Jumping Automata over Infinite Words^{*}

Shaull Almagor^{1[0000-0001-9021-1175]} and Omer Yizhaq²

Technion, Israel shaull@technion.ac.il
Technion, Israel omeryi@campus.technion.ac.il

Abstract. Jumping automata are finite automata that read their input in a non-consecutive manner, disregarding the order of the letters in the word. We introduce and study jumping automata over infinite words. Unlike the setting of finite words, which has been well studied, for infinite words it is not clear how words can be reordered. To this end, we consider three semantics: automata that read the infinite word in some order so that no letter is overlooked, automata that can permute the word in windows of a given size k, and automata that can permute the word in windows of an existentially-quantified bound. We study expressiveness, closure properties and algorithmic properties of these models.

Keywords: Jumping Automata · Parikh Image · Infinite Words

1 Introduction

Traditional automata read their input sequentially. Indeed, this is the case for most state-based computational models. In some settings, however, the order of the input does not matter. For example, when the input represents available resources, and we only wish to reason about their *quantity*. From a more language-theoretic perspective, this amounts to looking at the *commutative closure* of the languages, a.k.a. their *Parikh image*. To capture this notion in a computation model, *Jumping Automata* were introduced in [20]. A jumping automaton may read its input in a non-sequential manner, jumping from letter to letter, as long as every letter is read exactly once. Several works have studied the algorithmic properties and expressive power of these automata [10,11,24,9,18].

One of the most exciting developments in automata and language theory has been the extension to the setting of infinite words [4,19,16], which has led to powerful tools in formal methods. The infinite-word setting is far more complicated than that of finite words, involving several acceptance conditions and intricate expressiveness relationships. Most notably perhaps, nondeterministic Büchi automata cannot be determinized, but can be determinized to Rabin automata [17,19,22].

In this work, we introduce jumping automata over infinite words. The first challenge is to find meaningful definitions for this model. Indeed, the intuition

 $^{^{\}star}$ This research was supported by the ISRAEL SCIENCE FOUNDATION (grant No. 989/22)

of having the reading head "jump" to points in the word is ill-suited for infinite words, since one can construct an infinite run even without reading every letter (possibly even skipping infinitely many letters) (see Remark 2).

To this end, we propose three semantics for modes of jumping over infinite words for an automaton \mathcal{A} .

- In the *jumping* semantics, a word w is accepted if \mathcal{A} accepts a permutation of it, i.e., a word w' that has the same number of occurrences of each letter as w for letters occurring finitely often, and the same set of letters that occur infinitely often.
- In the *k*-window jumping semantics, w is accepted if we can make \mathcal{A} accept it by permuting w within contiguous windows of some fixed size k.
- In the \exists -window jumping semantics, w is accepted if there exists some k such that we can make \mathcal{A} accept it by permuting w within contiguous windows of size k.

Example 1. Consider a Büchi automaton for the language $\{(ab)^{\omega}\}$ (see Fig. 3). Its jumping language is $\{w : w \text{ has infinitely many } a\text{'s and } b\text{'s }\}$. Its 3-window jumping language, for example, consists of words whose a's and b's can be rearranged to construct $(ab)^{\omega}$ in windows of size 3, such as $(aab \cdot bba)^{\omega}$. As for its \exists -window jumping language, it contains e.g., the word $(aaaabbbb)^{\omega}$, which is not in the 3-window language, but does not contain $aba^2b^2a^3b^3\cdots$, which is in the jumping language.

The definitions above capture different intuitive meanings for jumping: the first definition only looks at the Parikh image of the word, and corresponds to e.g., a setting where a word describes resources, with some resources being unbounded. The second (more restrictive) definition captures a setting where the word corresponds to e.g., a sequence of tasks to be handled, and we are allowed some bounded freedom in reordering them, and the third corresponds to a setting where we may introduce any finite delay to a task, but the overall delays we introduce must be bounded.

We study the expressiveness of these semantics, as well as closure properties and decision problems. Specifically, we show that languages in the jumping semantics are closed under union, intersection and complement. Surprisingly, we also show that automata admit determinization in this semantics, in contrast with standard Büchi automata. We further show that the complexity of decision problems on these automata coincide with their finite-word jumping counterparts. Technically, these results are attained by augmenting semilinear sets to accommodate ∞ , and by showing that jumping languages can be described in a canonical form with such sets.

In the k-window semantics, we show a correspondence with ω -regular languages, from which we deduce closure properties and solutions for decision problems. Finally, we show that the \exists -window semantics is strictly more expressive than the jumping semantics. **Paper organization** In Section 2 we recap some basic notions and define our jumping semantics. Section 3 is the bulk of the paper, where we study the jumping semantics. In Section 3.1 we introduce our variant of semilinear sets, and prove that it admits a canonical form and that it characterizes the jumping languages. Then, in Sections 3.2 and 3.3 we study closure properties and decision problems for jumping languages. In Sections 4 and 5 we study the *k*-window and \exists -window semantics, respectively. Finally, we conclude with future research in Section 6. Throughout the paper, our focus is on clean constructions and decidability. We thus defer complexity analyses to notes that follow each claim.

Related work Jumping automata were introduced in [20]. We remark that [20] contains some erroneous proofs (e.g., closure under intersection and complement, also pointed out in [11]). The works in [10,11] establish several expressiveness results on jumping automata, as well as some complexity results. In [24] many additional closure properties are established. An extension of jumping automata with a two-way tape was studied in [9].

Technically, since jumping automata correspond to the semilinear Parikh image/commutative closure [21], tools on semilinear sets are closely related. The works in [3,7,23] provide algorithmic results on semilinear sets, which we use extensively, and in [18] properties of semilinear sets are related to the state complexity of automata.

More broadly, automata and commutative closure also intersect in Parikh Automata [13,6,5,12], which read their input and accept if a certain Parikh image relating to the run belongs to a given semilinear set. In particular, [12] studies an extension of these automata to infinite words. Another related model is that of symmetric transducers – automata equipped with outputs, such that permutations in the input correspond to permutations in the output. These were studied in [2] in a jumping-flavour, and in [1] in a k-window flavour.

2 Preliminaries and Definitions

Automata A nondeterministic automaton is a 5-tuple $\mathcal{A} = \langle \Sigma, Q, \delta, Q_0, \alpha \rangle$ where Σ is a finite alphabet, Q is a finite set of states, $\delta : Q \times \Sigma \to 2^Q$ is a nondeterministic transition function, $Q_0 \subseteq Q$ is a set of initial states, and $\alpha \subseteq Q$ is a set of accepting states. When $|Q_0| = 1$ and $|\delta(q, \sigma)| = 1$ for every $q \in Q$ and $\sigma \in \Sigma$, we say that \mathcal{A} is deterministic.

We consider automata both over finite and over infinite words. We denote by Σ^* the set of finite words over Σ , and by Σ^{ω} the set of infinite words. For a word $w = \sigma_1, \sigma_2, \ldots$ (either finite or infinite), let $|w| \in \mathbb{N} \cup \{\infty\}$ be its length. A *run* of \mathcal{A} on w is a sequence $\rho = q_0, q_1, \ldots$ such that $q_0 \in Q_0$ and for every $0 \leq i < |w|$ it holds that $q_{i+1} \in \delta(q_i, w_{i+1})$ (naturally, a run is finite if w is finite, and infinite if w is infinite). For finite words, the run ρ is *accepting* if $q_{|w|} \in \alpha$. For infinite words we use the Büchi acceptance condition, whereby the run ρ is accepting if it visits α infinitely often. Formally, define $\operatorname{Inf}(\rho) = \{q \in Q \mid \forall i \in \mathbb{N} \; \exists j > i, \rho_j = q\}$,

then ρ is accepting if $\operatorname{Inf}(\rho) \cap \alpha \neq \emptyset$. A word w is accepted by \mathcal{A} if there exists an accepting run of \mathcal{A} on w.

The *language* of an automaton \mathcal{A} , denoted $\mathfrak{L}(\mathcal{A})$ is the set of words it accepts. We emphasize an automaton is over finite words by writing $\mathfrak{L}_{fin}(\mathcal{A})$.

Remark 1. There are other acceptance conditions for automata over infinite words, e.g., parity, Rabin, Streett, co-Büchi and Muller. As we show in Proposition 3, for our purposes it is enough to consider Büchi.

Parikh Images and Permutations Fix an alphabet Σ . We start by defining Parikh images and permutations for finite words. Consider a finite word $x \in \Sigma^*$. For each letter $\sigma \in \Sigma$ we denote by $\#_{\sigma}[x]$ the number of occurrences of σ in x. The *Parikh image* of x is then the vector $\Psi(x) = (\#_{\sigma}[x])_{\sigma \in \Sigma} \in \mathbb{N}^{\Sigma}$. We say that a word y is a *permutation* of x if $\Psi(y) = \Psi(x)$, in which case we write $x \sim y$ (clearly \sim is an equivalence relation). We extend the Parikh image to sets of words by $\Psi(L) = \{\Psi(w) \mid w \in L\}$.

We now extend the definitions to infinite words. Let $\mathbb{N} = \{0, 1, ...\}$ be the non-negative integers, and write $\mathbb{N}_{\infty} = \mathbb{N} \cup \{\infty\}$. Consider an infinite word $w \in \Sigma^{\omega}$. We extend the definition of Parikh image to infinite words in the natural way: $\Psi(w) = (\#_{\sigma}[w])_{\sigma \in \Sigma}$ where $\#_{\sigma}[w]$ is the number of occurrences of σ in w if it is finite, and is ∞ otherwise. Thus, $\Psi(w) \in \mathbb{N}_{\infty}^{\Sigma}$. Moreover, since wis infinite, at least one coordinate of $\Psi(w)$ is ∞ , since we often restrict ourselves to the Parikh images of infinite words, we denote by $\mathbb{M}^{\Sigma} = \mathbb{N}_{\infty}^{\Sigma} \setminus \mathbb{N}^{\Sigma}$ the set of vectors that have at least one ∞ coordinate. Note that $v \in \mathbb{M}^{\Sigma}$ if and only if vis the Parikh image of some infinite word over Σ .

For words $w, w' \in \Sigma^{\omega}$, we abuse notation slightly and write $w \sim w'$ if $\Psi(w) = \Psi(w')$, as well as refer to w' as a *permutation* of w. We now refine the notion of permutation by restricting permutations to finite "windows": let $k \in \mathbb{N}$, we say that w is a *k*-window permutation of w', and we write $w \sim_{k \boxplus} w'$ (the symbol \boxplus represents "window") if $w = x_1 \cdot x_2 \cdots$ and $w' = y_1 \cdot y_2 \cdots$ where for all $i \geq 1$ we have $|x_i| = |y_i| = k$ and $x_i \sim y_i$. That is, w' can be obtained from w by permuting contiguous disjoint windows of size k in w. Note that if $w \sim_{k \boxplus} w'$ then in particular $w \sim w'$, but the former is much more restrictive.

Semilinear Sets Let $\mathbb{N} = \{0, 1, ...\}$ be the natural numbers. For dimension $d \geq 1$, we denote vectors in \mathbb{N}^d in bold (e.g., $\boldsymbol{p} \in \mathbb{N}^d$). Consider a vector $\boldsymbol{b} \in \mathbb{N}^d$ and $P \subseteq \mathbb{N}^d$, the *linear set* generated by the *base* \boldsymbol{b} and *periods* P is

$$\operatorname{Lin}(\boldsymbol{b},P) = \{\boldsymbol{b} + \sum_{\boldsymbol{p} \in P} \lambda_p \boldsymbol{p} \mid \boldsymbol{b} \in \mathbb{N}^d \text{ and } \lambda_p \in \mathbb{N} \text{ for all } \boldsymbol{p} \in P\}$$

A semilinear set is then $\bigcup_{i \in I} \text{Lin}(\boldsymbol{b}_i, P_i)$ for a finite set I and pairs (\boldsymbol{b}_i, P_i) .

Semilinear sets are closely related to the Parikh image of regular languages (in that the Parikh image of a regular language is semilinear) [21]. We will cite specific results relating to this connection as we use them. **Jumping Automata** Consider an automaton $\mathcal{A} = \langle \Sigma, Q, \delta, Q_0, \alpha \rangle$. Over finite words, we view \mathcal{A} as a *jumping automaton* by letting it read its input in a non-sequential way, i.e. "jump" between letters as long as all letters are read exactly once. Formally, \mathcal{A} accepts a word w as a jumping automaton if it accepts some permutation of w. Thus, we define the *jumping language* of \mathcal{A} to be

$$\mathfrak{J}_{\mathrm{fin}}(\mathcal{A}) = \{ w \in \Sigma^* \mid \exists w' \sim w \text{ such that } w' \in \mathfrak{L}_{\mathrm{fin}}(\mathcal{A}) \}$$

We now turn to define jumping automata over infinite words. As discussed in Section 1, we consider three jumping semantics for \mathcal{A} , as follows:

Definition 1. Consider an automaton $\mathcal{A} = \langle \Sigma, Q, \delta, Q_0, \alpha \rangle$.

Remark 2. For jumping automata over finite words, the model is sometimes defined by allowing the transition function of the automaton to "jump" to any letter in the input and consume it [20]. For infinite words, this definition does not capture the (arguably) correct notion of jumping automata, since an automaton could skip letters without ever returning to them, e.g., for input $(ab)^{\omega}$ read only a^{ω} by jumping over all the b's. Under our definition, the latter is not allowed, since $a^{\omega} \not\sim (ab)^{\omega}$. Clearly, however, one can still obtain our "Jumping" definition by considering permutations of words, rather than a jumping reading head.

3 Jumping Languages

In this section we study the properties of $\mathfrak{J}(\mathcal{A})$ for an automaton \mathcal{A} . We start by characterizing $\mathfrak{J}(\mathcal{A})$ using an extension of semilinear sets.

3.1 Jumping Languages and Masked Semilinear Sets

Recall that $\mathbb{M}^{\Sigma} = \mathbb{N}_{\infty}^{\Sigma} \setminus \mathbb{N}^{\Sigma}$ and consider a vector $\boldsymbol{v} \in \mathbb{M}^{\Sigma}$. We separate the ∞ coordinates of \boldsymbol{v} from the finite ones by, intuitively, writing \boldsymbol{v} as a sum of a vector in \mathbb{N}^{Σ} and a vector from $\{0, \infty\}^{\Sigma} \setminus \{\mathbf{0}\}$. Formally, let $\boldsymbol{v} = \{0, \infty\}^{\Sigma} \setminus \{\mathbf{0}\}$ be the set of *masks*, namely vectors with entries 0 and ∞ that are not all 0, we refer to each $\mathfrak{m} \in \boldsymbol{v}$ as a *mask*. We denote by $\mathfrak{m}|_0 = \{\sigma \in \Sigma \mid \mathfrak{m}(\sigma) = 0\}$ and $\mathfrak{m}|_{\infty} = \{\sigma \in \Sigma \mid \mathfrak{m}(\sigma) = \infty\}$ the 0 and ∞ coordinates of \mathfrak{m} , respectively.

For $\mathbf{x} \in \mathbb{N}^{\Sigma}$ and $\mathfrak{m} \in \mathbb{V}$ let $\mathbf{x} \oplus \mathfrak{m} \in \mathbb{M}^{\Sigma}$ be the vector such that for all $\sigma \in \Sigma$ $(\mathbf{x}+\mathfrak{m})_{\sigma} = \mathbf{x}_{\sigma}$ if $\sigma \in \mathfrak{m}|_{0}$ and $(\mathbf{x}+\mathfrak{m})_{\sigma} = \infty$ if $\sigma \in \mathfrak{m}|_{\infty}$. Note that every $\mathbf{v} \in \mathbb{M}^{\Sigma}$ can be written as $\mathbf{x} + \mathfrak{m}$ where \mathbf{x} is obtained from \mathbf{v} be replacing ∞ with 0 (or indeed with any number), and having \mathfrak{m} match the ∞ coordinates of \mathbf{v} .

We now augment semilinear sets with masks. A masked semilinear set is a union of the form $S = \bigcup_{\mathfrak{m} \in \mathbb{T}_p} S_{\mathfrak{m}} + \mathfrak{m}$ where $S_{\mathfrak{m}}$ is a semilinear set for every \mathfrak{m} . We also interpret S as a subset of \mathbb{M}^{Σ} by interpreting addition as adding \mathfrak{m} to each vector in the set $S_{\mathfrak{m}}$. Note that the union above is disjoint, since adding distinct masks always results in distinct vectors.

Consider a mask $\mathfrak{m} \in \mathbb{S}$. Two vectors $\boldsymbol{u}, \boldsymbol{v} \in \mathbb{N}^{\Sigma}$ are called \mathfrak{m} -equivalent if $\boldsymbol{u} + \mathfrak{m} = \boldsymbol{v} + \mathfrak{m}$ (i.e., if they agree on $\mathfrak{m}|_0$). We say that a semilinear set $R \subseteq \mathbb{N}^{\Sigma}$ is \mathfrak{m} -oblivious if for every \mathfrak{m} -equivalent vectors $\boldsymbol{u}, \boldsymbol{v}$ we have $\boldsymbol{u} \in R \iff \boldsymbol{v} \in R$. Intuitively, an \mathfrak{m} -oblivious set does not "look" at $\mathfrak{m}|_{\infty}$.

We say that the masked semilinear set S above is *oblivious* if every S_m is **m**-oblivious. Note that the property of being oblivious refers to a specific representation of S with semilinear sets S_m for every mask \mathfrak{m} . In a way, it is natural for a masked semilinear set to be oblivious, since semantically, adding \mathfrak{m} to a vector in S_m already ignores the ∞ coordinates of \mathfrak{m} , so S_m should not "care" about them. Moreover, if a set S_m is \mathfrak{m} -oblivious, then in each of its linear components we can, intuitively, partition its period vectors to two types: those that have 0 in all the masked coordinates, and the remaining vectors that allow attaining any value in the masked coordinates. We refer to this as a *canonical* \mathfrak{m} -oblivious representation, as follows.

Definition 2. A semilinear set $R \subseteq \mathbb{N}^{\Sigma}$ is in canonical m-oblivious form for a mask \mathfrak{m} if $R = \bigcup_{i=1}^{k} \operatorname{Lin}(\mathbf{b}_{i}, P_{i})$ such that for every $1 \leq i \leq k$ we have $P_{i} = Q_{i} \cup E_{\mathfrak{m}}$ with the following conditions:

- $\mathbf{b}_{\mathbf{i}}(\sigma) = 0 \text{ for every } \sigma \in \mathfrak{m}|_{\infty}.$
- $\mathbf{p}(\sigma) = 0 \text{ for every } p \in Q_i \text{ and } \sigma \in \mathfrak{m}|_{\infty}.$
- $E_{\mathfrak{m}} = \{ \boldsymbol{e}_{\boldsymbol{\sigma}} \mid \boldsymbol{\sigma} \in \mathfrak{m}|_{\infty} \} \text{ with } \boldsymbol{e}_{\boldsymbol{\sigma}}(\tau) = 1 \text{ if } \boldsymbol{\sigma} = \tau \text{ and } 0 \text{ otherwise.}$

We now show that every masked semilinear set can be translated to an equivalent one in canonical oblivious form (see example in Fig. 1).

$$\operatorname{Lin}\left(\begin{pmatrix}1\\0\\\mathbf{9}\\\mathbf{4}\end{pmatrix}, \left\{\begin{pmatrix}1\\2\\\mathbf{5}\\\mathbf{7}\end{pmatrix}, \begin{pmatrix}1\\0\\\mathbf{1}\\\mathbf{3}\end{pmatrix}\right\}\right) + \begin{pmatrix}0\\0\\\infty\\\infty\\\infty\end{pmatrix} \quad \rightsquigarrow \quad \operatorname{Lin}\left(\begin{pmatrix}1\\0\\0\\0\end{pmatrix}, \left\{\begin{pmatrix}1\\2\\0\\0\end{pmatrix}, \begin{pmatrix}1\\0\\0\\0\end{pmatrix}, \begin{pmatrix}0\\0\\1\\0\end{pmatrix}, \begin{pmatrix}0\\0\\1\\0\end{pmatrix}\right\}\right) + \begin{pmatrix}0\\0\\\infty\\\infty\end{pmatrix}$$

Fig. 1. On the left, a masked semilinear set (actually linear), and on the right an equivalent oblivious canonical form. The bold numbers on the left get masked away, so are replaced by 0.

Lemma 1. Consider a masked semilinear set $S = \bigcup_{\mathfrak{m} \in \mathbb{Z}} S_{\mathfrak{m}} + \mathfrak{m}$. Then there exists an oblivious masked semilinear set $T = \bigcup_{\mathfrak{m} \in \mathbb{Z}} T_{\mathfrak{m}} + \mathfrak{m}$ that represents the same subset of \mathbb{M}^{Σ} , and every $T_{\mathfrak{m}}$ is in canonical \mathfrak{m} -oblivious form.

Proof. Intuitively, we obtain the oblivious representation by adding to each $S_{\mathfrak{m}}$ vectors where the ∞ -coordinates of \mathfrak{m} can take any value.

Formally, for every mask $\mathfrak{m} \in \mathbb{S}$ define

$$T_{\mathfrak{m}} = \{ \boldsymbol{v} \in \mathbb{N}^{\Sigma} \mid \exists \boldsymbol{u} \in S_{\mathfrak{m}} \text{ such that } \boldsymbol{u} + \mathfrak{m} = \boldsymbol{v} + \mathfrak{m} \}$$

That is, $T_{\mathfrak{m}}$ is obtained from $S_{\mathfrak{m}}$ by including every vector that is \mathfrak{m} -equivalent to some vector in $S_{\mathfrak{m}}$.

By construction, we have that $T_{\mathfrak{m}}$ is \mathfrak{m} -oblivious. Moreover, it is easy to see that $S_{\mathfrak{m}} + \mathfrak{m} = T_{\mathfrak{m}} + \mathfrak{m}$. Indeed, $S_{\mathfrak{m}} \subseteq T_{\mathfrak{m}}$, thus giving one containment, and for every $\boldsymbol{x} \in T_{\mathfrak{m}} + \mathfrak{m}$, write $\boldsymbol{x} = \boldsymbol{v} + \mathfrak{m}$ for some $\boldsymbol{v} \in T_{\mathfrak{m}}$, then there exists some $\boldsymbol{u} \in S_{\mathfrak{m}}$ with $\boldsymbol{v} + \mathfrak{m} = \boldsymbol{u} + \mathfrak{m} \in S_{\mathfrak{m}} + \mathfrak{m}$.

It remains to show that $T_{\mathfrak{m}}$ is semilinear and represent it in canonical form. To this end, for every vector $\mathbf{p} \in \mathbb{N}^{\Sigma}$, define $\mathbf{p}|_{\mathfrak{m}}$ by $\mathbf{p}|_{\mathfrak{m}}(\sigma) = \mathbf{p}(\sigma)$ if $\sigma \in \mathfrak{m}|_0$ and $\mathbf{p}|_{\mathfrak{m}}(\sigma) = 0$ if $\sigma \in \mathfrak{m}|_{\infty}$. That is, we replace all the ∞ coordinates of \mathfrak{m} in \mathbf{p} by 0. We lift this to sets of vectors: for a set P, let $P|_{\mathfrak{m}} = \{\mathbf{p}|_{\mathfrak{m}} \mid \mathbf{p} \in P\}$.

Now, write $S_{\mathfrak{m}} = \bigcup_{i \in I} \operatorname{Lin}(\boldsymbol{b}_i, P_i)$, we claim the following:

$$T_{\mathfrak{m}} = \bigcup_{i \in I} \operatorname{Lin}(\boldsymbol{b_i}|_{\mathfrak{m}}, P_i|_{\mathfrak{m}} \cup E_{\mathfrak{m}})$$

That is, $T_{\mathfrak{m}}$ is obtained by zeroing the ∞ coordinates of \mathfrak{m} in $S_{\mathfrak{m}}$, and then arbitrarily adding numbers in these coordinates, using $e_{\sigma} \in E_{\mathfrak{m}}$ (as per Definition 2).

Let $\boldsymbol{v} \in T_{\mathfrak{m}}$, then there exists $\boldsymbol{u} \in S_{\mathfrak{m}}$ such that $\boldsymbol{v} + \mathfrak{m} = \boldsymbol{u} + \mathfrak{m}$. That is, $\boldsymbol{u}(\sigma) = \boldsymbol{v}(\sigma)$ for all $\sigma \in \mathfrak{m}|_0$. Since $\boldsymbol{u} \in S_{\mathfrak{m}}$, we can write $\boldsymbol{u} = \boldsymbol{b}_i + \lambda_1 p_1 + \ldots + \lambda_k p_k$ for some $i \in I, p_1, \ldots, p_k \in P_i$ and $\lambda_1, \ldots, \lambda_k \in \mathbb{N}$. It then follows that

$$\boldsymbol{v} = \boldsymbol{b}_{\boldsymbol{i}}|_{\mathfrak{m}} + \lambda_1 p_1|_{\mathfrak{m}} + \ldots + \lambda_k p_k|_{\mathfrak{m}} + \sum_{\sigma \in \mathfrak{m}|_{\infty}} \boldsymbol{v}(\sigma) \boldsymbol{e}_{\sigma}$$

and the latter form is in $\operatorname{Lin}(\boldsymbol{b_i}|_{\mathfrak{m}}, P_i|_{\mathfrak{m}} \cup E_{\mathfrak{m}})$.

Conversely, let $\boldsymbol{v} \in \operatorname{Lin}(\boldsymbol{b}_{\boldsymbol{i}}|_{\mathfrak{m}}, P_{\boldsymbol{i}}|_{\mathfrak{m}} \cup E_{\mathfrak{m}})$, and write

$$\boldsymbol{v} = \boldsymbol{b}_{\boldsymbol{i}}|_{\mathfrak{m}} + \lambda_1 p_1|_{\mathfrak{m}} + \ldots + \lambda_k p_k|_{\mathfrak{m}} + \sum_{\sigma \in \mathfrak{m}|_{\infty}} \beta_{\sigma} \boldsymbol{e}_{\sigma}$$

Define $\boldsymbol{u} = \boldsymbol{b}_i + \lambda_1 p_1 + \ldots + \lambda_k p_k$, then $\boldsymbol{u} \in S_{\mathfrak{m}}$ and $\boldsymbol{u} + \mathfrak{m} = \boldsymbol{v} + \mathfrak{m}$ (since the only coordinates where \boldsymbol{v} differs from \boldsymbol{u} are those in $\mathfrak{m}|_{\infty}$), so $\boldsymbol{v} \in T_{\mathfrak{m}}$, and we are done.

We conclude by defining $T = \bigcup_{\mathfrak{m} \in \mathbb{Z}} T_{\mathfrak{m}} + \mathfrak{m}$, which is an oblivious masked semilinear set in canonical form, as required.

Complexity Analysis 1 (of Lemma 1). The description of T involves introducing at most $|\Sigma|$ vectors to the periods of each semilinear set, and changing some entries to 0 in the existing vectors. Thus, the description of T is of polynomial size in that of S.

Our main result in this section is that the Parikh images of jumping languages coincide with masked semilinear sets. We prove this in the following lemmas.

Lemma 2. Let \mathcal{A} be an automaton, then $\Psi(\mathfrak{J}(\mathcal{A}))$ is a masked semilinear set. Moreover, we can effectively compute a representation of it from that of \mathcal{A} .

Proof. Consider $\mathcal{A} = \langle \Sigma, Q, \delta, Q_0, \alpha \rangle$. By Definition 1 we have that $\Psi(\mathfrak{J}(\mathcal{A})) = \Psi(\mathfrak{L}(\mathcal{A}))$. Indeed, both sets contain exactly the Parikh images of words in $\mathfrak{L}(\mathcal{A})$. Recall (or see Appendix A.1) that every ω -regular language can be (effectively) written as a union of the form $\mathfrak{L}(\mathcal{A}) = \bigcup_{i=1}^m S_i \cdot T_i^{\omega}$ where $S_i, T_i \subseteq \Sigma^*$ are regular languages over finite words.

Intuitively, we will show that $\Psi(T_i^{\omega})$ can be separated to letters that are seen finitely often, and those that are seen infinitely often, where the latter will induce the mask. To this end, for every $\emptyset \neq \Gamma \subseteq \Sigma^*$ define $\mathfrak{m}_{\Gamma} \in \mathfrak{s}$ by setting $\mathfrak{m}_{\Gamma}(\sigma) = \infty$ if $\sigma \in \Gamma$ and $\mathfrak{m}_{\Gamma}(\sigma) = 0$ if $\sigma \notin \Gamma$. Now, for every regular language T_i in the union above define

$$I(T_i) = \{ \mathfrak{m}_{\Gamma} \mid \forall \sigma \in \Gamma \; \exists w \in T_i \cap \Gamma^* \text{ s.t. } \Psi(w)(\sigma) > 0 \}$$

That is, $I(T_i)$ is the set of masks \mathfrak{m}_{Γ} such that every letter in Γ occurs in some word in T_i that contains only letters from Γ . Intuitively, $\mathfrak{m}_{\Gamma} \in I(T_i)$ can be attained by an infinite concatenation of words from T_i by covering all letters in Γ whilst not using letters outside of Γ .

We now claim that for every $1 \leq i \leq m$ it holds that $\Psi(S_i \cdot T_i^{\omega}) = \Psi(S_i \cdot T_i^*) + I(T_i)$. To reduce clutter, we drop the subscript *i* for this proof. Consider $\boldsymbol{v} \in \Psi(S \cdot T^{\omega})$, then there exists a word $w = x_1 x_2 \cdots$ such that $\Psi(w) = \boldsymbol{v}$ with $x_1 \in S$ and $x_j \in T$ for all j > 1. Let Γ be the set of letters that occur infinitely often in w, and let $N_0 \geq 2$ be such that for all $n \geq N_0$ we have $x_n \in \Gamma^*$. Observe that $\mathfrak{m}_{\Gamma} \in I(T)$ since for every $\sigma \in \Gamma$ there exists some word $x_j \in T_i \cap \Gamma^*$ (for $j \geq N_0$) such that σ occurs in x_j . Thus, $\Psi(x_{N_0}x_{N_0+1}\cdots) = \mathfrak{m}_{\Gamma}$. Moreover, $x_1 \cdots x_{N_0-1} \in S \cdot T^*$, so $\boldsymbol{v} = \Psi(w) = \Psi(x_1 \cdots x_{N_0-1}) + \mathfrak{m}_{\Gamma} \in \Psi(S \cdot T^*) + I(T)$.

Conversely, let $\boldsymbol{v} \in \Psi(S \cdot T^*) + I(T)$, then there exist $y \in S \cdot T^*$ and $\mathfrak{m}_{\Gamma} \in I(T)$ for some $\Gamma \subseteq \Sigma$ such that $\boldsymbol{v} = \Psi(y) + \mathfrak{m}_{\Gamma}$. Since $\mathfrak{m}_{\Gamma} \in I(T)$, there exists a set of words $\{z_1, \ldots, z_r\} \subseteq T \cap \Gamma^*$ such that every $\sigma \in \Gamma$ occurs in some word in this set. We thus have $\Psi((z_1 \cdots z_r)^{\omega}) = \mathfrak{m}_{\Gamma}$ and $(z_1 \cdots z_r)^{\omega} \in T^{\omega}$, so $y \cdot (z_1 \cdots z_r)^{\omega} \in S \cdot T^{\omega}$ and we have $\boldsymbol{v} = \Psi(y) + \mathfrak{m}_{\Gamma} = \Psi(y \cdot (z_1 \cdots z_r)^{\omega}) \in \Psi(S \cdot T^{\omega})$, concluding our claim.

It follows that $\Psi(\mathfrak{J}(\mathcal{A})) = \bigcup_{i=1}^{k} \Psi(S_i \cdot T_i^*) + I(T_i)$. We can now rearrange the sum by masks instead of by index *i*: for every mask $\mathfrak{m} \in \mathfrak{S}$ write $S_{\mathfrak{m}} = \bigcup_{i:\mathfrak{m} \in I(T_i)} \Psi(S_i \cdot T_i^*)$ (note that $S_{\mathfrak{m}}$ could be empty). Since $S_i \cdot T_i^*$ is a regular language, then by Parikh's theorem [21] its Parikh image is semilinear, so $S_{\mathfrak{m}}$ is semilinear. Thus, $\Psi(\mathfrak{J}(\mathcal{A})) = \bigcup_{\mathfrak{m} \in \mathfrak{S}} S_{\mathfrak{m}} + \mathfrak{m}$ is a masked semilinear set. \Box

Complexity Analysis 2 (of Lemma 2). Writing $\mathfrak{L}(\mathcal{A})$ as a union involves only a polynomial blowup (see Appendix A.1). Then, we split the resulting expression to a union over $2^{|\mathcal{L}|}$ masks. Within the union, we convert a nondeterministic automaton for $S_i \cdot T_i^*$ to a semilinear set. By [23, Theorem 4.1] (also [15]), the resulting semilinear set has description size polynomial in the number of states n of \mathcal{A} and singly-exponential in $|\mathcal{L}|$. Moreover, the translation can be computed in time $2^{O(|\mathcal{L}|^2 \log(n|\mathcal{L}|))}$.

For the converse direction, we present a stronger result, namely that we can construct a *deterministic* automaton to capture a masked semilinear set.

Lemma 3. Consider a masked semilinear set S, then there exists a deterministic automaton \mathcal{A} such that $\Psi(\mathfrak{J}(\mathcal{A})) = S$.

Proof. By Lemma 1, we can assume $S = \bigcup_{\mathfrak{m} \in \mathbb{Z}} S_{\mathfrak{m}} + \mathfrak{m}$ and that every $S_{\mathfrak{m}}$ is in canonical \mathfrak{m} -oblivious form. Let $\mathfrak{m} \in \mathbb{Z}$ and write $S_{\mathfrak{m}} = \bigcup_{i=1}^{k} \operatorname{Lin}(\mathbf{b}_{i}, P_{i} \cup E_{\mathfrak{m}})$ as per Definition 2. Consider a linear set $L = \operatorname{Lin}(\mathbf{b}, P \cup E_{\mathfrak{m}})$ in the union above, and we omit the subscript *i* for brevity. We start by constructing a deterministic automaton \mathcal{D} such that $\Psi(\mathfrak{J}_{\mathrm{fn}}(\mathcal{D})) = L$. To this end, let $w_{\mathbf{b}} \in \Sigma^{*}$ be a word such that $\Psi(w_{\mathbf{b}}) = \mathbf{b}$ and similarly for every $\mathbf{p} \in P$ let $w_{\mathbf{p}} \in \Sigma^{*}$ such that $\Psi(w_{\mathbf{p}}) = \mathbf{p}$. Note that $\Psi(\sigma) = \mathbf{e}_{\sigma}$ for every $\sigma \in \mathfrak{m}|_{\infty}$.

Let \mathcal{D}' be a deterministic automaton for the regular expression $w_b \cdot (w_{p_1} + \ldots + w_{p_n})^*$ where $P = \{p_1, \ldots, p_n\}$. Next, let $w_E = \sigma_1 \cdots \sigma_k$ be a word obtained by concatenating all the letters in $\mathfrak{m}|_{\infty}$ in some order. We obtain \mathcal{D} from \mathcal{D}' by connecting every accepting state of \mathcal{D}' , upon reading σ_1 , to a cycle that allows reading w_E^{ω} . The accepting states of \mathcal{D} are those on the w_E cycle. Crucially, the transition from \mathcal{D}' upon reading σ_1 retains determinism, since P is in canonical form, and therefore $p(\sigma_1) = 0$ for every $p \in P$. That is, the letter σ_1 is not seen in any transition of \mathcal{D}' , allowing us to use it in the construction of \mathcal{D} . The construction is demonstrated in Fig. 2.



Fig. 2. The automaton \mathcal{D} for the linear set from Fig. 1 (over $\Sigma = \{a, b, c, d\}$), with the representative words $w_b = a$ and $\{bab, a, c, d\}$ for the period. The blue (dashed) parts are the addition of w_E and change of accepting states to obtain \mathcal{D} from \mathcal{D}' .

By construction, we have that $\mathfrak{L}(\mathcal{D}) = w_{\boldsymbol{b}} \cdot (w_{\boldsymbol{p_1}} + \ldots + w_{\boldsymbol{p_n}})^* \cdot w_E^{\omega}$. Since $\Psi(w_E^{\omega}) = \mathfrak{m}$, we now have

$$\Psi(\mathfrak{J}(\mathcal{D})) = \Psi(\mathfrak{L}(\mathcal{D})) = \Psi(w_{\boldsymbol{b}} \cdot (w_{\boldsymbol{p_1}} + \ldots + w_{\boldsymbol{p_n}})^* \cdot w_E^{\omega}) = \operatorname{Lin}(\boldsymbol{b}, P) + \mathfrak{m} = L + \mathfrak{m}$$

where the last equality is because adding \mathfrak{m} to a vector in L masks any addition of vectors in $E_{\mathfrak{m}}$ by ∞ .

Next, we observe that for two deterministic automata $\mathcal{D}_1, \mathcal{D}_2$, we can construct a deterministic automaton \mathcal{D}_3 such that $\mathfrak{J}(\mathcal{D}_3) = \mathfrak{J}(\mathcal{D}_1) \cup \mathfrak{J}(\mathcal{D}_2)$. Indeed, the standard product construction (where the accepting states are pairs of states where either \mathcal{D}_1 or \mathcal{D}_2 accept) preserves determinism. Thus, we can take products to first capture $S_{\mathfrak{m}} = \bigcup_{i=1}^k \operatorname{Lin}(\mathbf{b}_i, P_i \cup E_{\mathfrak{m}})$ and then $S = \bigcup_{\mathfrak{m} \in \mathbb{T}} S_{\mathfrak{m}} + \mathfrak{m}$, as desired.

Complexity Analysis 3 (of Lemma 3). We can naively construct an automaton for the regular expression $w_{\mathbf{b}} \cdot (w_{\mathbf{p_1}} + \ldots + w_{\mathbf{p_n}})^*$ of size $\|\text{Lin}(\mathbf{b}, P)\| = \|\mathbf{b}\| + \sum_{i=1}^n \|\mathbf{p_i}\|$, where e.g., $\|\mathbf{b}\|$ is the sum of entries in \mathbf{b} (i.e., unary representation). Then, determinization can yield a DFA of size singly exponential. Then, we take unions by the product construction, where the number of copies corresponds to the number of linear sets in S, and then to $2^{|\Sigma|} - 1$ many masks. This results in a DFA of size singly exponential in the size of S and doubly-exponential in $|\Sigma|$ (assuming S is represented in unary). Finally, adding the cycle for the mask adds at most $2^{O(|\Sigma|)}$ states (both to a deterministic or nondeterministic automaton).

Combining Lemmas 2 and 3 we have the following.

Theorem 4. A set S is a masked semilinear set if and only if there exists an automaton \mathcal{A} such that $S = \Psi(\mathfrak{J}(\mathcal{A}))$.

3.2 Jumping Languages – Closure Properties

Using the characterizations obtained in Section 3.1, we can now obtain several closure properties of jumping languages. First, we remark that jumping languages are clearly closed under union, by taking the union of the automata and nondeterministically choosing which one to start at. We proceed to tackle intersection and complementation.

Note that the standard intersection construction of Büchi automata does not capture the intersection of jumping languages, as demonstrated in Fig. 3.



Fig. 3. The automata \mathcal{A} and \mathcal{B} satisfy $\mathfrak{L}(\mathcal{A}) = \{(ab)^{\omega}\}\)$ and $\mathfrak{L}(\mathcal{B}) = \{(ba)^{\omega}\}\)$. Thus, $\mathfrak{J}(\mathcal{A}) = \mathfrak{J}(\mathcal{B}) = \{w \in \Sigma^{\omega} \mid w \text{ has infinitely many } a \text{'s and } b \text{'s}\}\)$. However, if we start by taking the standard intersection of \mathcal{A} and \mathcal{B} , we would end up with the empty language.

Proposition 1. Let \mathcal{A} and \mathcal{B} be automata, then we can effectively construct an automaton \mathcal{C} such that $\mathfrak{J}(\mathcal{C}) = \mathfrak{J}(\mathcal{A}) \cap \mathfrak{J}(\mathcal{B})$.

Proof. Consider a word $w \in \Sigma^{\omega}$, and observe that $w \in \mathfrak{J}(\mathcal{A}) \cap \mathfrak{J}(\mathcal{B})$ if and only if $\Psi(w) \in \Psi(\mathfrak{J}(\mathcal{A})) \cap \Psi(\mathfrak{J}(\mathcal{B}))$. Thus, it suffices to prove that there exists automaton \mathcal{C} such that $\Psi(\mathfrak{J}(\mathcal{C})) = \Psi(\mathfrak{J}(\mathcal{A})) \cap \Psi(\mathfrak{J}(\mathcal{B}))$. By Theorem 4 we can write $\Psi(\mathfrak{J}(\mathcal{A})) = \bigcup_{\mathfrak{m} \in \mathbb{Z}} S_{\mathfrak{m}} + \mathfrak{m}$ and $\Psi(\mathfrak{J}(\mathcal{B})) = \bigcup_{\mathfrak{m} \in \mathbb{Z}} T_{\mathfrak{m}} + \mathfrak{m}$. Furthermore, by Lemma 1 we can assume that these are oblivious masked semilinear sets.

We claim that $\Psi(\mathfrak{J}(\mathcal{A})) \cap \Psi(\mathfrak{J}(\mathcal{B})) = \bigcup_{\mathfrak{m} \in \mathbb{N}} S_{\mathfrak{m}} \cap T_{\mathfrak{m}} + \mathfrak{m}$. That is, we can take intersection by intersecting each pair of semilinear sets, while keeping the masks. Intuitively, this holds because every word uniquely determines the mask it can use to join the union, so intersecting the unions amounts (using obliviousness)

11

to intersecting the set of finite-valued vectors corresponding to the finite parts of the Parikh images of words in the language.

Indeed, let $\boldsymbol{x} \in \bigcup_{\mathfrak{m} \in \mathfrak{m}} S_{\mathfrak{m}} \cap T_{\mathfrak{m}} + \mathfrak{m}$, then there exists $\mathfrak{m} \in \mathfrak{m}$ and $\boldsymbol{v} \in S_{\mathfrak{m}} \cap T_{\mathfrak{m}}$ such that $\boldsymbol{x} = \boldsymbol{v} + \mathfrak{m}$. In particular, $\boldsymbol{v} + \mathfrak{m} \in \Psi(\mathfrak{J}(\mathcal{A}))$ and $\boldsymbol{v} + \mathfrak{m} \in \Psi(\mathfrak{J}(\mathcal{B}))$, so $\boldsymbol{x} = \boldsymbol{v} + \mathfrak{m} \in \Psi(\mathfrak{J}(\mathcal{A})) \cap \Psi(\mathfrak{J}(\mathcal{B}))$.

Conversely, let $\boldsymbol{x} \in \Psi(\mathfrak{J}(\mathcal{A})) \cap \Psi(\mathfrak{J}(\mathcal{B}))$. Recall that \boldsymbol{x} uniquely defines a mask \mathfrak{m} , so there exist $\boldsymbol{u} \in S_{\mathfrak{m}}, \boldsymbol{v} \in T_{\mathfrak{m}}$ such that $\boldsymbol{x} = \boldsymbol{u} + \mathfrak{m} = \boldsymbol{v} + \mathfrak{m}$. It follows that \boldsymbol{u} and \boldsymbol{v} are \mathfrak{m} -equivalent. Since $S_{\mathfrak{m}}$ and $T_{\mathfrak{m}}$ are \mathfrak{m} -oblivious, we also have that $\boldsymbol{v} \in S_{\mathfrak{m}}$ and $\boldsymbol{u} \in T_{\mathfrak{m}}$. In particular, $\boldsymbol{v} \in S_{\mathfrak{m}} \cap T_{\mathfrak{m}}$ so $\boldsymbol{x} = \boldsymbol{v} + \mathfrak{m} \in \bigcup_{\mathfrak{m} \in \mathfrak{T}_{\mathfrak{m}}} S_{\mathfrak{m}} \cap T_{\mathfrak{m}} + \mathfrak{m}$.

Finally, semilinear sets are closed under intersection [3,7]. Thus, $S_{\mathfrak{m}} \cap T_{\mathfrak{m}}$ is semilinear for every \mathfrak{m} , so $\bigcup_{\mathfrak{m} \in \mathbb{Z}} S_{\mathfrak{m}} \cap T_{\mathfrak{m}} + \mathfrak{m}$ is a semilinear masked set and from Lemma 3 there exists an automaton \mathcal{C} such that $\mathfrak{J}(\mathcal{C}) = \mathfrak{J}(\mathcal{A}) \cap (\mathfrak{J}(\mathcal{B}))$. \Box

Complexity Analysis 5 (of Proposition 1). The description size of the intersection of semilinear sets is polynomial in the description size of the entries of the vectors and the size of the union, and singly exponential in $|\Sigma|$ [3,7]. Since the blowup in Lemmas 1 and 2 is polynomial, then the only significant blowup is Lemma 3, due to the construction of an automaton whose size is exponential in $|\Sigma|$. However, it is not hard to see that the two exponential blowups in $|\Sigma|$ are orthogonal, and can be merged to a singly-exponential blowup.

Proposition 2. Consider an automaton \mathcal{A} . There exists an automaton \mathcal{B} such that $\mathfrak{J}(\mathcal{B}) = \Sigma^{\omega} \setminus \mathfrak{J}(\mathcal{A})$.

Proof. Observe that $w \in \Sigma^{\omega} \setminus \mathfrak{J}(\mathcal{A})$ if and only if $\Psi(w) \in \mathbb{M}^{\Sigma} \setminus \Psi(\mathfrak{J}(\mathcal{A}))$. Indeed, $w \in \Sigma^{\omega} \setminus \mathfrak{J}(\mathcal{A})$ means that every permutation of w is not accepted by \mathcal{A} , i.e., the Parikh image of w is not a Parikh image of any word accepted by \mathcal{A} . Thus, it suffices to construct \mathcal{B} such that $\Psi(\mathfrak{J}(\mathcal{B})) = \mathbb{M}^{\Sigma} \setminus \Psi(\mathfrak{J}(\mathcal{A}))$.

By Theorem 4, we can write $\Psi(\mathfrak{J}(\mathcal{A})) = \bigcup_{\mathfrak{m} \in \P} S_{\mathfrak{m}} + \mathfrak{m}$. Moreover, by Lemma 1 we can assume that this is an oblivious masked semilinear set. We claim that $\mathbb{M}^{\Sigma} \setminus \Psi(\mathfrak{J}(\mathcal{A})) = \bigcup_{\mathfrak{m} \in \P} (\mathbb{N}^{\Sigma} \setminus S_{\mathfrak{m}}) + \mathfrak{m}$. That is, we can complement the union by complementing each semilinear set, while keeping the masks.

To show this, consider a word $w \in \Sigma^{\omega}$ and write $\Psi(w) = \boldsymbol{x}_{\boldsymbol{w}} + \boldsymbol{\mathfrak{m}}_{w}$ where $\boldsymbol{x}_{\boldsymbol{w}} \in \mathbb{N}^{\Sigma}$, as per Section 3.1.

For the first direction, assume $\boldsymbol{x}_{\boldsymbol{w}} + \boldsymbol{\mathfrak{m}}_{\boldsymbol{w}} \in \mathbb{M}^{\Sigma} \setminus \Psi(\mathfrak{J}(\mathcal{A}))$, then $\boldsymbol{x}_{\boldsymbol{w}} + \boldsymbol{\mathfrak{m}}_{\boldsymbol{w}} \notin \bigcup_{\mathfrak{m} \in \P} S_{\mathfrak{m}} + \mathfrak{m}$. In particular, $\boldsymbol{x}_{\boldsymbol{w}} \notin S_{\mathfrak{m}_{w}}$ (otherwise we would have $\boldsymbol{x}_{\boldsymbol{w}} + \mathfrak{m}_{w} \in S_{\mathfrak{m}_{w}} + \mathfrak{m}_{w}$, which is part of the union). It follows that $\boldsymbol{x}_{\boldsymbol{w}} + \mathfrak{m}_{w} \in (\mathbb{N}^{\Sigma} \setminus S_{\mathfrak{m}_{w}}) + \mathfrak{m}_{w}$, so $\Psi(w) \in \bigcup_{\mathfrak{m} \in \P} (\mathbb{N}^{\Sigma} \setminus S_{\mathfrak{m}}) + \mathfrak{m}$.

Conversely, assume $\Psi(w) = x_w + \mathfrak{m}_w \in \bigcup_{\mathfrak{m} \in \P} (\mathbb{N}^{\Sigma} \setminus S_\mathfrak{m}) + \mathfrak{m}$, then observe that $x_w + \mathfrak{m}_w \in (\mathbb{N}^{\Sigma} \setminus S_{\mathfrak{m}_w}) + \mathfrak{m}_w$. Indeed, no other part of the union has the mask \mathfrak{m}_w . Assume by way of contradiction that $x_w + \mathfrak{m}_w \in \Psi(\mathfrak{J}(\mathcal{A})) = \bigcup_{\mathfrak{m} \in \P} S_\mathfrak{m} + \mathfrak{m}$, then by the same reasoning, $x_w + \mathfrak{m}_w \in S_{\mathfrak{m}_w} + \mathfrak{m}_w$. Thus, there are two vectors $y \notin S_{\mathfrak{m}_w}$ and $z \in S_{\mathfrak{m}_w}$ such that $y + \mathfrak{m} = x_w + \mathfrak{m} = z + \mathfrak{m}$. This contradicts the assumption that $S_{\mathfrak{m}_w}$ is oblivious (note that if $S_{\mathfrak{m}_w}$ is not oblivious, this is not a contradiction).

We conclude that $\mathbb{M}^{\Sigma} \setminus \Psi(\mathfrak{J}(\mathcal{A})) = \bigcup_{\mathfrak{m} \in \mathbb{T}} (\mathbb{N}^{\Sigma} \setminus S_{\mathfrak{m}}) + \mathfrak{m}$, and since semilinear sets are closed under complementation [3,7], the latter is also a masked semilinear

set. By applying Theorem 4 in the converse direction, we conclude that there exists an automaton \mathcal{B} such that $\Psi(\mathfrak{J}(\mathcal{B})) = \mathbb{M}^{\Sigma} \setminus \Psi(\mathfrak{J}(\mathcal{A}))$, and as mentioned above, for such \mathcal{B} we have $\mathfrak{J}(\mathcal{B}) = \Sigma^{\omega} \setminus \mathfrak{J}(\mathcal{A})$.

Complexity Analysis 6 (of Proposition 2). The description size of the complement of a semilinear set is singly exponential, in both the degree and the description of the entries [3,7]. We proceed similarly to Complexity Analysis 5, ending up with a singly-exponential blowup (not just in $|\Sigma|$).

Recall that Büchi automata are generally not determinizable. That is, there exists an automaton \mathcal{A} such that $\mathfrak{L}(\mathcal{A})$ is not recognizable by a deterministic Büchi automaton [17]. In stark contrast, an immediate corollary of Lemmas 2 and 3 is that jumping languages do admit determinization (also see Complexity Analysis 3).

Proposition 3. For every automaton \mathcal{A} there exists a deterministic automaton \mathcal{D} such that $\mathfrak{J}(\mathcal{A}) = \mathfrak{J}(\mathcal{D})$.

In particular, this means that additional acceptance conditions (e.g., Rabin, Streett, Muller and parity) cannot add expressiveness to the model.

Remark 3. Discussing determinization of jumping languages is slightly misleading: the definition of acceptance in a jumping language asks that *there exists* some permutation that is accepted by the automaton. This existential quantifier can be thought of as "semantic" nondeterminism. Thus, in a way, jumping languages are inherently nondeterministic, regardless of the underlying syntactic structure.

Finally, as is the case for jumping languages over finite words [20,10,11], jumping languages and ω -regular languages are incomparable. Indeed, the automata in Fig. 3 are examples of Büchi automata whose languages are not permutationinvariant, and are hence not jumping languages, and in Fig. 4 we demonstrate that the converse also does not hold.



Fig. 4. An automaton \mathcal{A} for which $\mathfrak{J}(\mathcal{A}) = \{u \cdot c^{\omega} \mid u \in \{a, b, c\}^* \land \#_a[u] = \#_b[u]\},$ which is not ω -regular by simple pumping arguments.

3.3 Jumping Languages - Algorithmic Properties

We turn our attention to algorithmic properties of jumping languages. We first notice that given an automaton \mathcal{A} we have that $\mathfrak{J}(\mathcal{A}) \neq \emptyset$ if and only if $\mathfrak{L}(\mathcal{A}) \neq \emptyset$. Thus, non-emptiness of $\mathfrak{J}(\mathcal{A})$ is NL-Complete, as it is for Büchi automata [8,16]. Next, we consider the standard decision problems:

- Containment: given automata \mathcal{A}, \mathcal{B} , is $\mathfrak{J}(\mathcal{A}) \subseteq \mathfrak{J}(\mathcal{B})$?
- Equivalence: given automata \mathcal{A}, \mathcal{B} , is $\mathfrak{J}(\mathcal{A}) = \mathfrak{J}(\mathcal{B})$?
- Universality: given an automaton \mathcal{A} , is $\mathfrak{J}(\mathcal{A}) = \Sigma^{\omega}$?

We show that all of these problems reduce to their analogues in finite-word jumping automata.

Theorem 7. Containment, Equivalence and Universality for jumping languages are coNP-complete (for fixed size alphabet).

Proof. We start with containment. by Theorem 4 and Lemma 1, given automata \mathcal{A}, \mathcal{B} we obtain canonical oblivious masked semilinear representation: $\Psi(\mathfrak{J}(\mathcal{A})) = \bigcup_{\mathfrak{m} \in \P} S_{\mathfrak{m}} + \mathfrak{m}$ and $\Psi(\mathfrak{J}(\mathcal{B})) = \bigcup_{\mathfrak{m} \in \P} T_{\mathfrak{m}} + \mathfrak{m}$. Then, we have $\mathfrak{J}(\mathcal{A}) \subseteq \mathfrak{J}(\mathcal{B})$ if and only if $\Psi(\mathfrak{J}(\mathcal{A})) \subseteq \Psi(\mathfrak{J}(\mathcal{B}))$, which in turn happens if and only if for every mask \mathfrak{m} it holds that $S_{\mathfrak{m}} + \mathfrak{m} \subseteq T_{\mathfrak{m}} + \mathfrak{m}$. We claim that the latter holds if and only if $S_{\mathfrak{m}} \subseteq T_{\mathfrak{m}}$.

Indeed, clearly if $S_{\mathfrak{m}} \subseteq T_{\mathfrak{m}}$ then $S_{\mathfrak{m}} + \mathfrak{m} \subseteq T_{\mathfrak{m}} + \mathfrak{m}$. Conversely, assume $S_{\mathfrak{m}} + \mathfrak{m} \subseteq T_{\mathfrak{m}} + \mathfrak{m}$ and let $u \in S_{\mathfrak{m}}$, so $u + \mathfrak{m} \in T_{\mathfrak{m}} + \mathfrak{m}$. Then, there exists $v \in T_{\mathfrak{m}}$ such that $u + \mathfrak{m} = v + \mathfrak{m}$, but since $T_{\mathfrak{m}}$ is \mathfrak{m} -oblivious, this means that $u \in T_{\mathfrak{m}}$.

Thus, deciding whether $\mathfrak{J}(\mathcal{A}) \subseteq \mathfrak{J}(\mathcal{B})$ amounts to deciding whether $S_{\mathfrak{m}} \subseteq T_{\mathfrak{m}}$ for every \mathfrak{m} . Since the latter is decidable, we get decidability of containment. Moreover, we can obtain in polynomial time nondeterministic automata $\mathcal{C}_1^{\mathfrak{m}}, \mathcal{C}_2^{\mathfrak{m}}$ corresponding to $S_{\mathfrak{m}}$ and $T_{\mathfrak{m}}$ (see Complexity Analyses 1 and 3), so it is enough to decide if $\mathfrak{J}_{\mathrm{fin}}(\mathcal{C}_1^{\mathfrak{m}}) \subseteq \mathfrak{J}_{\mathrm{fin}}(\mathcal{C}_2^{\mathfrak{m}})$ for all \mathfrak{m} . The latter problem is coNP-complete [15,14] (for fixed-size alphabet), and since we can start by nondeterministically guessing \mathfrak{m} , we retain this complexity. Moreover, coNP hardness trivially follows (by e.g., reducing a given language L to $L \cdot \#^{\omega}$ for a new symbol #).

Similar arguments hold for equivalence and universality, concluding the proof. $\hfill \Box$

4 Fixed-Window Jumping Languages

Recall from Definition 1 that given an automaton \mathcal{A} and $k \in \mathbb{N}$, the language $\mathfrak{J}_{k\boxplus}(\mathcal{A})$ consists of the words that have a k-window permutation that is accepted by \mathcal{A} .

Since k is fixed, we can capture k-window permutations using finite memory. Hence, as we now show, $\mathfrak{J}_{k\boxplus}(\mathcal{A})$ is ω -regular.

Theorem 8. Consider an automaton \mathcal{A} , then for every $k \in \mathbb{N}$ there exists a Büchi automaton \mathcal{B}_k such that $\mathfrak{J}_{k\boxplus}(\mathcal{A}) = \mathfrak{L}(\mathcal{B}_k)$.

Proof (Sketch). Intuitively, we construct \mathcal{B}_k so that it stores in a state the Parikh image of k letters, then nondeterministically simulates \mathcal{A} on all words with the stored Parikh image, while noting when an accepting state may have been traversed. We illustrate the construction for k = 2 in Fig. 5. The details are in Appendix A.2.

13

Complexity Analysis 9 (of Theorem 8). If \mathcal{A} has n states, then \mathcal{B} has at most $k^{|\Sigma|} \times n + n$ states. Moreover, computing \mathcal{B} can be done effectively, by keeping for every Parikh image the set of states that are reachable (resp. reachable via an accepting state).



Fig. 5. An automaton \mathcal{D} , and the construction \mathcal{B}_2 for $\mathfrak{J}_{2\boxplus}(\mathcal{D})$ as per Theorem 8.

A converse of Theorem 8 also holds, in the following sense: we say that a language $L \subseteq \Sigma^{\omega}$ is $k \boxplus$ invariant if $w \in L \iff w' \in L$ for every $w \sim_{k \boxplus} w'$. The following result follows by definition.

Proposition 4. Let \mathcal{A} be an automaton such that $\mathfrak{L}(\mathcal{A})$ is $k \boxplus$ -invariant, then $\mathfrak{L}(\mathcal{A}) = \mathfrak{J}_{k \boxplus}(\mathcal{A}).$

Theorem 8 and Proposition 4 yield that k-window jumping languages are closed under the standard operations under which ω -regular languages are closed (union, intersection, complementation), as these properties retain $k \boxplus$ invariance. Similarly, all algorithmic problems can be reduced to the same problems on Büchi automata (by Complexity Analysis 9, assuming $|\Sigma|$ is fixed).

Notice that the construction in Theorem 8 yields a nondeterministic Büchi automaton. This is not surprising in light of Remark 3. It does raise the question of whether we can find a deterministic Büchi automaton for $\mathfrak{J}_{k\boxplus}(\mathcal{D})$ for a *deterministic* automaton \mathcal{D} . In Proposition 5, we prove that this is not the case. That is, even deterministic k-window jumping has inherent nondeterminism.

Proposition 5. There is no nondeterministic Büchi automaton whose language is $\mathfrak{J}_{k\boxplus}(\mathcal{D})$ for $k \geq 2$ and \mathcal{D} as in Fig. 5.

Proof. Consider the deterministic automaton \mathcal{D} in Fig. 5. We start by showing that $\mathfrak{L}(\mathcal{B}_2) = \mathfrak{J}_{2\boxplus}(\mathcal{D})$ cannot be recognized by a deterministic Büchi automaton. Below, we show this *mutatis mutandis* for every even k, and with slightly more effort – for all $k \geq 2$.

By way of contradiction, consider a deterministic automaton \mathcal{A} such that $\mathfrak{L}(\mathcal{A}) = \mathfrak{J}_{2\boxplus}(\mathcal{D})$, then \mathcal{A} accepts $(ba)(bb)^{\omega}$, and therefore its run passes through an accepting state after some $(ba)(bb)^{m_1}$. Then, \mathcal{A} also accepts $(ba)(bb)^{m_1}(ab)(bb)^{\omega}$, so again there is m_2 such that \mathcal{A} reaches an accepting state after $(ba)(bb)^{m_1}(ab)(bb)^{m_2}$, we consider the word $(ba)(bb)^{m_1}(ab)(bb)^{m_2}(ab)(bb)^{\omega}$ and proceed in this fashion to construct an accepting run on a word with infinitely many (ab), which is not in $\mathfrak{J}_{2\boxplus}(\mathcal{D})$.

We now turn to show the same for any $k \ge 2$.

Assume by way of contradiction that there exists $k \in \mathbb{N}, k \geq 2$ such that $\mathfrak{J}_{k\boxplus}(\mathcal{D}) = \mathfrak{L}(A)$ for a deterministic automaton \mathcal{A} . k can be either odd or even.

We first assume that k is even and consider the word $w = bab^{\omega}$. $w \in \mathfrak{J}_{k\boxplus}(\mathcal{D})$ as there is $w' \sim_{k\boxplus} w$ that is accepted by \mathcal{D} , namely w' = w. Since $bab^{\omega} \in \mathfrak{L}(\mathcal{A})$, the unique run of \mathcal{A} on it is accepting, and let m_1 be an index such that $|bab^{m_1}|$ is divisible by k and α was visited at least once while reading bab^{m_1} . Now, consider the word $w = bab^{m_1}bab^{\omega}$. $w \in \mathfrak{J}_{k\boxplus}(\mathcal{D})$ as there is $w' \sim_{k\boxplus} w$ that is accepted by \mathcal{D} , and that is $w' = abb^{m_1}bab^{\omega}$. $bab^{m_1}bab^{\omega} \in \mathfrak{L}(\mathcal{A})$, then the unique run of \mathcal{A} on it is accepting, and let m_2 be an index such that $|bab^{m_1}bab^{m_2}|$ is divisible by k and α was visited at least twice while reading $bab^{m_1}bab^{m_2}$. Consider the word $w = bab^{m_1}bab^{m_2}bab^{\omega}$, it is also in $\mathfrak{J}_{k\boxplus}(\mathcal{D})$ because $w' = abb^{m_1}abb^{m_2}bab^{\omega}$ is accepted by \mathcal{D} . Following this fashion, we can construct an infinite word $w = bab^{m_1}bab^{m_2}bab^{m_3}...$ such that \mathcal{A} 's run on it visits α infinitely many often despite the fact that $w \notin \mathfrak{J}_{k\boxplus}(\mathcal{D})$.

We then assume that $k \geq 2$ is odd and consider the word $w = ab^{\omega}$. We have that $w \in \mathfrak{J}_{k\boxplus}(\mathcal{D})$ as there is $w' \sim_{k\boxplus} w$ that is accepted by \mathcal{D} , namely $w' = bab^{\omega}$. Since $ab^{\omega} \in \mathfrak{L}(A)$, the unique run of \mathcal{A} on it is accepting, and let m_1 be an index $|ab^{m_1}|$ is divisible by k and α was visited at least once while reading ab^{m_1} . Now, consider the word $w = ab^{m_1}ab^{\omega}$. Again $w \in \mathfrak{J}_{k\boxplus}(\mathcal{D})$ as there is $w' \sim_{k\boxplus} w$ that is accepted by \mathcal{D} , namely w' = w (with the accepting run $\rho = q_0q_1(q_0q_2)^{k-2}q_3^{\omega})$. Since $ab^{m_1}ab^{\omega} \in \mathfrak{L}(A)$, the unique run of \mathcal{A} on it is accepting, and let m_2 be an index such that $|ab^{m_1}ab^{m_2}|$ is divisible by k and α was visited at least twice while reading $ab^{m_1}ab^{m_2}$. Similarly we can show that $w = ab^{m_1}ab^{m_2}ab^{\omega} \in \mathfrak{J}_{k\boxplus}(\mathcal{D})$, and we can proceed and construct an infinite word $w = ab^{m_1}ab^{m_2}ab^{m_3}...$ such that \mathcal{A} 's run on it visits α infinitely many often despite the fact that $w \notin \mathfrak{J}_{k\boxplus}(\mathcal{D}) = \mathfrak{L}(A)$ for any k > 1.

5 Existential-Window Jumping Languages

Recall from Definition 1 that given an automaton \mathcal{A} , the language $\mathfrak{J}_{\exists\boxplus}(\mathcal{A})$ contains the words that have a k-window permutation that is accepted in \mathcal{A} for some k, i.e., $\mathfrak{J}_{\exists\boxplus}(\mathcal{A}) = \bigcup_{k \in \mathbb{N}} \mathfrak{J}_{k\boxplus}(\mathcal{A})$. We briefly establish some expressiveness properties of $\mathfrak{J}_{\exists\boxplus}(\mathcal{A})$.

Perhaps surprisingly, we show that $\exists \boxplus$ languages are strictly more expressive than Jumping languages. We start by showing that every jumping language can be defined as the $\exists \boxplus$ language of some automaton.

Theorem 10. Let \mathcal{A} be an automaton, then there exists an automaton \mathcal{B} such that $\mathfrak{J}_{\exists\boxplus}(\mathcal{B}) = \mathfrak{J}(\mathcal{A})$.

Proof. Intuitively, we modify the construction of the jumping automaton in Lemma 3 so that it produces an automaton for which the $\exists \boxplus$ language coincides with the given semilinear set.

Consider an automaton \mathcal{A} . By Lemmas 1 and 2, we can write $\Psi(\mathfrak{J}(\mathcal{A})) = \bigcup_{\mathfrak{m}\in\mathbb{Z}} S_{\mathfrak{m}} + \mathfrak{m}$ in canonical oblivious form. Similarly to Lemma 3, consider a specific linear set $L = \operatorname{Lin}(\mathbf{b}, P \cup E_{\mathfrak{m}})$ in the union above, and consider the words $w_{\mathbf{b}} \in \Sigma^*$ and $w_{\mathbf{p}} \in \Sigma^*$ for $\mathbf{p} \in P$ as in Lemma 3. Let \mathcal{N}' be a nondeterministic automaton such that $\mathfrak{L}_{\operatorname{fin}}(\mathcal{N}') = w_{\mathbf{b}} \cdot (w_{\mathbf{p}_1} + \ldots + w_{\mathbf{p}_n})^*$. We now obtain an automaton \mathcal{N} by connecting every accepting state of \mathcal{N}' to a new nondeterministic automaton $\mathcal{N}_{\mathfrak{m}}$ which, as a Büchi automaton, accepts the language $L_{\mathfrak{m}} = \{w \in \mathfrak{m}|_{\infty}^{\omega} \mid \text{ every } \sigma \in \mathfrak{m}|_{\infty}^{\omega} \text{ occurs infinitely often in } w\}$. The accepting states of \mathcal{N} are those of $\mathcal{N}_{\mathfrak{m}}$.

Observe that $L_{\mathfrak{m}}$ is permutation invariant, and in particular $\mathfrak{J}_{\exists \boxplus}(\mathcal{N}_{\mathfrak{m}}) = L_{\mathfrak{m}}$. Indeed, trivially we have $L_{\mathfrak{m}} = \mathfrak{L}(\mathcal{N}_{\mathfrak{m}}) \subseteq \mathfrak{J}_{\exists \boxplus}(\mathcal{N}_{\mathfrak{m}})$, and for the other direction – if $w \in \mathfrak{J}_{\exists \boxplus}(\mathcal{N}_{\mathfrak{m}})$, then $w \sim_{k \boxplus} w'$ for some $w' \in \mathfrak{L}(\mathcal{N}_{\mathfrak{m}}) = L_{\mathfrak{m}}$ and some $k \in \mathbb{N}$, but since $L_{\mathfrak{m}}$ is permutation invariant, and since we have in particular that $w \sim w'$, then $w \in L_{\mathfrak{m}}$.

Furthermore, $\Psi(\mathfrak{L}(\mathcal{N})) = \operatorname{Lin}(\mathbf{b}, P \cup E_{\mathfrak{m}}) + \mathfrak{m}$. We construct \mathcal{B} by taking the union over \mathfrak{m} in the masked semilinear set.

We claim that $\mathfrak{J}_{\exists\boxplus}(\mathcal{B}) = \mathfrak{J}(\mathcal{A})$. For the "easy" direction, let $w \in \mathfrak{J}_{\exists\boxplus}(\mathcal{B})$, then in particular $\Psi(w) \in \Psi(\mathfrak{L}(\mathcal{N})) = \operatorname{Lin}(\mathbf{b}, P \cup E_{\mathfrak{m}}) \subseteq S = \Psi(\mathfrak{J}(\mathcal{A}))$ (for some set in the union), so $w \in \mathfrak{J}(\mathcal{A})$ (as $\mathfrak{J}(\mathcal{A})$ is permutation invariant).

For the "hard" direction, assume $w \in \mathfrak{J}(\mathcal{A})$, then $\Psi(w) \in \operatorname{Lin}(\mathbf{b}, P \cup E_{\mathfrak{m}}) + \mathfrak{m}$ for some set in the union. We construct a word $w' \sim_{k\boxplus} w$ for some $k \in \mathbb{N}$ such that $w' \in \mathfrak{L}(\mathcal{B})$, thus showing $w \in \mathfrak{J}_{\exists\boxplus}(\mathcal{B})$. Let $k \in \mathbb{N}$ be large enough so that $w = u \cdot z$ with $z \in \mathfrak{m}|_{\infty}^{\omega}$ and $|u| \leq k$. There exists such k since w contains only finitely many letters outside $\mathfrak{m}|_{\infty}$. Recall that we assumed our sets are \mathfrak{m} oblivious, and therefore $\Psi(u) \in \operatorname{Lin}(\mathbf{b}, P \cup E_{\mathfrak{m}})$. Moreover, $\Psi(z) = \mathfrak{m}$, since $z \in L_{\mathfrak{m}}$. In particular, we can find a permutation u' of u such that $u' \in \mathfrak{L}(\mathcal{N})$. Denote w' = u'z, then w' satisfies the conditions above, so $w \in \mathfrak{J}_{\exists\boxplus}(\mathcal{B})$ and we are done. \Box

Next, we show that jumping languages are strictly less expressive that $\exists \boxplus$ languages.

Example 2. Consider the automaton \mathcal{A} in Fig. 3. We claim that $\mathfrak{J}_{\exists\boxplus}(\mathcal{A})$ is not the jumping language of any automaton. Indeed, it suffices to show that $\mathfrak{J}_{\exists\boxplus}(\mathcal{A})$ is not permutation invariant. Trivially, $(ab)^{\omega} \in \mathfrak{J}_{\exists\boxplus}(\mathcal{A})$. However, note that $(ab)^{\omega} \sim (a^n b^n)_{n=1}^{\infty}$ (i.e., the word $aba^2 b^2 a^3 b^3 \cdots$), since their Parikh images are both (∞, ∞) . The latter, however, is not in $\mathfrak{J}_{\exists\boxplus}(\mathcal{A})$, since any k-window permutation of $(a^n b^n)_{n=1}^{\infty}$ will eventually reach windows of the form a^k (inside a long enough sequence a^n), so any permutation of it will cause \mathcal{A} to reject. \Box

We finally show that ω -regular languages are incomparable with $\exists \boxplus$ languages:

Example 3. Recall the automaton \mathcal{A} in Fig. 4, with $\mathfrak{L}(\mathcal{A}) = (ab)^* \cdot c^{\omega}$. Observe that $\mathfrak{J}_{\exists \boxplus}(\mathcal{A}) = \mathfrak{J}(\mathcal{A}) = \{u \cdot c^{\omega} \mid u \in \{a, b, c\}^* \land \#_a[u] = \#_b[u]\}$. Indeed, if $w = u \cdot c^{\omega}$ as described, then $w \in \mathfrak{J}_{|u|\boxplus}(\mathcal{A})$ by rearranging the equal number of a's and b's to $(ab)^* \cdot c^{\omega}$. Conversely, if $w \in \mathfrak{J}_{\exists \boxplus}(\mathcal{A})$ then in particular $w \in \mathfrak{J}(\mathcal{A})$

(indeed, every $\exists \boxplus$ -permutation is also a permutation). Thus, $\mathfrak{J}_{\exists \boxplus}(\mathcal{A})$ is not ω -regular.

For the converse (i.e., an ω -regular language that is not $\exists \boxplus$), we argue that every $\mathfrak{J}_{\exists\boxplus}(\mathcal{A})$ is k-window invariant for every k, and since e.g., $\{(ab)^{\omega}\}$ is not 2-window invariant, then it is not recognizable as an $\exists \boxplus$ language.

Indeed, consider $w \in \mathfrak{J}_{\exists\boxplus}(\mathcal{A})$, and let $w_1 \sim_{k\boxplus} w$ for some k. Then there exists $w_2 \sim_{k\boxplus} w$ such that $w_2 \in \mathfrak{L}(\mathcal{A})$ and $k' \in \mathbb{N}$. But then $w_2 \sim_{k\boxplus} w$ and $w_1 \sim_{k\boxplus} w$, so by transitivity $w_1 \sim_{k\boxplus} w_2$, and so $w_2 \in \mathfrak{J}_{\exists\boxplus}(\mathcal{A})$. Thus, $\mathfrak{J}_{\exists\boxplus}(\mathcal{A})$ is k-window invariant.

Remark 4 (Alternative Semantics). Note that instead of the $\exists \exists \exists \exists semantics, we could require that there exists a partition of <math>w$ into windows of size at most k for some $k \in \mathbb{N}$. All our proofs carry out under this definition as well.

6 Future Research

We have introduced three semantics for jumping automata over infinite words. For the existing definitions, the class of $\exists \boxplus$ semantics is yet to be explored for closure properties and decision problems.

In a broader view, it would be interesting to consider quantitative semantics for jumping – instead of the coarse separation between letters that occur finitely and infinitely often, one could envision a semantics that takes into account e.g., the frequency with which letters occur. Alternatively, one could return to the view of a "jumping reading head", and place constraints over the strategy used by the automaton to move the head (e.g., restrict to strategies implemented by one-counter machines).

References

- 1. Abu Nassar, A., Almagor, S.: Simulation by rounds of letter-to-letter transducers. In: 30th EACSL Annual Conference on Computer Science Logic (2022)
- Almagor, S.: Process symmetry in probabilistic transducers. In: 40th IARCS Annual Conference on Foundations of Software Technology and Theoretical Computer Science (2020)
- Beier, S., Holzer, M., Kutrib, M.: On the descriptional complexity of operations on semilinear sets. EPTCS 252 p. 41 (2017)
- Büchi, J.R.: Symposium on decision problems: On a decision method in restricted second order arithmetic. In: Studies in Logic and the Foundations of Mathematics, vol. 44, pp. 1–11. Elsevier (1966)
- Cadilhac, M., Finkel, A., McKenzie, P.: Affine parikh automata. RAIRO-Theoretical Informatics and Applications 46(4), 511–545 (2012)
- Cadilhac, M., Finkel, A., McKenzie, P.: Bounded parikh automata. International Journal of Foundations of Computer Science 23(08), 1691–1709 (2012)
- Chistikov, D., Haase, C.: The taming of the semi-linear set. In: 43rd International Colloquium on Automata, Languages, and Programming (ICALP 2016). Schloss Dagstuhl-Leibniz-Zentrum fuer Informatik (2016)

- 18 S. Almagor and O. Yizhaq
- Emerson, E.A., Lei, C.L.: Modalities for model checking (extended abstract) branching time strikes back. In: Proceedings of the 12th ACM SIGACT-SIGPLAN symposium on Principles of programming languages. pp. 84–96 (1985)
- Fazekas, S.Z., Hoshi, K., Yamamura, A.: Two-way deterministic automata with jumping mode. Theoretical Computer Science 864, 92–102 (2021)
- Fernau, H., Paramasivan, M., Schmid, M.L.: Jumping finite automata: characterizations and complexity. In: International Conference on Implementation and Application of Automata. pp. 89–101. Springer (2015)
- Fernau, H., Paramasivan, M., Schmid, M.L., Vorel, V.: Characterization and complexity results on jumping finite automata. Theoretical Computer Science 679, 31–52 (2017)
- Guha, S., Jecker, I., Lehtinen, K., Zimmermann, M.: Parikh automata over infinite words. In: 42nd IARCS Annual Conference on Foundations of Software Technology and Theoretical Computer Science (2022)
- Klaedtke, F., Rueß, H.: Monadic second-order logics with cardinalities. In: Automata, Languages and Programming: 30th International Colloquium, ICALP 2003 Eindhoven, The Netherlands, June 30–July 4, 2003 Proceedings 30. pp. 681–696. Springer (2003)
- 14. Kopczynski, E.: Complexity of problems of commutative grammars. Logical Methods in Computer Science **11** (2015)
- Kopczynski, E., To, A.W.: Parikh images of grammars: Complexity and applications. In: 2010 25th Annual IEEE Symposium on Logic in Computer Science. pp. 80–89. IEEE (2010)
- Kupferman, O.: Automata theory and model checking. Handbook of model checking pp. 107–151 (2018)
- 17. Landweber, L.H.: Decision problems for w-automata. Tech. rep., University of Wisconsin-Madison Department of Computer Sciences (1968)
- Lavado, G.J., Pighizzini, G., Seki, S.: Operational state complexity under parikh equivalence. In: Descriptional Complexity of Formal Systems: 16th International Workshop, DCFS 2014, Turku, Finland, August 5-8, 2014. Proceedings 16. pp. 294–305. Springer (2014)
- McNaughton, R.: Testing and generating infinite sequences by a finite automaton. Information and control 9(5), 521–530 (1966)
- Meduna, A., Zemek, P.: Jumping finite automata. International Journal of Foundations of Computer Science 23(07), 1555–1578 (2012)
- Parikh, R.J.: On context-free languages. Journal of the ACM (JACM) 13(4), 570– 581 (1966)
- 22. Safra, S.: On the complexity of $\omega\text{-automata. In: Proc. 29th IEEE Symp. Found. of Comp. Sci. pp. 319–327 (1988)$
- To, A.W.: Parikh images of regular languages: Complexity and applications. arXiv preprint arXiv:1002.1464 (2010)
- 24. Vorel, V.: On basic properties of jumping finite automata. International Journal of Foundations of Computer Science **29**(01), 1–15 (2018)

A Proofs

A.1 ω -regular languages as union

The following claim is a folklore simple exercise, and we bring it for completeness.

19

Claim. Consider an automaton $\mathcal{A} = \langle Q, \Sigma, \delta, q_0, \alpha \rangle$, then we can write $\mathfrak{L}(\mathcal{A}) = \bigcup_{i=1}^{k} S_i \cdot T_i^{\omega}$ where the S_i and T_i are regular languages.

Proof. For states $p, q \in Q$, denote \mathcal{A}_p^q the automaton \mathcal{A} with initial state p and accepting states $\{q\}$. Observe that \mathcal{A} accepts a word if there is a run of \mathcal{A} that starts in q_0 , and visits some state $q \in \alpha$ infinitely often (indeed, since Q is finite, then if α is visited infinitely often, so is some state in α).

We thus have
$$\mathfrak{L}(\mathcal{A}) = \bigcup_{q \in \alpha} \mathfrak{L}_{\mathrm{fin}}(\mathcal{A}_{q_0}^q) \cdot \mathfrak{L}_{\mathrm{fin}}(\mathcal{A}_q^q)^{\omega}$$
.

A.2 Proof of Theorem 8

Proof. Intuitively, we construct \mathcal{B}_k so that it stores in a state the Parikh image of k letters, then nondeterministically simulates \mathcal{A} on all words with the stored Parikh image, while noting when an accepting state may have been traversed. We illustrate the construction for k = 2 in Fig. 5.

We now formalize this intuition.

Let $\mathcal{A} = \langle Q, \Sigma, \delta, q_0, \alpha \rangle$. We construct $\mathcal{B}_k = \langle Q', \Sigma, \delta', q'_0, \alpha' \rangle$ as follows.

- Let $Q_{\alpha} = \{q_{\alpha} \mid q \in Q\}$ be a copy of Q, and $P = \{\Psi(x) \mid x \in \Sigma^{< k}\} \subseteq \{0, \ldots, k-1\}^{|\Sigma|}$ be the set of Parikh images of all words of length < k (note that P is finite), then the states of \mathcal{B}_k are $Q' = (Q \times P) \cup Q_{\alpha}$.
- The initial state is $q'_0 = (q_0, \mathbf{0})$ (Note that $\Psi(\epsilon) = \mathbf{0}$).
- $-\alpha' = Q_{\alpha}.$
- The transition function is defined as follows. For a state (q, \boldsymbol{v}) and letter $\sigma \in \Sigma$ let $\boldsymbol{u} = \boldsymbol{v} + \Psi(\sigma)$. If $\sum_{\sigma \in \Sigma} \boldsymbol{u}(\sigma) < k$ we define $\delta((q, \boldsymbol{v}), \sigma) = (q, \boldsymbol{u})$. If $\sum_{\sigma \in \Sigma} \boldsymbol{u}(\sigma) = k$, define $\delta^*(q, \boldsymbol{u})$ to be the set of states reachable from q upon reading a word x with $\Psi(x) = \boldsymbol{u}$, and define $\delta^*_{\alpha}(q, \boldsymbol{u})$ to be the states reachable from q upon reading a word x with $\Psi(x) = \boldsymbol{u}$ via a run that visits α . We then have

$$\delta'((q, \boldsymbol{v}), \sigma) = \{(p, \boldsymbol{0}) \mid p \in \delta^*(q, \boldsymbol{v})\} \cup \{p_\alpha \mid p \in \delta^*_\alpha(q, \boldsymbol{v})\}$$

Finally, for every $q \in Q_{\alpha}$ we define $\delta'(q, \sigma) = \delta'((q, \mathbf{0}), \sigma)$ (that is, the Q_{α} copy behaves identically to the **0** copy of Q).

We now claim that $\mathfrak{J}_{k\boxplus}(\mathcal{A}) = \mathfrak{L}(\mathcal{B}_k)$: Let $w \in \mathfrak{J}_{k\boxplus}(\mathcal{A})$ and write $w = x_1x_2x_3...$ where $|x_i| = k$ for all *i*. Then, there is $w' = y_1y_2y_3... \sim_{k\boxplus} w$ (with all $|y_i| = k$) such that \mathcal{A} has an accepting run ρ on w'. Denote by $q_0, q_1, ...$ the states where for $i \geq 1$ we have $q_i = \rho_{ik}$. That is, the states that ρ visits after reading each y_i . Observe that for all *i* we have that $q_{k(i+1)} \in \delta^*(q_{ik}, \psi(y_{i+1})) = \delta^*(q_{ik}, \psi(x_{i+1}))$. Thus, there is a run of \mathcal{B} that upon reading x_{i+1} reaches the state $(q_{(i+1)k}, \mathbf{0})$ if ρ does not go through an accepting state on y_{i+1} or $(q_{(i+1)k})_{\alpha}$ if it does. Since ρ traverses infinitely many accepting states, so does the run of \mathcal{B}_k on w, and thus $w \in \mathfrak{L}(\mathcal{B}_k)$.

Conversely, assume $w \in \mathfrak{L}(\mathcal{B}_k)$ and again write $w = x_1 x_2 x_3 \dots$ with $|x_i| = k$. Then there is an accepting run of \mathcal{B}_k on w. By viewing the run in consecutive k-windows, we can construct words y_1, y_2, \dots such that $y_i \sim x_i$ for all i and such that \mathcal{A} accepts $w' = y_1 \cdot y_2 \cdots$. Indeed, in every k-window, \mathcal{B} simulates the run of \mathcal{A} on some permutation of the read word. We conclude that $w \in \mathfrak{J}_{k \boxplus}(\mathcal{A})$. \Box