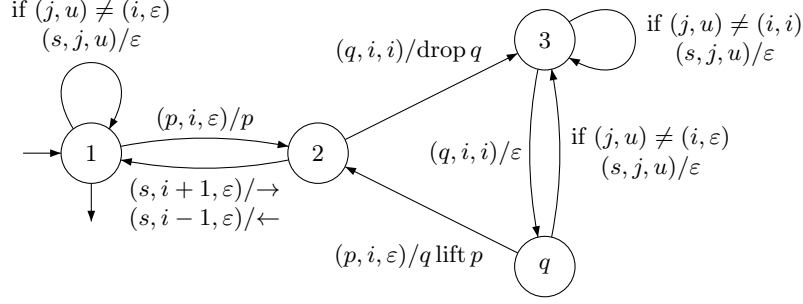
Fig. 7. Transducer computing π_i^ρ from the input ρ

Using the commutativity of $\mathbb{K}$, we deduce:

$$\llbracket \mathcal{A} \rrbracket_\sim(w) = \bigoplus_{\rho \ \sim\text{-simple}} \ \bigodot_{1 \le j \le n^\rho} k_j^\rho$$

$$= \bigoplus_{\rho \ \sim\text{-simple}} \ \bigodot_{0 \le i \le n+1} \text{weight}_{a_i,a_{i+1}}(\pi_i^\rho \mid \pi_{i+1}^\rho)$$

$$= \bigoplus_{q_1,\ldots,q_{n+1} \in Q} \lambda(q_1) \odot \nu(a_1)_{q_1,q_2} \odot \cdots \odot \nu(a_n)_{q_n,q_{n+1}} \odot \gamma(q_{n+1})$$

$$= \llbracket \langle \mathcal{A} \rangle_\sim \rrbracket(w)$$

Now suppose that $r \ge 1$ and that the transformation is correct for $(r-1)$-pwA. Here, i-steps are of the form $(p,i,\varepsilon)(q,i+1,\varepsilon)$ or $(p,i+1,\varepsilon)(q,i,\varepsilon)$ or $(p,i,\varepsilon)(q,i,i)$ (for a drop) or $(p,i,i)(q,i,\varepsilon)$ (for a lift) so that Equation (4) still holds. We call $\rho_{j-1}\rho_j$ an i-pebble-step if pebble r is placed on position i at configurations ρ_{j-1} and ρ_j, i.e., it is of the form $(p,\ell,ui)(p',\ell',u'i)$. Each step in ρ is either an i-pebble-step or an i-step for some $0 \le i \le n+1$. Then, using the commutativity of $\mathbb{K}$ we obtain:

$$\llbracket \mathcal{A} \rrbracket_\sim(w) = \bigoplus_{\rho \ \sim\text{-simple}} \ \bigodot_{1 \le j \le n^\rho} k_j^\rho$$

$$= \bigoplus_{\rho \ \sim\text{-simple}} \ \bigodot_{0 \le i \le n+1} \ \bigodot_{1 \le j \le n^\rho | \rho_{j-1}\rho_j \ i\text{-step}} k_j^\rho \odot \bigodot_{1 \le j \le n^\rho | \rho_{j-1}\rho_j \ i\text{-pebble-step}} k_j^\rho$$

$$= \bigoplus_{\rho \ \sim\text{-simple}} \ \bigodot_{0 \le i \le n+1} \text{weight}_{a_i,a_{i+1}}(\pi_i^\rho \mid \pi_{i+1}^\rho) \odot \llbracket \mathcal{A}_{q_i^\rho} \rrbracket_{\sim_{q_i^\rho}}(w,i) \ ^{(\dagger)}$$

$$= \bigoplus_{\rho \ \sim\text{-simple}} \ \bigodot_{0 \le i \le n+1} \text{weight}_{a_i,a_{i+1}}(\pi_i^\rho \mid \pi_{i+1}^\rho) \odot \llbracket \mathcal{B}_{q_i^\rho} \rrbracket(w,i) \ ^{(\dagger)}$$

$$= \bigoplus_{q_1,\ldots,q_{n+1} \in Q} \lambda(q_1) \odot \nu(a_1)_{q_1,q_2} \odot \cdots \odot \nu(a_n)_{q_n,q_{n+1}} \odot \gamma(q_{n+1})$$

$$= \llbracket \langle \mathcal{A} \rangle_\sim \rrbracket(w)$$

($\dagger$) with the convention that $\llbracket \mathcal{A}_{q_i^\rho} \rrbracket_{\sim_{q_i^\rho}}(w,i) = \mathbf{1} = \llbracket \mathcal{B}_{q_i^\rho} \rrbracket(w,i)$ when $i = 0$ or $i = n+1$. The induction hypothesis is used for the equalities $\llbracket \mathcal{A}_{q_i^\rho} \rrbracket_{\sim_{q_i^\rho}}(w,i) = \llbracket \mathcal{B}_{q_i^\rho} \rrbracket(w,i)$ for $1 \le i \le n$. $\qquad\square$

Proposition 18. *For every (possibly noncommutative) semiring $\mathbb{K}$, we have*

$$\mathbb{K}^{\mathrm{FO+BTC}^<}\langle\!\langle \Sigma^+ \rangle\!\rangle \subseteq \mathbb{K}^{\mathrm{pwA}}\langle\!\langle \Sigma^+ \rangle\!\rangle\,.$$

Proof. We proceed by induction on the structure of the formula and suppose that a valuation of free variables is given in terms of pebbles that are already placed on the word and cannot be lifted. Disjunction, conjunction, and first-order quantifications are easy to simulate. Indeed, we can proceed similarly to the proof of Proposition 12. To evaluate $[N\text{-}\mathrm{TC}^<_{xy}\varphi](z,t)$ for some formula $\varphi(x,y)$, we either evaluate $\varphi(x,y)$, with pebbles being placed on z and t such that $z \leq t \leq z + N$, or choose non-deterministically (and with weight $\mathbf{1}$) positions $z = z_0 < z_1 < \cdots < z_{n-1} < z_n = t$ with $n \geq 2$, using two additional pebbles, 2 and 1. We drop pebble 2 on position z_0 and pebble 1 on some guessed position z_1 with $z_0 < z_1 \leq \min(t, z_0 + N)$. We then run the subroutine to evaluate $\varphi(z_0, z_1)$. Next we move to position z_1 and lift pebble 1. We move left to position z_0 remembering the distance $z_1 - z_0$. We lift pebble 2 and move right to z_1 using the stored distance $z_1 - z_0$. We drop pebble 2 on z_1 and iterate this procedure until t is reached. $\square$

Remark 19. In the context of formal power series, one may call a sentence $\varphi \in \mathrm{FO+BTC}^<(\mathbb{K}, \Sigma)$ *satisfiable* if there is a word $w \in \Sigma^+$ such that $[\![\varphi]\!](w) \neq \mathbf{0}$. Over commutative positive semirings, satisfiability is decidable due to Theorem 16, which reduces the problem to non-emptiness of the support of a formal power series recognized by a pwA (or nwA). The latter problem, in turn, can be reduced to the decidable emptiness problem for classical pebble automata over the Boolean semiring. We leave it as an open problem to determine for which semirings the satisfiability problem is decidable.

Conclusion and perspectives

We have introduced pebble weighted automata and characterized their expressive power in terms of first-order logic with a bounded transitive closure operator. We do not know if allowing unbounded steps in the transitive closure leads beyond the power of pebble automata. Notice that Theorem 9 shows that allowing unbounded steps is harmless for bMSO-step formulas. It is also easy to show that such an unbounded transitive closure operator can be captured with *strong* pebble automata, *i.e.*, that can lift the last dropped pebble even when not scanning its position. A short-term perspective is to investigate whether strong pebble automata bring additional power. A first idea is to adapt the proof in the Boolean case [BSSS06] that they actually do not. As a mid-term perspective, we would like to adapt our results to tree languages.

References

Boj08.   M. Bojańczyk. Tree-walking automata. In *LATA'08*, volume 5196 of *LNCS*, pages 1–17. Springer, 2008.

BSSS06.   M. Bojańczyk, M. Samuelides, T. Schwentick, and L. Segoufin. Expressive power of pebble automata. In *Automata, Languages and Programming*, volume 4051 of *LNCS*, pages 157–168. Springer, 2006.

Büc60.   J. R. Büchi. Weak second-order arithmetic and finite automata. *Z. Math. Logik Grundlagen Math.*, 6:66–92, 1960.

C$^+$08.   H. Comon et al. *Tree Automata Techniques and Applications.* Available at http://tata.gforge.inria.fr/, 2008.

DG07.   M. Droste and P. Gastin. Weighted automata and weighted logics. *Theoretical Computer Science*, 380(1-2):69–86, 2007. Special issue of ICALP'05.

EH99.   J. Engelfriet and H. J. Hoogeboom. Tree-walking pebble automata. In *Jewels Are Forever, Contributions to Theoretical Computer Science in Honor of Arto Salomaa*, pages 72–83. Springer, 1999.

EH07.   J. Engelfriet and H. J. Hoogeboom. Automata with nested pebbles capture first-order logic with transitive closure. *Log. Meth. in Comput. Sci.*, 3, 2007.

Elg61.   C. C. Elgot. Decision problems of finite automata design and related arithmetics. *Trans. Amer. Math. Soc.*, 98:21–52, 1961.

KVD09.   W. Kuich, H. Vogler, and M. Droste, editors. *Handbook of Weighted Automata*. EATCS Monographs in Theoret. Comput. Sci. Springer, 2009.

NS00.   F. Neven and T. Schwentick. On the power of tree-walking automata. In Springer, editor, *ICALP*, volume 1853 of *LNCS*, pages 547–560, 2000.

Rab69.   M. O. Rabin. Decidability of second-order theories and automata on infinite trees. *Trans. AMS*, 141:1–35, 1969.

Sch61.   M. P. Schützenberger. On the definition of a family of automata. *Information and Control*, 4:245–270, 1961.

SS07.   M. Samuelides and L. Segoufin. Complexity of pebble tree-walking automata. In *FCT*, pages 458–469, 2007.

tCS10.   B. ten Cate and L. Segoufin. Transitive closure logic, nested tree walking automata, and XPath. *J. ACM*, 2010. To appear. Short version in PODS'08.

Tho82.   W. Thomas. Classifying regular events in symbolic logic. *Journal of Computer and System Sciences*, 25(3):360–376, 1982.

Tra61.   B. A. Trakhtenbrot. Finite automata and logic of monadic predicates. *Doklady Akademii Nauk SSSR*, 149:326–329, 1961. In Russian.