An Algebraic Decision Procedure for Two-Variable Logic with a Between Relation

Andreas Krebs
Universität Tübingen, Germany

Kamal Lodaya
The Institute of Mathematical Sciences, Chennai, India

Paritosh K. Pandya
Tata Institute of Fundamental Research, Mumbai, India

Howard Straubing
Boston College, USA

Abstract

In earlier work (LICS 2016), the authors introduced two-variable first-order logic supplemented by a binary relation that allows one to say that a letter appears between two positions. We found an effective algebraic criterion that is a necessary condition for definability in this logic, and conjectured that the criterion is also sufficient, although we proved this only in the case of two-letter alphabets. Here we prove the general conjecture. The proof is quite different from the arguments in the earlier work, and required the development of novel techniques concerning factorizations of words. We extend the results to binary relations specifying that a factor appears between two positions.

2012 ACM Subject Classification Theory of computation → Algebraic language theory, Theory of computation → Finite Model Theory

Keywords and phrases two-variable logic, finite model theory, algebraic automata theory

Digital Object Identifier 10.4230/LIPIcs.CSL.2018.28

Acknowledgements We would like to thank Boston College, IMSc and TIFR for hosting our collaborative visits.

1 Introduction

In this paper we work with finite word models. The first-order definable languages – those definable in the logic $FO[<]$– were shown equivalent to starfree expressions by the work of Schützenberger [14], McNaughton and Papert [9]. The algebraic viewpoint established decidability of the definability question, that is, whether a given regular language is first-order definable. The first level of the quantifier alternation hierarchy was characterized by Knast [7]. Recently Place and Zeitoun characterized some more levels of the hierarchy [12, 13]. Two-variable logic was algebraically characterized by Thérien and Wilke [20]. They also showed decidability of its definability, and also of levels of the until hierarchy of temporal logic LTL, which was shown equivalent to first-order logic by Kamp [6].

In our earlier paper [8] we extended two-variable logic over finite words with between relations and studied this logic $FO^2[<, \texttt{bet}]$ and associated temporal logics. A between relation $a(x, y)$, for letters a of the finite alphabet, says that there is a position z labelled with the letter a such that $x < z < y$. The monoid variety $\mathbf{M}_e\mathbf{DA}$ is obtained by applying an operation M_e (see Section 2) to the variety $\mathbf{DA}$ of two-variable logic [15]. We showed that

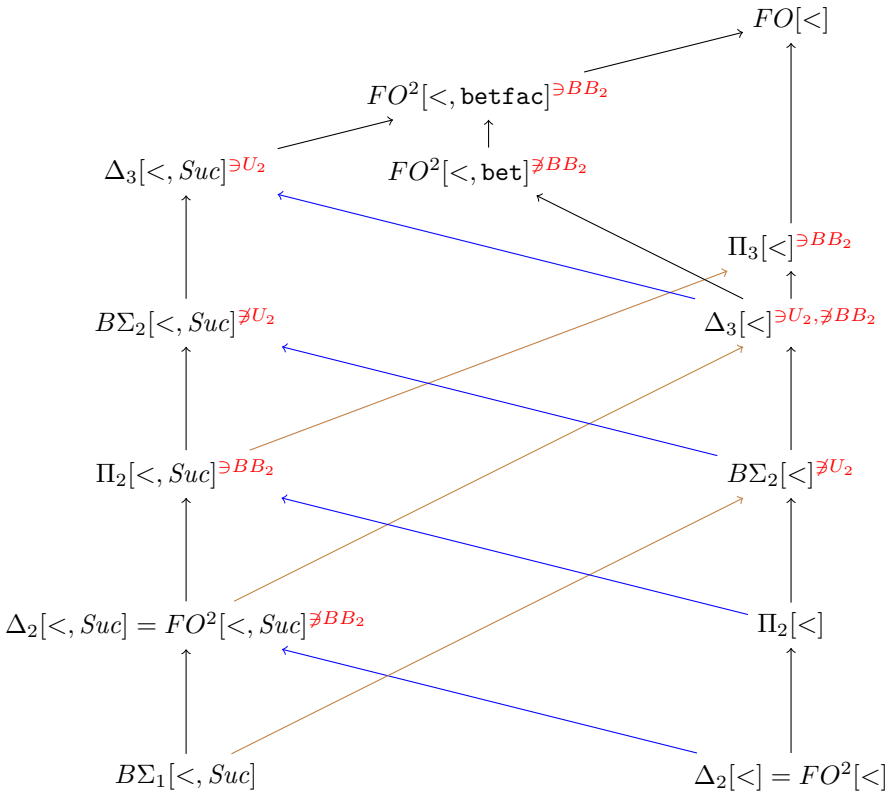

Figure 1 Dot depth and quantifier alternation hierarchies. The language U_2 over alphabet $A = \{a, b, c\}$ is $(A^* \setminus (A^*ac^*aA^*)) \cup (A^* \setminus (A^*bc^*bA^*))ac^*aA^*$, it consists of words which have no occurrence of bc^*b before an occurrence of ac^*a. The language BB_2 over $\{a, b\}$ is $(a(ab)^*b)^*$.

$\mathbf{M}_e\mathbf{DA}$ is an upper bound for $FO^2[<, \texttt{bet}]$, cutting across the quantifier alternation and until nesting depth hierarchies. We conjectured that this bound is tight and were able to show this for alphabets of size two. In this paper we establish this conjecture. Hence definability of a regular language in $FO^2[<, \texttt{bet}]$ is decidable. The variety $\mathbf{M}_e\mathbf{DA}$ first appeared in a paper by Weil [22]. Thus we provide a logical characterization of this variety.

The proof is somewhat intricate. We develop new techniques of factorization which are amenable to simulation using logic. At the end we rely on some hard algebra: the theorem on the locality of variety $\mathbf{DA}$, first shown by Almeida [1, 11]. Building on these techniques we show another main result, that the semigroup variety $\mathbf{M}_e\mathbf{DA} * \mathbf{D}$, obtained by applying to $\mathbf{M}_e\mathbf{DA}$ semidirect products with semigroups for the definite languages, characterizes a simple extension of our logic $FO^2[<, \texttt{bet}]$ to between relations $\langle u \rangle(x, y)$, for words u over the alphabet, which say that u is a factor, or substring, contained at positions between x and y. (So there are infinitely many between relations in this extended logic $FO^2[<, \texttt{betfac}]$.) The techniques we use here come from early work on providing a "delay" bound to varieties such as $\mathbf{DA}{*}\mathbf{D}$ [17, 21].

For the reader familiar with the lower levels of the quantifier alternation hierarchy of first-order logic (see [4] for a survey), these are the classes on the right in Figure 1. Those on the left are the classes of the original dot depth hierarchy of Cohen and Brzozowski [3]. The logics which we have introduced in [8] and in this paper are at top centre. They have a nonempty intersection with every level of both the hierarchies. The two example languages have played a prominent role in our work.

We also gave in [8] a tight complexity of *Expspace* for satisfiability of $FO^2[<, \mathtt{bet}]$. The techniques extend to provide the same complexity bounds for $FO^2[<, \mathtt{betfac}]$. This is an exponential blowup over LTL, but as noted in our earlier paper, these logics allow succinct binary representation of threshold constraints.

2 Setup

We denote by $FO^2[<, Suc]$ two-variable first-order logic with the successor relation Suc and the order relation $<$, interpreted in finite words over a finite alphabet A. (As usual, this stands for both the set of formulas, and the family of languages over A defined by such formulas.) Variables in first-order formulas are interpreted as positions in a word, and for each letter $a \in A$ there is a unary relation $a(x)$, interpreted to mean 'the letter in position x is a'. Thus sentences in this logic define properties of words, or, what is the same thing, languages $L \subseteq A^*$. Two-variable logic over words has been extensively studied, and has many equivalent characterizations in terms of temporal logic, regular languages, and the algebra of finite semigroups. (See, for instance, [19] and the many references cited therein.)

In [8] we extended $FO^2[<, Suc]$ to express 'betweenness' with only two variables. More precisely, predicates

$$a(x, y) = \exists z(a(z) \wedge x < z \wedge z < y),$$

which assert that there is an occurrence of the letter a strictly between x and y, were added to form the logic $FO^2[<, \mathtt{bet}]$. We also showed that counting the number of occurrences of the letter between x and y upto a threshold is definable in $FO^2[<, \mathtt{bet}]$. In Section 7 we will consider a further extension of this logic where we allow specification of factors between positions x and y.

We showed that languages defined by sentences of this logic satisfy an algebraic condition, which we explain next. For further background on the basic algebraic notions in this section, see Pin [10].

A *semigroup* is a set together with an associative multiplication. It is a *monoid* if it also has a multiplicative identity 1.

All of the languages defined by sentences of $FO[<]$ are regular languages. Our characterization of languages in these logics is based on properties of the *syntactic semigroup* $S(L)$ (resp. *syntactic monoid* $M(L)$) of a regular language L. This is the transition semigroup (monoid) of the minimal deterministic automaton recognizing L, and therefore finite. Equivalently, $S(L)$ is the smallest semigroup S that *recognizes* L, that is: There is a homomorphism $h : A^+ \to S$ and a subset $X \subseteq S$ such that $L = h^{-1}(X)$ (and similarly for monoids).

Let S be a finite semigroup. An *idempotent* $e \in S$ is an element satisfying $e^2 = e$.

If S is a finite semigroup and $s_1, s_2 \in S$, we write $s_1 \leq_{\mathcal{J}} s_2$ if $s_1 = rs_2t$ for some $r, t \in S$. This is a preorder, the so-called $\mathcal{J}$-ordering on S. Let $E(S)$ denote the set of all idempotents in S. If $e \in E(S)$, then M_e denotes the submonoid of M generated by the set $\{s \in S : e \leq_{\mathcal{J}} s\}$. The operation M_e appears in an unpublished memo of Schützenberger cited by Brzozowski [2]. He uses the submonoid generated by the generators of an idempotent element e of a semigroup. For example, if abc mapped to an idempotent element e, M_e would correspond to the language $(a + b + c)^*$.

The operation can be used at the level of semigroup and monoid classes. Thus the variety $\mathbf{M_eDA}$ has monoids M, all of whose submonoids of the form eM_ee for $e \in E(M)$, are in the variety $\mathbf{DA}$. Our main result is:

▶ **Theorem 1.** *Let $L \subseteq A^*$. L is definable in $FO^2[<, \mathtt{bet}]$ iff $M(L) \in \mathbf{M_eDA}$.*

In our earlier paper [8], we proved necessity of the algebraic condition, but only proved sufficiency in the case $|A| = 2$. Sections 3 to 6 are devoted to the proof of sufficiency for general alphabets.

The logical machinery we will use is quite standard (see [18]). In our paper [8], we defined Ehrenfeucht-Fraïssé games for the logic $FO^2[<, \texttt{bet}]$. We use the games in this paper to prove the existence of an $FO^2[<, \texttt{bet}]$ formula θ, by the equivalent formulation that there is an integer $k > 0$ such that, if (w, i), (w', j) are marked words in which i and j are inequivalent, then Player 1 has a winning strategy in the k-round game for $FO^2[<, \texttt{bet}]$ in (w, i), (w', j). That is, $(w, i) \models \theta$, $(w', j) \not\models \theta$, so Player 1 has a winning strategy in the k-round game, where k is the quantifier depth of θ. Conversely, suppose Player 1 always has such a winning strategy. Consider all marked words (w, i), and take the union of all the $\equiv_k$-classes of these w_i. This union is defined by a depth-k formula which we call θ. If there were any $(w', j) \models \theta$ where j is inequivalent, then we would have some (w, i) with $(w, i) \equiv_k (w', j)$, contrary to hypothesis. So θ is satisfied by exactly the required (w, i).

Notation. If $w \in A^*$, then we write $\alpha(w)$ to denote the set of letters that occur in w. We will interpret $a(x, y)$ to be false whenever $y \leq x$.

3 The factorization sequence

We are going to prove Theorem 1, that our algebraic condition from [8] indeed holds over all alphabets. We only need to prove one direction.

▶ **Lemma 2** ($\mathbf{M_e DA}$ characterizes $FO^2[<, \texttt{bet}]$)**.** *Suppose finite monoid M satisfies the property $e \cdot M_e \cdot e \in \mathbf{DA}$ for all $e \in E(M)$. Then for all $m \in M$, $h^{-1}(m) \in FO^2[<, \texttt{bet}]$.*

This will be proved by induction on the alphabet size. It is trivial for a one-letter alphabet, so assume $|A| > 1$ and that the theorem holds for all strictly smaller alphabets.

The bulk of the proof is combinatorics on words and finite model theory. We only use the algebra at the end.

For now we distinguish a letter $a \in A$, and restrict our attention to *a-words* w with the following three properties:

- $\alpha(w) = A$
- a is the first letter of w
- a is the *last* letter to appear in a *right-to-left* scan of w; that is, $w = xay$ where $\alpha(y) = A \backslash \{a\}$.

We describe an algorithm for constructing a sequence of factorizations for any a-word. Each step of the algorithm is divided into two sub-steps, and we will refer to each of these sub-steps as a factorization scheme. The factors that occur in each scheme are formed by concatenating factors from the previous scheme. That is, at each step, we clump smaller factors into larger ones, so the number of factors decreases (non-strictly) at each step.

We begin by putting a linear ordering $<$ on the set of proper subalphabets of A that contain the letter a. This will be a topological sort of the subset partial order. That is, if B, C are two such subalphabets with $B \subsetneq C$, then $B < C$, but otherwise the ordering is arbitrary. For example, with $A = \{a, b, c, d\}$, we can take

$$\{a\} < \{a, b\} < \{a, c\} < \{a, b, c\} < \{a, d\} < \{a, b, d\} < \{a, c, d\},$$

as one of many possibilities. One way to think about our techniques is as a refinement of Thérien and Wilke's combinatorial characterization of **DA** [20] which only used the inclusion order over an alphabet.

Here is the algorithm, which is the new development over **DA**:

- Initially factor w as $au_1 \cdots au_k$, where each $\alpha(u_i)$ is properly contained in A.
- For each proper subalphabet B of A with $a \in B$, following the linear order
  - For each factor u such that $\alpha(u) = B$, combine all sequences of consecutive factors of this kind into a single factor. We say that B is now *collected*.
  - For each factor u such that $\alpha(u) = B$, combine each such factor with the factor immediately to its right. We say that B is now *capped*.

Here is an example. We begin with an a-word and its initial factorization:

$adccdcc \cdot adc \cdot a \cdot a \cdot a \cdot addccdcccdbcdc \cdot a \cdot ac \cdot abcbbd$

We use the ordering in the example above. We start with $B = \{a\}$ and collect B:

$adccdcc \cdot adc \cdot \mathrm{aaa} \cdot addccdcccdbcdc \cdot \mathrm{a} \cdot ac \cdot abcbbd$

then cap it:

$adccdcc \cdot adc \cdot \mathrm{aaaaddccdcccdbcdc} \cdot \mathrm{aac} \cdot abcbbd$

We choose $B = \{a, b\}$. There is nothing to do here, because no factor contains just a and b. $B = \{a, c\}$ is already collected, because there is no pair of consecutive factors with this alphabet, so we cap it:

$adccdcc \cdot adc \cdot \mathrm{aaaaddccdcccdbcdc} \cdot \mathrm{aacabcbbd}$

The next subalphabet in order that occurs as a factor is $\{a, c, d\}$. We collect:

$\mathrm{adccdccadc} \cdot \mathrm{aaaaddccdcccdbcdc} \cdot \mathrm{aacabcbbd}$

then cap:

$\mathrm{adccdccaddadaaaaddccdcccdbcdc} \cdot \mathrm{acabcbbd}$

Let us make a few general observations about this algorithm: Every proper subset of A containing a that occurs as the alphabet of a factor will eventually be capped, because the rightmost factor au_k of the initial factorization contains all the letters of A. Once B has been collected, there is no pair of consecutive factors with content B. Once B has been capped, there are no more factors with content B nor with strictly smaller content. Thus at the end of the process, every factor contains all the letters of A.

Note as well that immediately after a subalphabet B is collected to create a (possibly) larger factor u, both the factor immediately to the right of u and immediately to the left of u must contain a letter that is not in u.

4 Starts and jumps

We establish below several model-theoretic properties of the factorization schemes produced by the above algorithm.

▶ **Lemma 3.**

(a) *There is a formula* start *in* $FO^2[<, \texttt{bet}]$ *such that for all a-words* w, $(w, i) \models start(x)$ *if and only if* i *is the first position in a factor of* w.

(b) *Let* $\phi_1(x)$ *be a formula in* $FO^2[<, \texttt{bet}]$. *Then there is a formula* next *in* $FO^2[<, \texttt{bet}]$ *with the following property: Let* w *be an a-word and let* i *be the first position in some factor of* w *that is not the rightmost factor. Then* $(w, i) \models \phi_1(x)$ *if and only if* $(w, i_{succ}) \models next(x)$, *where* i_{succ} *is the first position in the next factor of* w. *We also define the analogous property, with 'rightmost' replaced by 'leftmost',* next *by* previous, *and* i_{succ} *by* i_{pred}.

Proof. We prove these properties by induction on the construction of the sequence of factorization schemes. That is, we prove that they hold for the initial factorization scheme, and that they are preserved in each sub-step of the algorithm. For the induction, we will use Ehrenfeucht-Fraïssé games for the logic $FO^2[<, \texttt{bet}]$ to argue for the existence of the formula $start = start_\tau$ (see Section 2).

We note that the claim in Item (b) implies the condition on games (possibly with different parameters). If for every formula ϕ_1 there is a corresponding 'successor' formula $next$, then there is some constant c such that $\texttt{qd}(next) \leq c + \texttt{qd}(\phi_1)$, where $\texttt{qd}$ denotes quantifier depth. Suppose that Player 2 wins the $(k + c)$-round game in $(w, i_{succ}), (v, i_{succ})$. Then $(w, i_{succ}) \equiv_{k+c} (v, j_{succ})$. Consider the formula ϕ_1 that defines the $\equiv_k$-class of (w, i). Then $(w, i_{succ}) \models next$, so $(v, j_{succ}) \models next$. Thus $(v, j) \models \phi_1$, so $(w, i) \equiv_k (v, j)$, and Player 2 wins the k-round game in these words.

We begin with Item (a): For the initial factorization, we simply take $start(x)$ to be $a(x)$: the factor starts are exactly the positions that contain a. We now assume that τ is some factorization scheme in the sequence, and that for the preceding factorization scheme σ, the required formula, which we denote $start_\sigma$, exists.

To establish this formulation, let $(w, i), (w', j)$ be as described. Since, by the inductive hypothesis, the formula $start_\sigma$ for the preceding scheme σ exists, we can treat this as if it is an atomic formula, in describing our game strategy. Observe that i must also be the start of a factor of w according to the previous factorization scheme σ. We write this as $start_\sigma(i)$ rather than the more verbose $(w, i) \models start_\sigma(x)$. If j does not satisfy $start_\sigma(j)$, then by induction we are done, and can take the number k of rounds to be the quantifier depth of $start_\sigma$. Thus j is the start of a factor with respect to the scheme σ, not with respect to τ. This can happen in one of two ways, depending on whether the most recent sub-step collected a subalphabet B, or capped a subalphabet B.

In the first case, we will describe a winning strategy for Player 1 in a game that lasts just a few more rounds than the game for the previous scheme. Position j was the start of a factor in the prior scheme σ, and has been collected into a larger factor that begins at position to the left of j. First suppose that i is the start of a factor with content different from B. Then this factor must contain some $c \notin B$. Player 1 then wins as follows: He moves right in (w', j), jumping to the start j' of the next factor (which must satisfy $start_\sigma(j')$). In so doing, all the letters he jumps belong to B. Player 2 must also jump to the right in (w, i), and must also land on the start of a factor in the scheme σ; otherwise, by induction, Player 1 will win the game in the next k rounds. But to do so, Player 2 will have to jump over a position containing c, so she cannot legally make this move. Thus i must be the start of

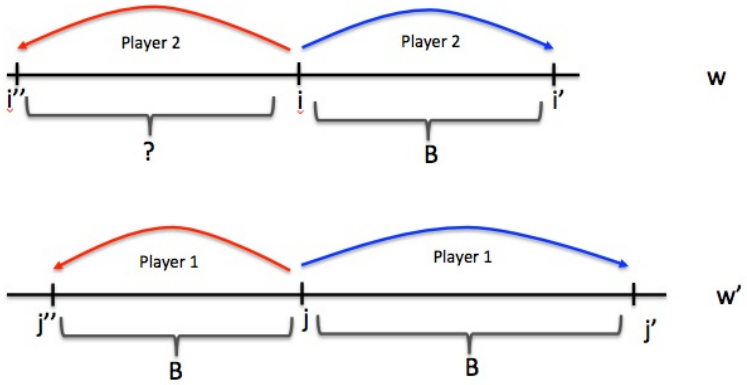

Figure 2 Game-based proof of definability of factor starts. The figure shows the two words just after the step collecting the subalphabet B. We suppose i, j are factor starts for the preceding factorization scheme σ, and that i, but not j, is a factor start for the present scheme τ. This means that the factor with respect to σ beginning at j was joined to the previous factor as a result of the collection. If Player 1 moves to the start j' of the next factor of w' with respect to σ (blue arrow), then he jumps over precisely the letters of B. Thus for Player 2 to have a response, i must be the start of a factor with alphabet B. But this means that the factor with respect to σ in w that precedes i must contain a letter not in B. As a result, Player 2 cannot reply to a move by Player 1 to the start j'' of the factor with respect to σ that precedes j (red arrow).

a factor with content B. In this case, Player 1 moves left in (w', j) to j'', the start of the previous factor with respect to σ. In doing so, he jumps over letters in B. Now Player 2 must also jump to the left in (w, i) to a position that was the start of a factor with respect to σ, but must jump over a letter not in B to do this, so Player 1 wins again. (See Figure 2.)

In the second case, where B was capped, j was the start of a factor that immediately followed a newly-collected factor with content B. Player 1 jumps left to j', the start position of this factor, and in doing so jumps over a segment with content B. Thus Player 2 must jump to the start of a factor with respect to σ. For this to be a legal move, the segment she jumps must have content B. However, this is impossible, for any factor with this content in the scheme σ would have been capped by the following factor, so that i cannot be the start of a factor for τ. (Figure 3.)

Now for Item (b). Again, we use a game argument. We claim it will be enough to establish the following for sufficiently large values of k: Let $(w, i), (v, j)$ be marked words, where i, j are the starts of factors, and let $(w, i_{succ}), (v, i_{succ})$ be the same words, where the indices i_{succ}, j_{succ} mark the start of the successor factors. If Player 1 has a winning strategy in the k-round game in $(w, i), (v, j)$, then he has a winning strategy in the k'-round game in $(w, i_{succ}), (v, i_{succ})$ for some k' that depends only on k and the alphabet size, and not on v and w. Equivalently, if Player 2 wins in $(w, i_{succ}), (v, i_{succ})$ then she wins in $(w, i), (v, j)$. Of course, there is the analogous formulation for *previous*.

So we will suppose Player 1 has a winning strategy in the k-round game in $(w, i), (v, j)$, where k is at least as large as the quantifier depth of $start_\tau$. We will prove the existence of a strategy in $(w, i_{succ}), (v, j_{succ})$ for the k'-round game, where k' is larger than k. (By tracing through the various cases of the proof carefully, you can figure out how large k' needs to be.) What we will show in fact is that for each τ, Player 1 can force the starting configuration $(w, i_{succ}), (v, j_{succ})$ to the configuration $(w, i), (v, j)$, and from there apply his winning strategy in $(w, i), (v, j)$.

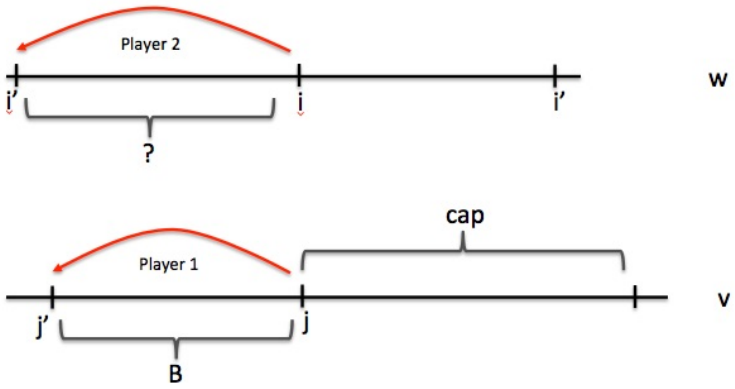

Figure 3 This shows the case just after the step that caps the subalphabet B. Again suppose i, j are factor starts for the preceding factorization scheme σ, and that i, but not j, is a factor start for the present scheme τ. If Player 1 moves in v from j to j', the start of the factor preceding j with respect to σ, then only letters in B are jumped. If Player 2 moves left from i to another factor start with respect to σ, she will have to jump over letters that are not in B, because all factors with alphabet B have been capped; thus Player 2 cannot respond to this move.

The base step is where τ is the initial factorization scheme. Here the factor starts are just the positions where the letter a occurs. Player 1 begins by jumping from i_{succ} to i. For Player 2 to respond correctly, she must jump from j_{succ} to j, because she is required to move left and land on a position containing a while jumping over a segment that does not contain the letter a.

So now we will suppose that τ is not the initial factorization scheme. We again denote the previous factorization scheme by σ. We assume that the property in Item (b) holds for σ. Thanks to what we proved above, we know that the property in Item (a) holds for both τ and σ. This means that we can treat $start_\tau$ and $start_\sigma$ essentially as atomic formulas.

If i_{succ} is also the successor of i (that is, the start of the next factor) with respect to the previous factorization scheme σ, and j_{succ} is the successor of j, then we have the desired result by induction. Thus we may suppose that one or both of the factor starts, either between i and i_{succ} or between j and j_{succ}, or both, were eliminated in the most recent sub-step of the algorithm.

Let us suppose first that the most recent sub-step was a collection step, collecting the subalphabet B. Player 1 jumps from i_{succ} left to i. The set of jumped letters is B. Player 2 must respond by jumping to some $j' < j_{succ}$ where j' satisfies $start_\tau$. If $j' < j$, then the set of jumped letters necessarily contains a letter not in B, so such a move is not legal. Thus $j' = j$. Player 1 now follows his winning strategy in $(w, i), (v, j)$. The identical strategy works for the predecessor version, because any factor following the sequence of collected factors must contain a letter not in B.

So suppose that the most recent sub-step was a capping step, and that the subalphabet B was capped. We may suppose that there is some i' with $i < i' < i_{succ}$ such that $start_\sigma(i')$, but not $start_\tau(i')$. Thus the interval from i to $i' - 1$ has content B and constitutes a factor that was collected during the prior sub-step, before being capped in the present one. Player 1 uses his strategy from the previous factorization scheme to force the configuration to $(w, i'), (v, j')$, where j' is the start of the factor preceding j in the scheme σ. Observe that we must have that j' does not satisfy $start_\tau$ because i' does not satisfy $start_\tau$. Thus $j < j' < j_{succ}$, so the interval from j to $j' - 1$ is also a factor with content B that was collected during the previous

substep. Player 1 now moves from i' left to i. Player 2 must respond with a move to $j'' \leq j$ such that $start_\tau(j'')$ holds. We cannot have $j'' < j$, for then the set of jumped letters would include a letter not in B. Thus $j'' = j$, and the game is now in the configuration $(w, i), (v, j)$.

The strategy for a capped step in the predecessor game uses the same idea: We may assume there is some i' with $i_{prec} < i' < i$ such that the interval from i_{prec} to $i'-1$ has content B and constitutes a factor that was collected during the prior sub-step, before being capped in the present one. Thus in the previous scheme σ, i' was the successor position of i_{prec}. Player 1 uses his strategy from the previous scheme to force the game to the configuration $(w, i'), (v, j')$, where j' is the successor of j_{prec} in the scheme σ. We must have the set of jumped letters to be B in each case, so the intervals from i' to $i-1$ and j' to $j-1$ are the caps applied in the scheme τ, and thus i is the successor of i', and j the successor of j', in the scheme σ. Player 1 now uses his strategy for the scheme σ to force the game from the configuration $(w, i'), (v, j')$ to $(w, i), (v, j)$. ◀

5 Simulating factorization in logic

A factorization scheme σ gives a factorization $\sigma(w) = (w_1, \ldots, w_k)$ of an a-word w. This in turn gives a word $\sigma_h(w) = m_1 \cdots m_k \in M^+$. We say that σ *admits simulations* if the following properties hold.

- For each sentence $\psi \in FO^2[<, Suc]$ over the alphabet M, there exists a sentence $\phi \in FO^2[<, \mathtt{bet}]$ over the alphabet A with the following property. Let w be an a-word.

  $$w \models \phi \quad \mathtt{iff} \quad \sigma_h(w) \models \psi.$$

- For each formula $\psi(x) \in FO^2[<, Suc]$ with one free variable over the alphabet M, there exists a formula $\phi(x) \in FO^2[<, \mathtt{bet}]$ with one free variable over the alphabet A with the following property. Let w be an a-word, $1 \leq i \leq k$ and let j_i be the position within w of the first letter of w_i in $\sigma(w)$. Then

  $$(w, j_i) \models \phi(x) \quad \mathtt{iff} \quad (\sigma_h(w), i) \models \psi(x).$$

▶ **Lemma 4** (Simulation). *Each factorization scheme in our sequence admits simulations.*

It is useful to have abbreviations for commonly used subformulas of $FO^2[<, \mathtt{bet}]$. If B is a subalphabet of A, we write $[B](x, y)$ to mean the conjunction of $\neg c(x, y)$ over all $c \notin B$; in other words, 'every letter between x and y belongs to B'. $[a](x, y)$ is always true if $y \leq x$ because $a(x, y)$ is false whenever $y \leq x$. We denote by $[\![B]\!](x, y)$ the conjunction of $[B](x, y)$ together with the conjunction of $b(x, y)$ over all $b \in B$; in other words, B is exactly the set of letters between x and y.

Proof. The first claim in the Theorem follows easily from the second. So we will begin with the formula $\psi(x) \in FO^2[<, Suc]$ over M and and show how to produce $\phi(x)$. We prove this by induction on the construction of formula ψ. So the base case is where $\psi(x)$ is an atomic formula $m(x)$, where $m \in M$. This means that for each factorization scheme σ, we have to produce a formula $\phi_{m,\sigma}(x)$ such that for an a-word w, $(w, i) \models \phi_{m,\sigma}(x)$ if and only if the factor starting at i maps to m under h.

We do this by induction on the sequence of factorization schemes. In the initial factorization, every factor is of the form au, where $a \notin \alpha(u)$. This factor maps to m if and only if $h(u) = m'$ for some $m' \in M$ satisfying $h(a) \cdot m' = m$. Since we suppose the main theorem holds for every alphabet strictly smaller than A, there is a sentence $\rho \in FO^2[<, \mathtt{bet}]$ such that

$u \models \rho$ if and only if $h(u) = m'$ where $h(a) \cdot m' = m$. We now relativize ρ to obtain a formula ρ' with one free variable that is satisfied by (w, i) if and only if the factor of w starting at i has the form au, where $u \models \rho$. To do this, we do a standard relativization trick, working from the outermost quantifier of ρ inward. We can assume that all the quantifiers at the outermost level quantify the variable y. We replace each of these quantified formulas $\exists y \eta(y)$ by $\exists y(y > x \wedge \neg a(x, y) \wedge \eta(y))$. Similarly, as we work inward, we rewrite each occurrence of $\exists z'(z' > z \wedge \eta)$ and $\exists z'(z' < z \wedge \eta)$, where $\{z, z'\} = \{x, y\}$, by adding the clause $\neg a(z, z')$ or $\neg a(z', z)$. In essence, each time we jump left or right to a new position, we check that in so doing we did not jump over any occurrence of a, and thus remain inside the factor.

We now assume that τ is not the initial factorization scheme, and that the formula $\phi_{m,\sigma}(x)$ exists for the preceding factorization scheme σ. We first consider the case where τ was produced during a step that collected a subalphabet B. Observe that we can determine within a formula whether i is the start of a factor that was produced during this collection step, with the criterion

$$\exists y(x < y \wedge start_\tau(y) \wedge [\![B]\!](x, y)).$$

(This includes the case where the collection is trivial because there is only one factor to collect.) If this condition does not hold, then we can test whether the factor maps to m with the formula produced during the preceding step. So we suppose that i is the first position of one of the new 'collected' factors. Since $B \subsetneq A$, there is a sentence ρ of $FO^2[<, \mathtt{bet}]$ satisfied by exactly the words over this smaller alphabet that map to m. Once again, we must relativize ρ to make sure that whenever we introduce a new quantifier $\exists x(y > x \wedge \cdots)$ or $\exists x(y < x \wedge \cdots)$ we do not jump to a position outside the factor. To do this, we can replace $\exists x(y > x \wedge \cdots)$ by

$$\exists x(y > x \wedge [B](x, y) \wedge \exists x(y < x \wedge start_\tau(x) \wedge [B](x, y))).$$

In other words, we did not jump over any letter not in $\{a\} \cup B$, and there is a factor start farther to the right that we can reach without jumping over any letter not in B. We replace $\exists x(y < x \wedge \cdots)$ by

$$\exists x(y < x \wedge [B](y, x) \wedge \exists x(x \leq y \wedge start_\tau(x) \wedge [B](x, y))),$$

using essentially the same idea.

Now suppose that τ was produced during a step that capped the subalphabet B. Again, we can write a formula that says that i is the start of a new factor produced in this process: it is exactly the formula that said i was the start of a factor that collected B in the preceding scheme σ. So we only need to produce a formula that says the factor of w beginning at i maps to m under the assumption that this is one of the new 'capped' factors. Our factor has the form u_1u_2, where u_2 is the cap and u_1 is the factor in which B was collected. We consider all pairs m_1, m_2 such that $m_1 \cdot m_2 = m$. We know that there are formulas $\rho_1(x)$ and $\rho_2(x)$ telling us that the factors in the preceding scheme σ map to m_1 and m_2. We use the same formula $\rho_1(x)$, and take its conjunction with $next(x)$, the successor formula derived from $\rho_2(x)$ by means of Item (b) in Lemma 3. We are using the fact that the start of u_2 is the successor of the start of u_1 under the preceding scheme σ.

We are almost done (and we no longer need to induct on the sequence of factorization schemes) because $FO^2[<, Suc]$ formulas can be reduced to a few normal forms [5]. Let us first suppose that our formula ψ has the form $\exists x(Suc(x, y) \wedge \kappa(x))$. The inductive hypothesis

is that there is a formula μ simulating κ. Let *previous* be the predecessor formula whose existence is given by Item (b) of Lemma 3. We claim that *previous* simulates ψ. To see this, suppose w is an a-word, and j_i is the position where the i^{th} factor of w begins.

Suppose $(w, j_i) \models previous$. Then $(w, j_{i+1}) \models \mu$.

So $(\sigma_h(w), i+1) \models \kappa$, which gives $(\sigma_h(w), i) \models \psi$.

This implication also runs in reverse, so we have shown that *previous* simulates ψ. Using the successor formula in place of the predecessor formula gives us the analogous result for ψ in the form $\exists x(Suc(y,x) \wedge \kappa(x))$. ◀

6 Proof of the main lemma

Proof of Lemma 2. Again, we assume $|A| > 1$ and that the theorem holds for all strictly smaller alphabets. Let $m \in M$, where M satisfies the $\mathbf{M_e DA}$ property. We need to show $h^{-1}(m)$ is defined by a sentence of $FO^2[<, \mathtt{bet}]$. As an overview, we will first, through a series of quite elementary steps, reduce this to the problem of showing that for each $a \in A$ and $s \in M$, the set of a-words mapping to s is defined by a sentence of $FO^2[<, \mathtt{bet}]$. We then use Lemma 4 on simulations, together with the identity $\mathbf{LDA} = \mathbf{DA} * \mathbf{D}$ [1] to find a defining sentence for the set of a-words that map to s.

First note that $h^{-1}(m) = \bigcup_{B \subseteq A} \{w \in h^{-1}(m) : \alpha(w) = B\}$.

It thus suffices to find, for each subalphabet B, a sentence ψ_B of $FO^2[<, \mathtt{bet}]$ defining the set of words $\{w \in h^{-1}(m) : \alpha(w) = B\}$. We then obtain a sentence for $h^{-1}(m)$ as

$$\bigvee_{B \subseteq A} (\psi_B \wedge \bigwedge_{b \in B} \exists x b(x) \wedge \bigwedge_{b \notin B} \neg \exists x b(x)).$$

Since we obtain the sentences ψ_B for proper subalphabets B of A by the induction hypothesis, we only need to find ψ_A.

For each w with $\alpha(w) = A$, let $last(w)$ be the last letter of w to appear in a right-to-left scan of w. It will be enough to find, for each $a \in A$, a sentence ϕ_a of $FO^2[<, \mathtt{bet}]$ defining $\{w \in h^{-1}(m) : last(w) = a\}$, since we then get ψ_A as

$$\exists y(a(y) \wedge \forall x(x > y \to \neg a(x)) \wedge \bigwedge_{b \neq a} \exists x(x > y \wedge b(x))) \wedge \phi_a.$$

A word w with $\alpha(w) = A$ and $last(w) = a$ has a unique factorization $w = uv$, where $\alpha(u) = A \backslash \{a\}$, and v is an a-word. We consider all factorizations $m = m_1 m_2$ in M. By the inductive hypothesis, there is a sentence μ of $FO^2[<, \mathtt{bet}]$ defining the set of all words over $A \backslash \{a\}$ that map to m_1. Suppose that we are able to find a sentence ν defining the set of all a-words mapping to m_2. We can then use a simple relativizing trick to obtain a sentence defining all concatenations uv such that $u \models \mu$ and $v \models \nu$. One simply modifies each quantified subformula $\exists x \zeta$ of μ and ν, starting from the outside, changing them to

$$\exists x(\neg \exists y(y \leq x \wedge a(y))) \text{ and } \exists x(\exists y(y \leq x \wedge a(y))).$$

The conjunction of the two modified sentences now says that μ holds in the factor preceding the first occurrence of a, and ν holds in the factor that begins at the first occurrence of a. Take the disjunction of these conjunctions over all factorizations $m_1 m_2$ of m to obtain ϕ_a.

It remains to show how to construct a sentence that defines the set of a-words that map to a given element s of M. Let $w \in A^*$ be an a-word. Let σ be the final factorization scheme in our sequence, so that

$$\sigma(w) = (w_1, \ldots, w_k), \quad \sigma_h(w) = m_1 \cdots m_k \in M^+.$$

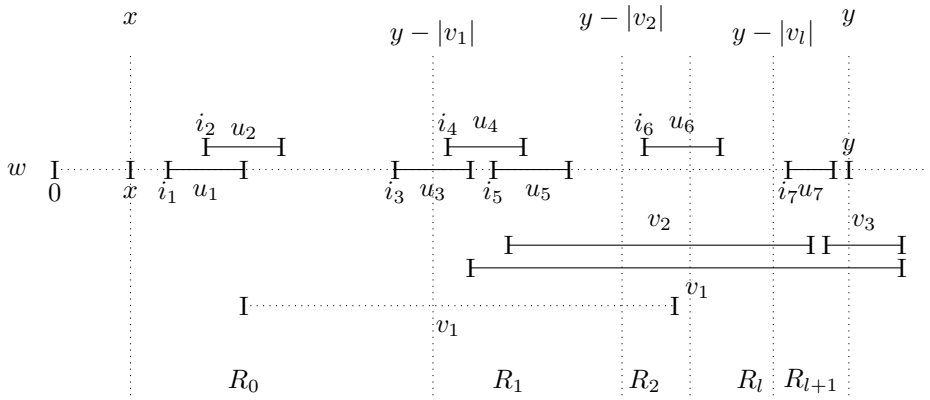

Figure 4 Occurrence sequence for model w, x: (1) Region R_1 starts at position $y - |v_1|$ where v_1 is the longest negative requirement. This means any negative factor v_i which starts in region R_0 will finish before y. Similarly, any negative factor v_i other than v_1 starting in R_2 will end before y. On the other hand, any v_1 starting after R_0 will necessarily end after y. (2) Positive requirements start in order $u_1 < u_2 \ldots < u_7$. Moreover, u_1, u_2, u_3 start in R_0, words u_4, u_5 start in R_1 and u_6 starts in R_2. Finally, u_7 starts in R_{l+1}.

In fact, each w_i can be mapped to the subalphabet

$$N = \{h(v) \in M : \alpha(v) = A, v \in aA^*\},$$

so we can restrict to this subalphabet N of M.

The map $n \mapsto n$ extends to a homomorphism from N^+ into the subsemigroup S of M generated by the elements of N. Since the generators of S are images of words v with $\alpha(v) = A$, we have $eSe \subseteq eM_e e$, which is in **DA** for every idempotent $e \in E(S)$ by definition of $\mathbf{M}_e\mathbf{DA}$. Locality of **DA** means that having all eSe in **DA**, the semigroup S is in $\mathbf{DA}{*}\mathbf{D}$. Thus the set of words over N multiplying to $s \in S$ is defined by a sentence ψ over N in $FO^2[<, \mathtt{succ}]$ [20]. We can take the conjunction of this with a sentence that says every letter belongs to the alphabet N, and thus obtain a sentence ψ' over M, also in $FO^2[<, Suc]$, defining this same set of words. Thus by the Simulation Lemma 4, there is a sentence ϕ in $FO^2[<, \mathtt{bet}]$ that defines the set of a-words that map to s. This completes the proof. ◀

7 A logic for intermediate occurrences of factors

As an extension of the techniques we developed, we add to two-variable logic 'betweenness' predicates $\langle u \rangle(x, y)$ for $u \in A^+$. If $u = a_1 \ldots a_n$, then

$$\langle u \rangle(x, y) = \exists z_1 \ldots \exists z_n (x < z_1 < \ldots z_n < y \wedge Suc(z_1, z_2) \wedge \cdots \wedge Suc(z_{n-1}, z_n) \wedge a_1(z_1) \wedge \cdots \wedge a_n(z_n)).$$

We call the logic $FO^2[<, \mathtt{betfac}]$. Its increased expressiveness does not translate to computational difficulty, which we will show by translation to temporal logic LTL[6]. For convenience, for $u = a_1 u_2 \ldots a_n$, we will abbreviate by u the LTL formula $a_1 \wedge \mathsf{X}(a_2 \wedge \cdots \wedge \mathsf{X}a_n)$.

▶ **Theorem 5.** *Satisfiability of $FO^2[<, \mathtt{betfac}]$ is Expspace-complete.*

Proof. In [8] we gave an *Expspace* lower bound for $FO^2[<, \mathtt{bet}]$, so we only have to give an *Expspace* upper bound. We give an exponential translation from an $FO^2[<, \mathtt{betfac}]$ sentence to temporal logic LTL, whose satisfiability is decidable in *Pspace* [16].

For a fixed betweenness predicate mentioning x and y in a $FO^2[<, \texttt{betfac}]$ sentence, consider all such predicates within the same scope, because they refer to the same x and y points. They may specify existence or non-existence requirements. Existence of a factor uvw implies the existence of a factor v and conversely for non-existence, we discard such implied requirements.

As an example of the interaction of these requirements, consider the positive requirements $a(x, y)$ and $b(x, y)$ and the negative requirement $\neg cacbc(x, y)$ on the word $cccccacbc$ where $x = 1$ and $y = 9$ are the first and last positions. All three requirements are satisfied, because the factor $cacbc$ is not present *strictly* between x and y. Order the negative requirements by length, without loss of generality we have $|v_1| > \cdots > |v_l|$ for negative requirements $\neg v_1(x, y), \ldots, \neg v_l(x, y)$. All these must be satisfied at the positions from $x + 1$ to $y - |v_1|$, all except $\neg v_1$ at positions from there upto $y - |v_2|$, and so on. We can express this by the formula Neg below:

$$(\neg v_1 \wedge \cdots \wedge \neg v_l)\mathsf{U}\big(\mathsf{X}^{|v_1|-1}y \wedge (\neg v_2 \wedge \cdots \wedge \neg v_l)\mathsf{U}_{|v_1|-|v_2|}\mathsf{X}^{|v_2|-1}y \wedge \ldots ((\neg v_l)\mathsf{U}_{|v_{l-1}|-|v_l|}\mathsf{X}^{|v_l|-1}y)\ldots\big),$$

where the *bounded untils* are defined by $p\mathsf{U}_i q = p \wedge \mathsf{X}(p\mathsf{U}_{i-1}q)$ and $p\mathsf{U}_0 q = q$. The subformulae $\mathsf{X}^{|v_1|-1}y, \mathsf{X}^{|v_2|-1}y, \ldots, \mathsf{X}^{|v_{l-1}|-1}y$ in Neg are redundant since they follow from the last $\mathsf{X}^{|v_l|-1}y$ and the durations of the *bounded untils*. We will develop this idea below.

Neg is not quite an LTL formula since y is a first-order variable. Abbreviate by N the formula $(\neg v_1 \wedge \cdots \wedge \neg v_l)$ to the left of the first *until* in Neg. We can write Neg more properly as $Neg(Q(y)) = N\mathsf{U}(Q(y))$ where we will replace $Q(y)$ later by a temporal formula.

There are also the positive requirements to satisfy. We take a disjunction over the possible orderings of positions where they are satisfied for the first time, which we abbreviate specifying where in three intervals $(x, y - |v_1|], (y - |v_1|, y - |v_l|], (y - |v_l|, y)$ they are to be placed. (The first two intervals are left-open and right-closed.) It follows from the fact that we have no implied factors that if the starting point of a factor is before the starting point of another, its ending point also precedes the ending point of the other.

$$\begin{aligned} O = u_1(x, y - |v_1|] < \cdots < u_k(x, y - |v_1|] < u_{k+1}(y - |v_1|, y - |v_l|] < \cdots < \\ u_{k+j}(y - |v_1|, y - |v_l|] < u_{k+j+1}(y - |v_l|, y - |u_{k+j+i}|] < \cdots < u_{k+j+i}(y - |v_l|, y - |u_{k+j+i}|]. \end{aligned}$$

More precisely there are $l + 1$ intervals to consider, by dividing up the middle interval $(y - |v_1|, y - |v_l|]$ into $l - 1$ subintervals as was done in formula Neg above.

The formula

$$Pos_0 = \neg u_1 \mathsf{U}(u_1 \wedge (\neg u_2 \mathsf{U}(u_2 \cdots \wedge (\neg u_k \mathsf{U}(u_k \wedge (true \mathsf{U} \mathsf{X}^{|u_k-1|}y)\ldots)))$$

takes care of the first block of requirements. This has to be interleaved to the left of the first *until* in Neg. That is, $Neg(Q(y)) = N\mathsf{U}(Q(y))$ is replaced by

$$Pos'_0(Q(y)) = (\neg u_1 \wedge N)\mathsf{U}\big(u_1 \wedge N \wedge ((\neg u_2 \wedge N)\mathsf{U}(u_2 \wedge N \wedge \cdots \wedge ((\neg u_k \wedge N)\mathsf{U}(u_k \wedge (N\mathsf{U}(Q(y)))))\ldots))\big).$$

Similarly the next j requirements have to be divided and interleaved with the *bounded untils* in the middle intervals in Neg, specified by formulae $Pos_1, \ldots, Pos_{l-1}$ in much the same manner, and the last i requirements specified by formula Pos_l, have to be interleaved with the last $|v_l| - 1$ *nexts* in Neg and updated to $Pos'_1(Q(y)), \ldots, Pos'_{l-1}(Q(y)), Pos'_l(Q(y))$ to form:

$$Neg' = Pos'_0(Pos'_1(\ldots(Pos'_{l-1}(Pos'_l(\mathsf{X}^{min(|v_l|,|u_{k+j+i}|)-1}y)))\ldots)).$$

The outcome of this interleaving procedure is that we have a formula having a *single* occurrence of the non-temporal variable y at the end. The size of this formula, for one ordering O, is polynomial in the size of the between requirements. The number of possible orderings O is exponential in the number of between requirements, $l + k + j + i$ above.

The technique of Etessami, Vardi and Wilke allows replacing the point y using its *type* [5], which produces an *LTL* formula. As argued by them, the complete *LTL* formula produced is exponential in terms of the sentence we started with. The exponentially many disjunctions produced by different orderings above compose with their procedure to give an exponential-sized formula. ◀

8 Characterization of $FO^2[<, \texttt{betfac}]$

The class of languages definable in the logic $FO^2[<, \texttt{betfac}]$ corresponds to a variety of finite semigroups rather than monoids. An operation which can be lifted to the level of semigroup and monoid classes is the semidirect product (which is not effective in general). We have obtained an *effective* algebraic characterization of $FO^2[<, \texttt{betfac}]$. Presenting the proof will require a detour into the algebraic theory of finite categories, so we will restrict ourselves here with the statement and the algebraic characterization, and reserve the proof of effectiveness for the full version of the paper.

▶ **Theorem 6** ($FO^2[<, \texttt{betfac}]$ characterizes $\mathbf{M}_e\mathbf{DA} * \mathbf{D}$). *Let $L \subseteq A^+$. L is definable in $FO^2[<, \texttt{betfac}]$ if and only if $S(L) \in \mathbf{M}_e\mathbf{DA} * \mathbf{D}$. Moreover, there is an effective procedure for determining if $S(L) \in \mathbf{M}_e\mathbf{DA} * \mathbf{D}$.*

Since $\mathbf{M}_e\mathbf{DA}$ contains $\Delta_3[<]$ in the quantifier alternation hierarchy [22], $\mathbf{M}_e\mathbf{DA} * \mathbf{D}$ contains $\Delta_3[<, Suc]$, which includes the language $BB_2 = (a(ab)^*b)^+$ which we showed in [8] was not in $\mathbf{M}_e\mathbf{DA}$. On the other hand it does not contain $BB_3 = (a(a(ab)^*b)^*b)^+$. Consider the language U_3 which is a sublanguage of $A^*c(a+b)^*cA^*$ such that between the marked c's, the factor bb does not occur before the factor aa. This is in $\mathbf{M}_e\mathbf{DA} * \mathbf{D}$ since it is defined by the $\Pi_2[<, Suc]$ sentence

$$\begin{aligned}\forall x \forall y \forall z \forall z' (\quad & c(x) \wedge c(y) \wedge x < z < z' < y \wedge Suc(z, z') \wedge b(z) \wedge b(z') \\ & \rightarrow \exists w \exists w' (x < w < w' < z \wedge Suc(w, w') \wedge a(w) \wedge a(w'))).\end{aligned}$$

The proof of the theorem, in both directions, depends on the characterization of $\mathbf{V} * \mathbf{D}$ in terms of $\mathbf{V}$ [17]. This can be stated in several different ways, but all depend on some scheme for treating words of length k over A as individual letters. Here is a standard version. Let $k > 0$. Let A be a finite alphabet, and let $B = A^k$. We treat B as a finite alphabet itself – to distinguish the *word* $w \in A^*$ of length k from the same object considered as a *letter* of B, we write $\{w\}$ in the latter case. We will define, for a word $w \in A^+$ with $|w| \geq k - 1$, a new word $w' \in B^*$, where w' is simply the sequence of length-k factors of w. So, for example, with $A = \{a, b\}$ and $k = 3$, if $w = aa$, then $w' = 1 \in B$, while if $w = ababba$, then

$$w' = \{aba\}\{bab\}\{abb\}\{bba\}.$$

To make sure that the lengths match up, we supplement A with a new symbol $*$ and define B' as $(A \cup \{*\})^k$, and w'' as the sequence of length-k factors of $*^{k-1}w$. For example, with this new definition, if $k = 3$ and $w = ababba$, then

$$w'' = \{**a\}\{*ab\}\{aba\}\{bab\}\{abb\}\{bba\}.$$

▶ **Theorem 7** (characterization of $\mathbf{V} * \mathbf{D}$ [17]). *Let $h : A^+ \to S$ be a homomorphism onto a finite semigroup. $S \in \mathbf{V} * \mathbf{D}$ if and only if there exist: an integer $k > 1$, and a homomorphism $h' : B^* \to M \in \mathbf{V}$, where $B = A^k$, such that whenever $v, w \in A^+$ are words that have the same prefix of length $k-1$, and the same suffix of length $k-1$, and v', w' are the sequence of k-length factors of v, w respectively, with $h'(v') = h'(w')$, then $h(v) = h(w)$.*

In brief, you can determine $h(w)$ by looking at the prefix and suffix of w of length $k-1$, and checking the value of w' under a homomorphism h' into an element of $\mathbf{V}$. Note that the statement is false if $\mathbf{V}$ is the trivial variety (and only in this case), but we can correct by replacing $\mathbf{D}$ in the statement by $\mathbf{LI}$.

In the full version of the paper we will show:

▶ **Proposition 8** (Delay). *Let ϕ be a sentence of $FO^2[<, \mathtt{betfac}]$. Then there exist $k > 1$ and a sentence ϕ' of $FO^2[<, \mathtt{bet}]$ interpreted over $(A \cup \{*\})^k$, with this property: if $w \in A^+$ with $|w| \geq k-1$, then $w \models \phi$ if and only if $w'' \models \phi'$.*

▶ **Proposition 9** (Expansion). *Let ϕ' be a sentence of $FO^2[<, \mathtt{bet}]$ interpreted over $(A \cup \{*\})^k$, where $k > 1$. Then there is a sentence ϕ of $FO^2[<, \mathtt{betfac}]$ with this property: if $w \in A^+$ with $|w| \geq k-1$, then $w \models \phi$ if and only if $w'' \models \phi'$.*

Proof of Characterization Theorem 6. Let $L \subseteq A^+$, and suppose that L is definable by a sentence ϕ of $FO^2[<, \mathtt{betfac}]$. Let $k > 1$ and ϕ' in $FO^2[<, \mathtt{bet}]$ be as given by Proposition 8. Let $L' \subseteq ((A \cup \{*\})^k)^*$ be the language defined by ϕ'. We will show that $S(L) \in \mathbf{M}_e\mathbf{DA} * \mathbf{D}$.

Let $h : A^+ \to S(L)$ be the syntactic morphism of L. Let h' be the syntactic morphism of L' and let h'' be the restriction of h' to elements of $(A^k)^*$. Since ϕ' is a sentence of $FO^2[<, \mathtt{bet}]$, the syntactic monoid of L', and hence the image of h'', belongs to $\mathbf{M}_e\mathbf{DA}$. It is therefore enough, in view of Theorem 7, to suppose that $v, w \in A^+$ have the same prefix of length $k-1$ and the same suffix of length $k-1$, and that $h''(v') = h''(w')$, and then conclude that $h(v) = h(w)$. To show $h(v) = h(w)$ we must show that for any $x, y \in A^*$, $xvy \in L$ if and only if $xwy \in L$. Given the symmetric nature of the statement, it is enough to show $xvy \in L$ implies $xwy \in L$. So let $xvy \in L$. Then $xvy \models \phi$, so $(xvy)'' \models \phi'$. We take apart $(xvy)''$: Suppose $x = a_1 \cdots a_r, v = b_1 \cdots b_s, y = c_1 \cdots c_t$.

The leftmost $r + k - 1$ letters of $(xvy)''$ are

$$\{*^{k-1}a_1\}\{*^{k-2}a_1a_2\} \cdots \{a_r b_1 \cdots b_{k-1}\}.$$

The rightmost t letters of $(xvy)''$ are

$$\{b_{s-k+2} \cdots b_s c_1\}\{b_{s-k+3} \cdots b_s c_1 c_2\} \cdots \{c_{t-k+1} \cdots c_t\}.$$

(The exact form of the last factor will be different if $t < k-1$.) In between these two factors, we have the $s - k + 1$ letters of v'. Thus $h'((xvy)'') = m_1 h''(v') m_2$, where m_1, m_2 depend only on x, y and the prefix and suffix of v of length at most $k-1$. It follows that we likewise have $h'((xwy)'') = m_1 h''(w') m_2$, with the same m_1, m_2. Since $h''(v') = h''(w')$, we conclude $h'((xvy)'') = h'((xwy)'')$, so $(xwy)'' \models \phi'$. Thus $xwy \models \phi$, and so $xwy \in L$. This concludes the proof that $S(L) \in \mathbf{M}_e\mathbf{DA} * \mathbf{D}$.

Conversely, suppose $L \subseteq A^+$ and that $S(L) \in \mathbf{M}_e\mathbf{DA} * \mathbf{D}$. Let $h : A^+ \to S(L)$ be the syntactic morphism of L. Let $h' : (A^k)^* \to M \in \mathbf{M}_e\mathbf{DA}$ be the homomorphism given by Theorem 7. We extend h' to $((A \cup \{*\})^k)^*$ by defining $h'(b) = 1$ for any b that contains the new symbol $*$. Then for each $m \in M$, we have a sentence ϕ'_m of $FO^2[<, \mathtt{bet}]$ interpreted over $((A \cup \{*\})^k)^*$ defining $(h')^{-1}(m)$. Let ϕ_m be the sentence over $FO^2[<, \mathtt{betfac}]$ given

by Proposition 9. For each $x \in A^{k-1}$, let pref_x be a sentence defining the set of strings over A whose prefix of length $k-1$ is x, and similarly define suff_x. Observe that both of these sentences can be chosen to be in $FO^2[<, \mathtt{betfac}]$. In fact, these properties are definable in $FO^2[<, \mathtt{bet}]$ over A. It follows that the set of words in A^+ of length at least $k-1$ mapping to a given value s of $S(L)$ is given by a disjunction of finitely many sentences of the form

$$\mathrm{pref}_x \wedge \mathrm{suff}_y \wedge \phi'_m.$$

We thus get the complete preimage $h^{-1}(s)$ by taking the disjunction with a sentence that says the word lies in a particular finite set. So L itself is definable in $FO^2[<, \mathtt{betfac}]$. ◀

References

1 Jorge Almeida. A syntactical proof of the locality of DA. *Int. J. Alg. Comput.*, 6:165–177, 1996.

2 Janusz Brzozowski. A generalization of finiteness. *Semigr. Forum*, 13:239–251, 1977.

3 Rina Cohen and Janusz Brzozowski. Dot-depth of star-free events. *J. Comput. Syst. Sci.*, 5(1):1–16, 1971.

4 Volker Diekert, Paul Gastin, and Manfred Kufleitner. First-order logic over finite words. *Int. J. Found. Comp. Sci.*, 19:513–548, 2008.

5 Kousha Etessami, Moshe Vardi, and Thomas Wilke. First-order logic with two variables and unary temporal logic. *Inf. Comput.*, 179(2):279–295, 2002.

6 Johan Anthony Willem Kamp. Tense logic and the theory of linear order. UCLA, 1968. PhD thesis.

7 Robert Knast. A semigroup characterization of dot-depth one languages. *Inf. Theor. Appl.*, 17(4):321–330, 1983.

8 Andreas Krebs, Kamal Lodaya, Paritosh Pandya, and Howard Straubing. Two-variable logic with a between relation. In Martin Grohe, Erik Koskinen, and Natarajan Shankar, editors, *Proc. 31st LICS, New York*, pages 106–115. ACM/IEEE, 2016.

9 Robert McNaughton and Seymour Papert. *Counter-free automata.* MIT Press, 1971.

10 Jean-Éric Pin. *Varieties of formal languages.* Plenum, 1986.

11 Thomas Place and Luc Segoufin. Decidable characterization of $fo^2(<, +1)$ and locality of DA. Preprint, ENS Cachan, 2014.

12 Thomas Place and Marc Zeitoun. Going higher in the first-order quantifier alternation hierarchy on words. In Javier Esparza, Pierre Fraigniaud, Thore Husfeldt, and Elias Koutsoupias, editors, *Proc. 41st Icalp, Part 2, Copenhagen*, volume 8573 of *LNCS*, pages 342–353, 2014.

13 Thomas Place and Marc Zeitoun. Separation and the successor relation. In Ernst W. Mayr and Nicolas Ollinger, editors, *Proc. 32nd Stacs, Garching*, volume 30 of *Lipics*, pages 662–675, 2015.

14 Marcel-Paul Schützenberger. On finite monoids having only trivial subgroups. *Inf. Contr.*, 8:190–194, 1965.

15 Marcel-Paul Schützenberger. Sur le produit de concaténation non ambigu. *Semigr. Forum*, 13:47–75, 1976.

16 A. Prasad Sistla and Edmund Clarke. The complexity of propositional linear temporal logics. *J. ACM*, 32(3):733–749, 1985.

17 Howard Straubing. Finite semigroup varieties of the form V*D. *J. Pure Appl. Alg.*, 36:53–94, 1985.

18 Howard Straubing. *Finite automata, formal languages, and circuit complexity.* Birkhäuser, 1994.

19 Pascal Tesson and Denis Thérien. Logic meets algebra: the case of regular languages. *Log. Meth. Comp. Sci.*, 3(1:4):1–37, 2007.

20 Denis Thérien and Thomas Wilke. Over words, two variables are as powerful as one quantifier alternation. In Jeffrey Vitter, editor, *Proc. 30th STOC, Dallas*, pages 234–240. ACM, 1998.

21 Bret Tilson. Categories as algebra. *J. Pure Appl. Alg.*, 48:83–198, 1987.

22 Pascal Weil. Some results on the dot-depth hierarchy. *Semigr. Forum*, 46:352–370, 1993.