Block products for algebras over countable words and applications to logic

Bharat Adsul
IIT Bombay, India
Email: adsul@cse.iitb.ac.in

Saptarshi Sarkar
IIT Bombay, India
Email: sapta@cse.iitb.ac.in

A. V. Sreejith
IIT Goa, India
Email: sreejithav@iitgoa.ac.in

***Abstract*—We propose a seamless integration of the block product operation to the recently developed algebraic framework for regular languages of countable words. A simple but subtle accompanying block product principle has been established. Building on this, we generalize the well-known algebraic characterizations of first-order logic (resp. first-order logic with two variables) in terms of strongly (resp. weakly) iterated block products. We use this to arrive at a complete analogue of Schützenberger-McNaughton-Papert theorem for countable words. We also explicate the role of block products for linear temporal logic by formulating a novel algebraic characterization of a natural fragment.**

I. Introduction

The seminal works of Büchi and Elgot (see the survey article [23]) established fundamental connections between languages, automata and logics. This was achieved via effective back-and-forth translations between finite-state automata and monadic second-order (MSO) logic over finite words. These connections were enriched with the introduction of algebraic structures such as the syntactic monoids/semigroups (see [12], [13]). The celebrated theorems of Schützenberger [17] and McNaughton-Papert [11] showed the equivalence between star-free regular languages, first-order (FO) logic and recognizability by finite *aperiodic* monoids. Many such equivalences have been established for 'small' fragments of FO. See [7] for a survey.

One of the central tools in the algebraic theory of monoids is the notion of a block product and its related cousin - a wreath product. The classical Krohn-Rhodes decomposition theorem (see [18]) asserts that every finite monoid divides (can be simulated by) block/wreath products of 'simple' monoids. As a special case, every finite aperiodic monoid divides block products of a simple canonical two-element aperiodic monoid called U_1. In other words, the simple monoid U_1 serves as a basis for generating all aperiodic monoids under the block product operation!

The famous works of Büchi and McNaughton (see [23]) introduced more intricate automata models working over ω-words and extended the automata-logic connection to ω-words. It was established in [25] that the right algebraic structures in this context are ω-semigroups and Wilke algebras.

Linear temporal logics (LTL) [8] provide a very convenient formalism for specifying properties of words. They typically admit decision procedures of low complexity (unlike FO/MSO) and have high expressive power (LTL has the same expressive power as FO over finite and ω-words by Kamp's theorem). These properties have contributed to the wide-spread use of temporal logics in formal verification. Underlying these applications are efficient translations from temporal logic specifications to automata.

978-1-7281-3608-0/19/$31.00

The algebraic approach has also been very useful in understanding the expressive power of LTL operators such as since and until. For instance, [22] provides an effective algebraic characterization (in terms of block products) of the until-since hierarchy. See the survey [20] for many such results.

Going beyond finite and infinite words, Rabin [14] already showed that decidability of MSO over *countable words* (labelled linear orders with countable domains) can be obtained using his deep result on the decidability of MSO over infinite trees. However, an appropriate automaton model naturally working over countable words was missing. There have been some earlier works extending the automata-logic connection to labelled linear orderings beyond finite and infinite words. More recently, [3], [15], [1] showed such connections for countable scattered words that is, labelled countable linear orders without a dense subset. These works introduced appropriate automata, rational expressions and algebras and showed that they all equal MSO in expressive power when restricted to countable scattered words.

In a recent breakthrough, [5] established an algebraic approach to MSO-definability over countable linear orderings. It also showed very impressive applications to logic. One such is the first known (optimal) collapse of the (set)-quantifier alternation hierarchy of MSO to its second level. This study restores the beautiful automata-logic connections from the well-understood settings to the setting of countable words, albeit in the form of an algebra-logic connection. Further, it opens up the possibility of characterizing various sublogics (and their natural fragments) via algebraic means. An elaborate study over a variety of sublogics of MSO over countable words was carried out in [6] where FO, weak MSO etc. were characterized algebraically. It is important to note that all these characterizations are decidable. In a similar vein, a decidable algebraic characterization of the two variable fragment of FO was obtained in [10].

Our investigations in this work are directed towards enriching the algebraic framework of [5]. Motivated by the decisive role played by block products in the standard settings, we introduce block products in the countable setting. The block product construction in the standard setting associates to a pair of monoids (M, N) a new monoid $M\Box N$. From a formal-language theoretic viewpoint, the importance of the block product construction is best described by the accompanying block product principle. Roughly speaking the block product principle (see [18], [13]) says that evaluating a *finite* word u in $M\Box N$ can be achieved by the following two-stage process: 1) evaluate the word u *in* M and label every position x of u with the additional information about evaluations of $u_{<x}$ and $u_{>x}$ *in* M where $u_{<x}$ and $u_{>x}$ are such that $u = u_{<x}u[x]u_{>x}$ (that is, $u_{<x}$ and $u_{>x}$ are the left and right factors/contexts at position x); 2) now evaluate this extended word (with the additional information) *in* N. Said differently, M 'operates' on u as usual; while when N 'operates' on u, it has access, at *every* position, to evaluations of M on left-right contexts at that position.

Our generalization of the block product operation and the accompanying block product principle extend this intuitively appealing 'operational' description from finite words to countable words. More specifically, we work with the central algebraic objects in [5] such as *finite* $\circledast$-monoids and $\circledast$-algebras. As shown in [5], a $\circledast$-algebra is an effective version of a $\circledast$-monoid and captures its essence through finitely presentable operators.

As $\circledast$-monoids allow evaluations of *arbitrary* countable words, we first define the block product operation for $\circledast$-monoids. We further show that this operation descends to the level of $\circledast$-algebras and remains effective.

Other sources of difficulties when working with countable words are dense subsets and the presence of gaps in the underlying linear ordering. A gap in a countable word u is a factorization of u as u_1u_2 such that the word u_1 (resp. u_2) has no last (resp. first) position. The above mentioned (point-based) operational description of the block product principle is oblivious to left-right contexts at gaps. In spite of this, we provide first algebraic characterizations of some natural logics in terms of the newly defined block products.

Our first application is in the context of first-order logic. We provide generalizations of the well-known algebraic characterizations of FO (resp. FO^2, first-order logic with two variables) in terms of strongly (resp. weakly) iterated block products of a natural two-element $\circledast$-algebra. We crucially use this result to show the equivalence between languages definable in FO and those which admit *marked* star-free regular expressions. Combining our result with an *equational* algebraic characterization of FO from [6], we get a complete analogue of Schützenberger-McNaughton-Papert theorem for countable words. An interesting consequence of these results is a suitable version of the aforementioned special case of the Krohn-Rhodes theorem.

Our next application concerns LTL. It is known [9] that over countable words, Kamp's theorem does not hold! That is, LTL[S,U] (with natural strict since-until temporal operators) is not as expressive as FO. We provide a very natural characterization of LTL[S,U] in terms of *weakly* iterated block products of two canonical $\circledast$-algebras. This result is delicate as the weak iteration of the *standard* block product (of monoids) of the *underlying monoids* of these two $\circledast$-algebras is as expressive as FO over finite words! This supports our hypothesis that the block product construction proposed in this work will allow a fine-grained distinction of several natural fragments of LTL.

Structure of the paper: In the next section we define countable linear orderings and countable words, followed by a section on $\circledast$-monoids and the block product operation on them. We end the section with the block product principle. Section IV talks about the application of block product principle to first order logic and its two variable sub-class. Finally, in section V we characterize an interesting subclass of linear temporal logic.

II. Countable words

A *linear ordering* $\alpha = (X, <)$ is a set X equipped with a total order $<$. The ordering is called *countable* if X is countable. For example, the reals $(\mathbb{R}, <)$ with the less than relation form a linear ordering which is not countable while the natural numbers $(\mathbb{N}, <)$, the rationals $(\mathbb{Q}, <)$ are countable linear orderings. Two linear orderings are of the same *order type* if there is an order-preserving bijection between them. We denote by ω, ω^* and η the order types of $(\mathbb{N}, <), (-\mathbb{N}, <)$ and $(\mathbb{Q}, <)$ respectively. To ease use of notations, we'll sometimes use an ordering and its order type interchangeably.

We say $(I, <')$ is an *induced subordering* of a linear ordering α if I is a subset of α and $<'$ is the restriction of the ordering in α to I. We denote the induced subordering on the subset I by α_I. Given two subsets I, J of α, we say that $I < J$ if $x < y$ for all x in I and all y in J. A subset I of α is called *convex* if for any $x < z < y$ of α, $x, y \in I$ implies $z \in I$. A set I is a *prefix* of $\alpha = (X, <)$ if $X = I \cup J$ for some $J \subseteq X$ and $I < J$. Similarly the subset J is called a *suffix* of α.

The sum $\alpha_1 + \alpha_2$ of two linear orderings $\alpha_1 = (X_1, <_1)$ and $\alpha_2 = (X_2, <_2)$ is the linear ordering $(X_1 \cup X_2, <)$ where $<$ coincides with $<_1$ on X_1, with $<_2$ on X_2 and $X_1 < X_2$. We assume, upto renaming, the sets X_1 and X_2 to be disjoint. This notion can be generalized to sum of countably many linear orderings $\alpha_i = (X_i, <_i)$ which are themselves indexed by some linear ordering $\alpha = (X, <)$. The result of this sum is the linear ordering $(Y, <')$ where $Y = \bigcup_{i\in\alpha} X_i$ and for any two points $x, y \in Y$, $x <' y$ if $x <_i y$ or $x \in X_i$ and $y \in X_j$ and $i < j$.

A subordering $(I, <)$ of α is *dense* in α if for any two points $x, y \in \alpha$, there exists $z \in I$ such that $x < z < y$. For example, $(\mathbb{Q}, <)$ is dense in $(\mathbb{R}, <)$ and $(\mathbb{R}, <)$ is dense in itself. If a linear ordering is dense in itself, we simply call it dense. A linear ordering is called *scattered* if all its dense suborderings are singleton or empty. A linear ordering α is *right-open* (resp. *left-open*) if it has no maximum element (resp. minimum element). For example, $(\mathbb{Z}, <)$ is a scattered linear ordering that is both right-open as well as left-open.

Refer to [16] for an in-depth study of linear orderings.

A countable word w is a labelled countable linear ordering. More formally, given a finite alphabet Σ and a countable linear ordering α, a countable word w is a map $w : \alpha \to \Sigma$. In this paper, we mainly deal with countable words. So a word, unless explicitly stated otherwise, will denote a countable word. We call α the *domain* of w, denoted $\mathsf{dom}(w)$. For a word w, we say a point or position x in the word to refer to an element of its domain. The notation $w[x]$ denotes the letter at the x^{th} position in the word w. A *subword* is a restriction of a word w to some induced subordering I of its domain, and is denoted by w_I. If the subset I is convex, then the subword is called a *factor*. If I is a prefix (resp. suffix) of $\mathsf{dom}(w)$, then w_I is called a *prefix* (resp. *suffix*) of the word w. In particular, for any position $x \in \mathsf{dom}(w)$, $w_{<x}$ (resp. $w_{>x}$) refers to the prefix (resp, suffix) w_I where I is the set of all positions less than (resp. greater than) x.

If u, v are words, then their *concatenation*, denoted uv, is the unique word w where u (resp. v) is a prefix (resp. suffix) of w and $\mathsf{dom}(w) = \mathsf{dom}(u) + \mathsf{dom}(v)$. This operation can be generalized to a countable sequence of words $\{w_i\}_{i\in\alpha}$ indexed by a linear countable ordering α as $\prod_{i\in\alpha} w_i = w$ where $\mathsf{dom}(w)$ is the sum of the domains of w_i for all i and $w[x] = w_i[x]$ if $x \in \mathsf{dom}(w_i)$.

The following countable words are of special interest. The notation ε stands for the *empty word* (the word over the empty domain). The *omega word*, a^ω denotes the word over the domain $(\mathbb{N}, <)$ such that every position is mapped to the letter a. Similarly, the *omega* word* a^{ω^*} denotes the word over the domain $(-\mathbb{N}, <)$ where every position is mapped to letter a. A *perfect shuffle* over a nonempty set $P \subseteq \Sigma$ of letters, denoted by P^η, is the word w over domain $(\mathbb{Q}, <)$ such that $w[x] \in P$ for all positions x in $\mathsf{dom}(w)$ and the set $w^{-1}(a) = \{x \in \mathsf{dom}(w) \mid w[x] = a\}$ for each letter $a \in P$ is dense in $\mathsf{dom}(w)$. This is a unique word (up to isomorphism) (see [5]) and is an example of a dense word, i.e. a word whose domain is dense.

For an alphabet Σ, the set of all countable words is denoted by $\Sigma^{\circledast}$ and the set of all countable words over non-empty domain is denoted by $\Sigma^{\oplus}$. A *language* over the alphabet Σ is a subset of $\Sigma^{\circledast}$.

III. Block Product

A. Countable Products, Semigroups and Algebras

We recall the algebraic framework from [5] along with some technical definitions/notions which are useful for this work.

A $\oplus$-*semigroup* (S, π) consists of a set S with an operation $\pi : S^{\oplus} \to S$ such that, $\pi(a) = a$ for all $a \in S$ and π satisfies the *generalized associativity property* – that is $\pi\big(\prod_{i\in\alpha} u_i\big) = \pi\big(\prod_{i\in\alpha} \pi(u_i)\big)$ for every countable linear ordering α. If the generalized associativity holds with $\pi : S^{\circledast} \to S$, then (S, π) is called a $\circledast$-monoid.

For a set Σ, $(\Sigma^{\oplus}, \Pi)$ (resp. $(\Sigma^{\circledast}, \Pi)$) is the *free* $\oplus$-semigroup (resp. free $\circledast$-monoid) generated by Σ.

A *neutral element* of a $\oplus$-semigroup (S, π) is an element $1 \in S$ such that for every $w \in S^{\oplus}$, $\pi(w) = \pi(w_{\neq 1})$ if $w_{\neq 1}$ is non-empty. Here $w_{\neq 1}$ is w restricted to positions i where $w[i] \neq 1$. If a neutral element exists, it is unique. A $\circledast$-monoid (S, π) admits $\pi(\varepsilon)$ as the unique neutral element.

Clearly, a $\circledast$-monoid can be naturally viewed as a $\oplus$-semigroup. Further, a $\oplus$-semigroup S can be easily extended to a $\circledast$-monoid (denoted S^1) by introducing an additional neutral element if necessary. Thanks to this, any result for $\oplus$-semigroups has a suitable analogue for $\circledast$-monoids. In view of this, most of the technical results in this work are simply stated and proved for $\oplus$-semigroups.

Example 1. $\mathrm{U}_1 = (\{1, 0\}, \pi)$ *is a* $\circledast$*-monoid where* π *is defined for all* $u \in \{1, 0\}^{\circledast}$ *as:*

$$\pi(u) = \begin{cases} 1 & \textit{if } u \in \{1\}^{\circledast} \\ 0 & \textit{otherwise} \end{cases}$$

Let (S, π) be a $\oplus$-semigroup. Even if S is finite, π need not be finitely presentable and hence not suitable for algorithmic purposes. Fortunately, it is possible to capture π through finitely presentable operators. This is precisely the reason for introducing $\oplus$-algebras.

A $\oplus$-*algebra* $(S, \cdot, \tau, \tau^*, \kappa)$ consists of a set S with $\cdot : S^2 \to S, \tau : S \to S, \tau^* : S \to S, \kappa : \mathcal{P}(S) \setminus \{\emptyset\} \to S$ such that (with infix notation for $\cdot$ and superscript notation for τ, τ^*, κ)

- A-1 $(S, \cdot)$ is a semigroup.
- A-2 $(a \cdot b)^\tau = a \cdot (b \cdot a)^\tau$ and $(a^n)^\tau = a^\tau$ for $a, b \in S$ and $n > 0$.
- A-3 $(b \cdot a)^{\tau^*} = (a \cdot b)^{\tau^*} \cdot a$ and $(a^n)^{\tau^*} = a^{\tau^*}$ for $a, b \in S$ and $n > 0$.
- A-4 For every non-empty subset P of S, every element c in P, every subset P' of P, and every non-empty subset P'' of $\{P^\kappa, a.P^\kappa, P^\kappa.b, a.P^\kappa.b \mid a, b \in P\}$, we have $P^\kappa = P^\kappa.P^\kappa = P^\kappa.c.P^\kappa = (P^\kappa)^\tau = (P^\kappa.c)^\tau = (P^\kappa)^{\tau^*} = (c.P^\kappa)^{\tau^*} = (P' \cup P'')^\kappa$.

A $\circledast$-*algebra* is a $\oplus$-algebra with a special element 1 where $(S, \cdot, 1)$ is a monoid, $1^\tau = 1^{\tau^*} = \{1\}^\kappa = 1$ and for all non-empty subsets $P \subseteq \mathcal{P}(S)$, $P^\kappa = (P \cup \{1\})^\kappa$.

Example 2. *The* $\circledast$*-algebra induced by* U_1 *is given below. It plays a crucial role in this work and will also be denoted by* U_1.

	1	0	τ	τ^*
1	1	0	1	1
0	0	0	0	0

Fig. 1. The $\circledast$-algebra U_1. We also have $0^\kappa = \{1, 0\}^\kappa = 0$ and $1^\kappa = 1$

It is shown in [5] that a $\oplus$-semigroup ($\circledast$-monoid) naturally induces a $\oplus$-algebra ($\circledast$-algebra respectively). We simply set $a \cdot b = \pi(ab)$, $a^\tau = \pi(a^\omega), a^{\tau^*} = \pi(a^{\omega^*})$ and $P^\kappa = \pi(P^\eta)$ (recall that P^η is the unique perfect shuffle word over P). It is also shown there that a finite $\oplus$-algebra has a unique extension to a finite $\oplus$-semigroup. This is one of the fundamental results of [5] which we make use of.

Theorem 1. *A* $\oplus$*-semigroup* (S, π) *induces a unique* $\oplus$*-algebra. Also, any finite* $\oplus$*-algebra is induced by a unique* $\oplus$*-semigroup.*

Now we briefly discuss some natural algebraic notions. Let (S, π) and (S', π') be $\oplus$-semigroups. A morphism from (S, π) to (S', π') is a map $h : S \to S'$ such that, for every $w \in S^{\oplus}$, $h(\pi(w)) = \pi'(\bar{h}(w))$ where $\bar{h}$ is the pointwise extension of h to words. A $\oplus$-*subsemigroup* of (S, π) is a subset $\bar{S} \subseteq S$ such that

π restricts from $\bar{S}^{\oplus}$ to $\bar{S}$. We simply denote this by $(\bar{S}, \pi) \subseteq (S, \pi)$. We say (S, π) *divides* (S', π') if there exists $(S'', \pi') \subseteq (S', \pi')$ and a surjective morphism from (S'', π') to (S, π). We write $(S, \pi) \preceq (S', \pi')$ to denote that (S, π) *divides* (S', π'). The notions of morphism, subsemigroup (subalgebra), division are also defined for algebras along expected lines. Further, they are naturally compatible with those of $\oplus$-semigroups.

B. Semidirect Product Construction

The *direct product* is an important standard construction in algebra. It is straightforward to adapt this notion to $\oplus$-semigroups and $\oplus$-algebras. In this section, we propose a generalization of semidirect product from semigroups (see [18], [13]) to $\oplus$-semigroups. More accurately, we generalize the so called 2-sided semidirect product construction. We note that [4] provides a 1-sided semidirect product for ω-semigroups and uses it to define the wreath products thereof.

In this work, we will be working exclusively with 2-sided semidirect products and we simply refer to them as semidirect products.

We begin by introducing the setup of two commuting actions of a $\oplus$-semigroup on another.

Consider two $\oplus$-semigroups (M, π) and $(N, \hat{\pi})$. A function $\delta_l : M^1 \times N \to N$ is said to be a left action of M on N if it satisfies the following conditions (for notational convenience, $\delta_l(m, n)$ is denoted by $m * n$).

C-1 $\quad 1 * n = n$
C-2 $\quad (\pi(m_1 m_2)) * n = m_1 * (m_2 * n)$
C-3 $\quad m * (\hat{\pi}(\Pi_{i \in \alpha} n_i)) = \hat{\pi}(\Pi_{i \in \alpha} m * n_i)$

The right action $\delta_r : N \times M^1 \to N$ is defined similarly and $\delta_r(n, m)$ is denoted by $n * m$.

C-4 $\quad n * 1 = n$
C-5 $\quad n * (\pi(m_1 m_2)) = (n * m_1) * m_2$
C-6 $\quad (\hat{\pi}(\Pi_{i \in \alpha} n_i)) * m = \hat{\pi}(\Pi_{i \in \alpha} n_i * m)$

Actions δ_l and δ_r are compatible with each other if they satisfy the following condition.

C-7 $\quad (m_1 * n) * m_2 = m_1 * (n * m_2)$

For $m \in M$, define as $\delta_l^m : N \to N$ as $\delta_l^m(n) = m * n$. By abuse of notation, the natural pointwise extension of δ_l^m from $N^{\oplus}$ to itself will also be denoted by δ_l^m. The above conditions for the left action may be stated simply by saying that, for each $m \in M$, δ_l^m is a morphism of $\oplus$-semigroups and for $m_1, m_2 \in M$, the morphism $\delta_l^{m_1 \cdot m_2}$ is the composition of $\delta_l^{m_1}$ and $\delta_l^{m_2}$. In other words, a left action is specified by a *monoid* morphism from M^1 into the monoid of $\oplus$-semigroup endomorphisms of N. A similar remark applies to the right action and these two actions commute.

Suppose M and N are $\circledast$-monoids with neutral elements 1 and $\hat{1}$ respectively. We say that δ_l is monoidal if $m * \hat{1} = \hat{1}\ \forall m \in M$. Similarly, the right action δ_r is monoidal if $\hat{1} * m = \hat{1}\ \forall m \in M$.

Before defining the semidirect product, we need to introduce some notations. Consider a word $u \in (M \times N)^{\oplus}$ with domain α being the underlying linear ordering. To the word u, we associate two words $v \in M^{\oplus}$ and $w \in N^{\oplus}$ with domain α. We begin by defining $u_1 \in M^{\oplus}$ and $u_2 \in N^{\oplus}$ (with domain α) as simply the 'projections' of u on M and N respectively. In other words, for $x \in \alpha$, $u_1[x] = m$ and $u_2[x] = n$ if $u[x] = (m, n)$. The word v is simply u_1. On the other hand, the word w is derived from u_2 by making use of the evaluations of π of M on appropriate 'point-based-factors' of u_1 and their actions on N. Precisely, for $x \in \alpha$, $w[x] = \pi(u_{1_{<x}}) * u_2[x] * \pi(u_{1_{>x}})$.

Definition 1. *The map $\theta : (M \times N)^{\oplus} \to M^{\oplus} \times N^{\oplus}$ is defined as: $\theta(u) = (v, w)$.*

The following simple lemma records a useful property of the map θ. Its proof is straightforward.

Lemma 1. *Suppose $u = \Pi_{i \in \alpha} u_i$ with $\theta(u) = (v, w)$ and, for $i \in \alpha$, $\theta(u_i) = (v_i, w_i)$. Then $v = \Pi_{i \in \alpha} v_i$ and $w = \Pi_{i \in \alpha} w'_i$ where, for $i \in \alpha$*

$$w'_i = \pi\left(\Pi_{j \in \alpha : j < i} v_j\right) * w_i * \pi\left(\Pi_{j \in \alpha : j > i} v_j\right)$$

Now we 'define' the semidirect product.

Definition 2 (Semidirect Product)**.** *Given a $\oplus$-semigroup (M, π) with compatible left and right actions on another $\oplus$-semigroup $(N, \hat{\pi})$, their semidirect product $(M \ltimes N, \tilde{\pi})$ has underlying set $M \times N$ and $\tilde{\pi} : (M \ltimes N)^{\oplus} \to (M \ltimes N)$ is defined as $\tilde{\pi}(u) = (\pi(v), \hat{\pi}(w))$ where $\theta(u) = (v, w)$.*

The proof of the following lemma verifies that $M \ltimes N$ is indeed a $\oplus$-semigroup by showing that $\tilde{\pi}$ satisfies the generalized associativity property.

Lemma 2. *The structure $(M \ltimes N, \tilde{\pi})$ is a $\oplus$-semigroup.*

Proof: Let $u = \Pi_{i \in \alpha} u_i$ where $u, u_i \in (M \ltimes N)^{\oplus}$. We have to prove $\tilde{\pi}(u) = \tilde{\pi}(\Pi_{i \in \alpha} \tilde{\pi}(u_i))$. Rewriting $\Pi_{i \in \alpha} \tilde{\pi}(u_i)$ as z, we have to prove $\tilde{\pi}(u) = \tilde{\pi}(z)$.

Suppose $\theta(u) = (v, w)$ and for $i \in \alpha$, $\theta(u_i) = (v_i, w_i)$. Then by Lemma 1, $v = \Pi_{i \in \alpha} v_i$ and $w = \Pi_{i \in \alpha} w'_i$ where w'_i is as given in the lemma statement. By the definition of semidirect product, $\tilde{\pi}(u) = (\pi(v), \hat{\pi}(w))$. Using the generalized associativity properties of π and $\hat{\pi}$, we can rewrite this as $\tilde{\pi}(u) = (\pi(\Pi_{i \in \alpha} \pi(v_i)), \hat{\pi}(\Pi_{i \in \alpha} \hat{\pi}(w'_i)))$.

Next we analyse the word z. Note that $\mathsf{dom}(z) = \alpha$ and $z[i] = \tilde{\pi}(u_i)$. Further, recall that $\theta(u_i) = (v_i, w_i)$. From Definition 2, we get $\tilde{\pi}(u_i) = (\pi(v_i), \hat{\pi}(w_i))$. So $z[i] = (\pi(v_i), \hat{\pi}(w_i))$. We now compute $\theta(z)$ using Definition 1. Let $\theta(z) = (z', z'')$. It is easy to see that $z'[i] = \pi(v_i)$. Using this, we see that $z''[i]$ equals

$$\begin{aligned}
&= \pi(\Pi_{j<i} \pi(v_j)) * \hat{\pi}(w_i) * \pi(\Pi_{j>i} \pi(v_j)) \\
&= \hat{\pi}\left(\pi(\Pi_{j<i} \pi(v_j)) * w_i * \pi(\Pi_{j>i} \pi(v_j))\right) && \text{[by C-3, C-6, C-7]} \\
&= \hat{\pi}\left(\pi(\Pi_{j<i} v_j) * w_i * \pi(\Pi_{j>i} v_j)\right) && \text{[by gen. assoc. of } \pi] \\
&= \hat{\pi}(w'_i))
\end{aligned}$$

The last line follows from the previous one by the explicit expression for w'_i in Lemma 1. Further note that we have really used the pointwise-extension of the actions when going from the first line to the second.

Now we proceed with the computation of $\tilde{\pi}(z)$ by using Definition 2 of semidirect product.

$$\begin{aligned}\tilde{\pi}(z) &= (\pi(z'), \hat{\pi}(z''))\\ &= (\pi(\Pi_{i\in\alpha} z'[i]), \hat{\pi}(\Pi_{i\in\alpha} z''[i]))\\ &= (\pi(\Pi_{i\in\alpha} \pi(v_i)), \hat{\pi}(\Pi_{i\in\alpha} \hat{\pi}(w'_i)))\end{aligned}$$

Comparing this with the expression for $\tilde{\pi}(u)$ derived earlier, we see that $\tilde{\pi}(u) = \tilde{\pi}(z)$. This completes the proof. ∎

We remark here that if M and N are $\circledast$-monoids (with neutral elements 1 and $\hat{1}$ respectively) and the actions are monoidal, then $M \ltimes N$ is a $\circledast$-monoid with $(1, \hat{1})$ as the neutral element.

Having shown $M \ltimes N$ to be a valid $\oplus$-semigroup, we now turn to the problem of the construction of semidirect product for *finite* $\oplus$-algebras. Thanks to Theorem 1, we can restrict our attention to *induced* $\oplus$-algebras.

Towards this, let $\mathbf{M} = (M, \cdot, \tau, \tau^*, \kappa)$ and $\mathbf{N} = (N, \hat{\cdot}, \hat{\tau}, \hat{\tau}^*, \hat{\kappa})$ be $\oplus$-algebras induced by $\oplus$-semigroups (M, π) and $(N, \hat{\pi})$ respectively. Further, let $\mathbf{M} \ltimes \mathbf{N} = (M \times N, \tilde{\cdot}, \tilde{\tau}, \tilde{\tau}^*, \tilde{\kappa})$ denote the $\oplus$-algebra induced by $(M \ltimes N, \tilde{\pi})$. Note that, by Theorem 1, the induced algebra operators of $\mathbf{M} \ltimes \mathbf{N}$ satisfy the four axioms mentioned in the definition of a $\oplus$-algebra. In particular, $(M \times N, \tilde{\cdot})$ is a semigroup.

Henceforth we work with the assumption that M and N are finite. Our aim is to show that the algebra operators of $\mathbf{M} \ltimes \mathbf{N}$ can be effectively computed from the algebra operators of $\mathbf{M}$ and $\mathbf{N}$.

We first observe that the action requirements can be equivalently stated in terms of algebra operators as follows:

C'-1 $1 * n = n$
C'-2 $(m_1 \cdot m_2) * n = m_1 * (m_2 * n)$
C'-3 $m * (n_1 \mathbin{\hat{\cdot}} n_2) = m * n_1 \mathbin{\hat{\cdot}} m * n_2$
C'-4 $m * n^{\hat{\tau}} = (m * n)^{\hat{\tau}}$
C'-5 $m * n^{\hat{\tau}^*} = (m * n)^{\hat{\tau}^*}$
C'-6 $m * \{n_1, \ldots, n_j\}^{\hat{\kappa}} = \{m * n_1, \ldots, m * n_j\}^{\hat{\kappa}}$
C'-7 $n * 1 = n$
C'-8 $n * (m_1 \cdot m_2) = (n * m_1) * m_2$
C'-9 $(n_1 \mathbin{\hat{\cdot}} n_2) * m = n_1 * m \mathbin{\hat{\cdot}} n_2 * m$
C'-10 $n^{\hat{\tau}} * m = (n * m)^{\hat{\tau}}$
C'-11 $n^{\hat{\tau}^*} * m = (n * m)^{\hat{\tau}^*}$
C'-12 $\{n_1, \ldots, n_j\}^{\hat{\kappa}} * m = \{n_1 * m, \ldots, n_j * m\}^{\hat{\kappa}}$
C'-13 $(m_1 * n) * m_2 = m_1 * (n * m_2)$

The following lemma says that the binary operator $\tilde{\cdot}$ of $\mathbf{M} \ltimes \mathbf{N}$ can be expressed using the binary operators $\cdot$ (of $\mathbf{M}$) and $\hat{\cdot}$ (of $\mathbf{N}$). It follows easily from the definition of the *induced* operator $\tilde{\cdot}$.

Lemma 3. *The operator* $\tilde{\cdot}$ *can be computed as follows:* $(m_1, n_1) \mathbin{\tilde{\cdot}} (m_2, n_2) = (m_1 \cdot m_2, n_1 * m_2 \mathbin{\hat{\cdot}} m_1 * n_2)$.

Recall that, as observed earlier, $(M \times N, \tilde{\cdot})$ is a semigroup. Moreover, it is finite as both M and N are finite. A basic result about finite semigroups says that every element of $M \times N$ admits a power which is an idempotent element (an element whose 'square' is itself). An easy consequence of the previous lemma is that if (m, n) is an idempotent element of $M \times N$ then m is also an idempotent element of M.

Now we focus on the unary operator $\tilde{\tau} : (M \times N) \to (M \times N)$. In view of the second axiom in the definition of a $\oplus$-algebra, it suffices to show that the operator $\tilde{\tau}$ can be computed at idempotent elements of $M \times N$ in terms of the algebra operators of $\mathbf{M}$ and $\mathbf{N}$.

Lemma 4. *Let* (e, n) *be an idempotent element of* $M \times N$. *Then* $(e, n)^{\tilde{\tau}} = (e^{\tau}, n * e^{\tau} \mathbin{\hat{\cdot}} (e * n * e^{\tau})^{\hat{\tau}})$.

Proof: By definition of the induced operator $\tilde{\tau}$, $(e, n)^{\tilde{\tau}} = \tilde{\pi}(u)$ where $u = (e, n)^{\omega}$ is the ω-word over the domain $(\mathbb{N}, <)$ such that every position is mapped to (e, n). We first compute $\theta(u) = (v, w)$ according to the Definition 1. It is easy to see that $v = e^{\omega}$ and w is the ω-word whose first position is mapped to $n * e^{\tau}$ and all other positions are mapped to $e * n * e^{\tau}$. As a result, $\pi(v) = e^{\tau}$ and $\hat{\pi}(w) = (n * e^{\tau}) \mathbin{\hat{\cdot}} ((e * n * e^{\tau})^{\hat{\tau}})$. The proof now follows by observing that $\tilde{\pi}(u) = (\pi(v), \hat{\pi}(w))$. ∎

An analogous result for the unary operator $\tilde{\tau}^* : (M \times N) \to (M \times N)$ is recorded in the next lemma. Its proof is omitted.

Lemma 5. *Let* (e, n) *be an idempotent element of* $M \times N$. *Then* $(e, n)^{\tilde{\tau}^*} = (e^{\tau^*}, (e^{\tau^*} * n * e)^{\hat{\tau}^*} \mathbin{\hat{\cdot}} e^{\tau^*} * n)$.

Finally, the next lemma shows that the operator $\tilde{\kappa}$ of $\mathbf{M} \ltimes \mathbf{N}$ can be computed using the algebra operators of $\mathbf{M}$ and $\mathbf{N}$. We skip its proof.

Lemma 6. *The operator* $\tilde{\kappa}$ *can be computed as follows:*

$\{(m_1, n_1), \ldots, (m_p, n_p)\}^{\tilde{\kappa}} = (m, \{m * n_1 * m, \ldots, m * n_p * m\}^{\hat{\kappa}})$

where $m = \{m_1, \ldots, m_p\}^{\kappa}$.

We now present an example of a semidirect product construction and highlight some important elements in the $\circledast$-algebra. Recall that an element e of M is called an *idempotent* if $e \cdot e = e$. We call it an *ω-idempotent* (resp. *ω^*-idempotent* and *η-idempotent*) if $e^{\tau} = e$ (resp. $e^{\tau^*} = e$ and $e^{\kappa} = e$).

An element r of M is called a *right zero* if $m \cdot r = r$ for every element m of M. The notion of a left zero is defined similarly. An element z is called a *zero*, if it is both a left zero and a right zero. If a zero exists, it is unique.

Example 3. *Consider* $M = \mathrm{U}_1$ *acting on* $N = \mathrm{U}_1$ *with a trivial left action and a non-trivial monoidal right action where everything in* N *maps to* 1. *The* $\circledast$*-algebra* $\mathrm{U}_1 \ltimes \mathrm{U}_1$ *is given below. We write the element* (i, j) *as* ij *in this example.*

	11	10	00	01	τ	τ^*
11	11	10	00	01	11	11
10	10	10	00	01	10	10
00	00	00	00	01	01	00
01	01	00	00	01	01	01

Fig. 2. Note that all elements are ω^* idempotents, and the elements $00, 01$ are right zero. We have $\{11\}^{\kappa}$, $\{10\}^{\kappa} = \{11, 10\}^{\kappa} = 10$ and $X^{\kappa} = 01$ if X contains any right zero.

The $\circledast$-algebra in the above example is crucially used in Section V. It is called $\mathbf{M}^{r}$ there.

		$\overset{x}{\downarrow}$	$\rightsquigarrow$ (evaluation)
	$u:$	$\ldots\boxed{a}\ldots$	
$h:\Sigma\to M\Box N$	$h(u):$	$\ldots\boxed{(m_a,f_a)}\ldots$	(m,f)
$h_1:\Sigma\to M$	$h_1(u):$	$\ldots\boxed{m_a}\ldots$	m
$\sigma:\Sigma^{\oplus}\to(M\times\Sigma\times M)^{\oplus}$	$u'=\sigma(u):$	$\ldots\boxed{h_1(u_{<x}),a,h_1(u_{>x})}\ldots$	
$h_2:(M\times\Sigma\times M)\to N$	$h_2(u'):$	$\ldots\boxed{f_a(h_1(u_{<x}),h_1(u_{>x}))}\ldots$	$f(1,1)$

Fig. 3. The block product operational view

C. Block Product Construction

Let $\mathbf{M}=(M,\cdot,\tau,\tau^*,\kappa)$ and $\mathbf{N}=(N,\hat{\cdot},\hat{\tau},\hat{\tau}^*,\hat{\kappa})$ be $\oplus$-algebras. The corresponding $\oplus$-semigroups (M,π) and $(N,\hat{\pi})$ will also be denoted by $\mathbf{M}$ and $\mathbf{N}$. The set $N^{M\times M}$ of all functions from $M\times M$ into N also forms a $\oplus$-algebra ($\oplus$-semigroup) under the componentwise product. This $\oplus$-algebra (denoted by $\mathbf{K}$ with underlying set $K=N^{M\times M}$) can be simply viewed as the direct product of $M\times M$ copies of $\mathbf{N}$. The notations $\hat{\cdot},\hat{\tau},\hat{\tau}^*,\hat{\kappa}$ of $\mathbf{N}$ will also be used to denote the corresponding operators on $\mathbf{K}$.

The block product of $\mathbf{M}$ and $\mathbf{N}$ is denoted by $\mathbf{M}\Box\mathbf{N}$ and is the semidirect product of $\mathbf{M}$ and $\mathbf{K}$ (with underlying set $M\ltimes K$) with respect to the *canonical* 'actions': for $m\in M$ and $f\in N^{M\times M}$,

$$(m*f)(m_1,m_2)=f(m_1m,m_2)$$

$$(f*m)(m_1,m_2)=f(m_1,mm_2)$$

Lemma 7. *The canonical actions satisfy the axioms of actions.*

D. Block Product Principle

In this subsection, we state and prove the block product principle. Roughly speaking the block product principle allows to express the formal languages recognized by a block product $\mathbf{M}\Box\mathbf{N}$ in terms of languages recognized by $\mathbf{M}$ and $\mathbf{N}$.

Fix a finite alphabet Σ. As $\Sigma^{\oplus}$ is a free $\oplus$-semigroup, a morphism from $\Sigma^{\oplus}$ to $\mathbf{M}\Box\mathbf{N}=\mathbf{M}\ltimes\mathbf{K}$ is simply given (determined) by a map $h:\Sigma\to M\ltimes K$. Its pointwise extension will be denoted by $\bar{h}:\Sigma^{\oplus}\to(M\ltimes K)^{\oplus}$. Sometimes, it will be convenient to denote this extension also by h. Further, composing this with the countable product $\tilde{\pi}$ results into a *morphism* which, to a word $u\in\Sigma^{\oplus}$, associates the element $\tilde{\pi}(\bar{h}(u))\in M\ltimes K$. This morphism may also be denoted by h (that is, $h(u)$ may simply equal $\tilde{\pi}(\bar{h}(u))$). The context will make it clear as to which interpretation of 'h' applies. These slight abuses of notations are used several times in what follows in order to keep the notation simple and improve readability.

Let $h:\Sigma\to M\ltimes K$ be a morphism and let $(m_a,f_a)=h(a)$ for each $a\in\Sigma$. We define the map/morphism $h_1:\Sigma\to M$ by letting $h_1(a)=m_a$ for each letter a. Next we define the *state-based* transducer $\sigma:\Sigma^{\oplus}\to(M\times\Sigma\times M)^{\oplus}$ as follows: let $u\in\Sigma^{\oplus}$ with domain α. The word $u'=\sigma(u)$ has domain α and is defined over the alphabet $M\times\Sigma\times M$. For $x\in\alpha$,

$$u'[x]=(h_1(u_{<x}),u[x],h_1(u_{>x}))$$

Next, we define the map/morphism $h_2:(M\times\Sigma\times M)\to N$ as: for $(m_1,a,m_2)\in(M\times\Sigma\times M)$,

$$h_2((m_1,a,m_2))=f_a(m_1,m_2)$$

Going ahead, given a word $u'\in(M\times\Sigma\times M)^{\oplus}$ and $m_1,m_2\in M$, we define $m_1u'm_2$ to be the word (with the same domain as u') such that for a position x with $u'[x]=(m_1',a,m_2')$, $(m_1u'm_2)[x]=(m_1m_1',a,m_2'm_2)$.

Now we are ready to state a key technical lemma which will help us establish the block product principle.

Lemma 8. *For $u\in\Sigma^{\oplus}$ and $m_1,m_2\in M$, $h(u)=(m,f)$ iff $h_1(u)=m$ and $h_2(m_1\sigma(u)m_2)=f(m_1,m_2)$.*

Proof: Fix $u\in\Sigma^{\oplus}$ and $u'=\sigma(u)$. Let $h(u)\in(M\ltimes K)^{\oplus}$ be the image of the pointwise extension of h applied to u. The words $h_1(u)\in M^{\oplus}$ and $h_2(u')\in N^{\oplus}$ are defined similarly. Observe that, for a position x of u, with $u[x]=a$ and $h(a)=(m_a,f_a)$, $h(u)[x]=(m_a,f_a)$, $h_1(u)[x]=m_a$, $u'[x]=(h_1(u_{<x}),a,h_1(u_{>x}))$ and $h_2(u')[x]=f_a(h_1(u_{<x}),h_1(u_{>x}))$. See figure 3.

Consider the map $\theta:(M\ltimes K)^{\oplus}\to M^{\oplus}\times K^{\oplus}$ from Lemma 1 (with K playing the role of N in the statement). Let $\theta(h(u))=(v,w)$. Observe that $v\in M^{\oplus}$ and $w\in K^{\oplus}$. It is straightforward to check that $v=h_1(u)$. Further, by the definition of θ, for a position x of u, with $h(u)[x]=(m_a,f_a)$, $w[x]=h_1(u_{<x})*f_a*h_1(u_{>x})$.

	$\overset{x}{\downarrow}$
	$\ldots\boxed{(m_a,f_a)}\ldots$
$v=h_1(u):$	$\ldots\boxed{m_a}\ldots$
$w:$	$\ldots\boxed{h_1(w_{<x})*f_a*h_1(w_{>x})}\ldots$

Fig. 4. $\theta:(M\ltimes K)^{\circledast}\to M^{\circledast}\times K^{\circledast}$ and $\theta(u)=(v,w)$

Now we relate the word $w \in K^{\oplus}$ with $u' \in (M \times \Sigma \times M)^{\oplus}$. Towards this, consider the projection morphisms: for $m_1, m_2 \in M$, $\Pi_{m_1,m_2} : K \to N$ defined as $\Pi_{m_1,m_2}(g) = g(m_1, m_2)$. As expected, the pointwise extensions of Π_{m_1,m_2} are also denoted by Π_{m_1,m_2}.

For further analysis, fix a choice of $m_1, m_2 \in M$. Let x be a position with $u[x] = a$ and $h(a) = (m_a, f_a)$. As observed earlier $u'[x] = (h_1(u_{<x}), a, h_1(u_{>x})) \in M \times \Sigma \times M$ and $w[x] = h_1(u_{<x}) * f_a * h_1(u_{>x}) \in K$. Clearly, $m_1 u'[x] m_2 = (m_1 h_1(u_{<x}), a, h_1(u_{>x}) m_2)$.

We proceed further with some simple calculations.

$$\begin{aligned}\Pi_{m_1,m_2}(w[x]) &= (h_1(u_{<x}) * f_a * h_1(u_{>x}))\,(m_1, m_2) \\ &= f_a(m_1 h_1(u_{<x}), h_1(u_{>x}) m_2)\end{aligned}$$

$$\begin{aligned}h_2(m_1 u'[x] m_2) &= h_2\left((m_1 h_1(u_{<x}), a, h_1(u_{>x}) m_2)\right) \\ &= f_a(m_1 h_1(u_{<x}), h_1(u_{>x}) m_2)\end{aligned}$$

This reveals that for each position x, $\Pi_{m_1,m_2}(w[x]) = h_2(m_1 u'[x] m_2)$. Thanks to the fact that both $\Pi_{m_1,m_2}(w)$ and $m_1 u' m_2$ are defined pointwise, we have $\Pi_{m_1,m_2}(w) = h_2(m_1 u' m_2)$. We let f denote the evaluation of w in K and exploit the fact that both Π_{m_1,m_2} and h_2 are *morphisms* to conclude that, for $m_1, m_2 \in M$, $f(m_1, m_2) = h_2(m_1 u' m_2) \in N$.

With $h_1(u) = m$, the proof of the proposition is now immediate by Definition 2 which asserts that $h(u) = (m, f)$. ∎

We now use this lemma to derive the following result often referred to as *the block product principle* (see [13], [18] for the related wreath product principle).

Proposition 1 (Block Product Principle). *Let $L \subseteq \Sigma^{\oplus}$ be recognized by $h : \Sigma \to \mathbf{M} \Box \mathbf{N}$ via a subset $F \subseteq M \times K$. Then L can be expressed as a finite union of languages of the form $L_1 \cap (\bigcap_{i,j} \sigma^{-1}(L_{ij}))$ where L_1 and L_{ij} are recognized by $\mathbf{M}$ and $\mathbf{N}$ respectively, for $1 \le i, j \le |M|$.*

Proof: Consider an element $(m, f) \in M \times K$. By Lemma 8, for $u \in \Sigma^{\oplus}$, $h(u) = (m, f)$ iff $h_1(u) = m$ and $h_2(m_1 \sigma(u) m_2) = f(m_1, m_2)$ for all $m_1, m_2 \in M$.

Next, for $1 \le i, j \le |M|$, we define the maps/morphisms $h_{ij} : (M \times \Sigma \times M) \to N$ as follows: $h_{ij}((m_1, a, m_2)) = h_2((m_i m_1, a, m_2 m_j))$. It is easy to see that, for any word $u' \in (M \times \Sigma \times M)^{\oplus}$, $h_{ij}(u') = h_2(m_i u' m_j)$.

As a consequence, we get

$$L = \bigcup_{(m,f) \in F} \left({h_1}^{-1}(m) \cap \left(\bigcap_{i,j} \sigma^{-1}({h_{ij}}^{-1}(f(m_i, m_j))) \right) \right)$$

This completes the proof. ∎

IV. First order logic

A. Definitions

First-order logic (FO[<]) over a finite alphabet Σ is a logic which can be inductively built using the following operations.

$$a(x) \mid x < y \mid x = y \mid \phi \vee \phi \mid \neg \phi \mid \exists x\ \phi$$

Here $a \in \Sigma$ and ϕ is any FO[<] formula. We use the letters ϕ, ψ, φ (with or without subscripts) to represent FO[<] formulas, and the letters x, y, z (with or without subscripts) to represent FO[<] variables.

A variable is free if it is not quantified in the formula. The set of free variables in a formula φ is denoted by $\mathsf{free}(\varphi)$. A formula with no free variables is called a sentence. The language of a sentence φ (denoted by $\mathrm{L}(\varphi)$) is the set of all words $u \in \Sigma^{\oplus}$ that satisfies φ.

Let us look at some examples of countable languages definable in FO[<] and its two variable fragment FO^2[<]. Over finite words, FO^2[<] can talk about occurrence of letters and also about the order in which they appear [24]. Over countable linear orderings, it can also say that there is no maximum position. For example, the following formula states that every position is labelled by a and there is no maximum position.

$$\big(\forall x\ \exists y\ x < y\big) \wedge \big(\forall x\ a(x)\big)$$

Analogously, FO^2[<] can also talk about left-open words. However, the two variable fragment is not as expressive as full first order. FO^2[<] satisfies a downward property (similar to Löwenheim-Skolem downward theorem for first order logic): a satisfiable FO^2[<] formula has a scattered satisfying model (see [10]). Therefore, the following language, which says the linear ordering is dense and has at least two distinct positions, is not definable in FO^2[<].

$$\exists x, y\ x < y \wedge \forall x, y\ \ (x < y) \Rightarrow \big(\exists z\ x < z < y\big)$$

Example 4. *Consider the language $L \subseteq \{a, b\}^{\circledast}$ of all words w which satisfy the property: There is a gap in w towards which the letter b approaches from the left and on the right of the gap there is an interval with only a's. This is definable in FO[<] by first guessing two points x_ℓ and x_r on both sides of the gap. Let $\psi(x_\ell, x_r, y)$ be a formula which is true if and only if y is between x_ℓ and x_r and is after all occurrences of b's between x_ℓ and x_r. The three formulas below say that (1) the b's form an omega sequence, (2) the a's after b form an omega* sequence, and (3) there is factor after the gap which contains only a's.*

1) $\phi(x_\ell, x_r) ::= \forall y \in (x_\ell, x_r)\ \big(b(y) \Rightarrow (\exists z > y\ b(z))\big)$
2) $\chi(x_\ell, x_r) ::= \forall y \big(\psi(x_\ell, x_r, y) \Rightarrow \exists z < y\ \psi(x_\ell, x_r, z)\big)$
3) $\varphi(x_\ell, x_r) ::= \forall x, y\ \big(\psi(x_\ell, x_r, x) \wedge \psi(x_\ell, x_r, y)\big) \Rightarrow \big(\forall z \in (x, y)\ \psi(x_\ell, x_r, z)\big)$

The formula $\exists x_\ell, x_r\ (\phi \wedge \chi \wedge \varphi)$ defines the language L.

In the following subsections, we provide block product characterizations of $\oplus$-algebras recognizing FO[<] and FO^2[<] languages over linear countable orderings. We also provide a characterization of FO[<] in terms of regular expressions.

B. Iterated block product

Block product of $\oplus$-algebras is not associative and so the order of product (equivalently nesting of brackets) varies the resulting structure for a given list of $\oplus$-algebra. For example, for three $\oplus$-algebras, there are only two distinct nesting possible and the following lemma shows that one of them is at least as powerful as the other.

Lemma 9. *For any three $\oplus$-algebras M, N and P,*

$$M\Box(N\Box P) \preceq (M\Box N)\Box P$$

For multiple $\oplus$-algebras, there can be several ways of bracketing. One particular nesting is known as the strongly iterated block product, also referred to simply as the iterated block product. For any set R of $\oplus$-algebras, it is defined recursively as follows:

1) N is an iterated block product for any $N \in R$
2) If M is an iterated product, then $M\Box N$ is an iterated product for any $N \in R$

For example, in Lemma 9 the RHS is an iterated block product over $\{M, N, P\}$. The set of all iterated block products of a set R is denoted by $\Box^* R$. For a singleton set, we drop the set notation.

For ease of notation, we refer to a $\oplus$-algebra by simply *algebra*.

Theorem 2. $\mathrm{L}(\mathrm{FO}[<]) = \mathrm{L}(\Box^*\mathrm{U}_1)$

Proof: First we show left to right inclusion, which goes via structural induction on the formulas of first order logic. We know that FO[<] has letter and order predicates, is closed under boolean operations and existential quantification. Inductively we prove that for any FO formula $\varphi = \phi(x_1, x_2, \ldots, x_n)$, the language $\mathrm{L}(\varphi) \subseteq (\Sigma \times \{0,1\}^n)^\oplus$ is recognized by an iterated block product of U_1.

It is easy to show that both the languages definable by $\varphi = a(x)$ and $\varphi = x < y$ can be recognized by $\mathrm{U}_1\Box\mathrm{U}_1$. Similarly, boolean combinations of first order formulas can also be recognized by cartesian products of the corresponding algebras (clearly, cartesian products divides block products). The interesting case is when $\varphi = \exists x\phi$. Let $\mathrm{L}(\phi) \subseteq (\Sigma \times \{0,1\})^\oplus$ be recognized by an algebra $M \in \Box^*\mathrm{U}_1$ via the morphism $h' : (\Sigma \times \{0,1\}) \to M$. That is, $\exists F' \subseteq M, h'^{-1}(F') = \mathrm{L}(\phi)$. Consider the morphism $h : \Sigma \to M\Box\mathrm{U}_1$, where $h(a) = (h'(a,0), f_a)$ and

$$f_a(m_1, m_2) = \begin{cases} 0, & \text{if } m_1.h'(a,1).m_2 \in F' \\ 1, & \text{otherwise} \end{cases}$$

Now we use the induced morphism $h_1 : \Sigma \to M$ (which maps a to $h'(a,0)$) and the associated state-based transducer $\sigma : \Sigma^\oplus \to (M \times \Sigma \times M)^\oplus$. Further the other (see the notation before Lemma 8) induced morphism $h_2 : (M \times \Sigma \times M) \to \mathrm{U}_1$ maps (m_1, a, m_2) to, 0 if $m_1.h'(a,1).m_2 \in F'$ and to 1 otherwise. Note that h_2 maps a word to 0 iff the word has a position which is labelled by a letter which maps to 0.

We claim that $u \in \mathrm{L}(\varphi)$ if and only if $h_2(\sigma(u)) = 0$. First we prove the forward direction. Let $u \in \mathrm{L}(\varphi)$. Then $u = u_1 a u_2$ such that $u_1^0(a,1)u_2^0 \models \phi$ (for a word $v \in \Sigma^\oplus$, we denote by v^0 the word over the same domain as v such that $v^0[i] = (v[i], 0)$). Then $h'(u_1^0(a,1)u_2^0) \in F'$ and hence $h_2(\sigma(u)) = 0$. For the other direction, let us assume $h_2(\sigma(u)) = 0$. Therefore, $\sigma(u)$ has a position which is labelled by (m_1, a, m_2) such that $m_1.h'(a,1)m_2 \in F'$. But this means u can be factored as $u_1 a u_2$ such that $m_1 = h_1(u_1)$ and $m_2 = h_1(u_2)$. Therefore $u_1^0(a,1)u_2^0 \in \mathrm{L}(\phi)$ and hence $u \in \mathrm{L}(\varphi)$.

Now with a suitable choice of F, it can be seen that $h^{-1}(F) = \mathrm{L}(\varphi)$. This completes the proof of the left to right inclusion.

The right to left inclusion is via induction on the number of iterated blocks of U_1. It is easy to see that all languages recognized by exactly one U_1 can be defined in first order logic. This takes care of the base case. Let the hypothesis hold for algebra M. We show that, a language L recognized by some morphism $h : \Sigma \to M\Box\mathrm{U}_1$ can be defined in FO[<]. As we have seen in subsection III-D, h induces a natural morphism $h_1 : \Sigma \to M$. Let $\sigma : \Sigma^\oplus \to (M \times \Sigma \times M)^\oplus$ be the state-based transducer induced by h_1. From the block product principle, L can be expressed as a finite boolean combination of languages of the form L_1 and $\sigma^{-1}(L_2)$ where L_1 and L_2 are recognized by M and U_1 respectively. By the induction hypothesis both L_1 and L_2 are FO[<] definable. So it suffices to show that for an FO[<] language L_2 over the alphabet $(M \times \Sigma \times M)$ the language $\sigma^{-1}(L_2)$ is also FO[<] definable. We prove by structural induction that for every FO[<] formula φ over alphabet $M \times \Sigma \times M$, there exists a FO[<] formula ψ over alphabet Σ such that $\mathsf{free}(\varphi) = \mathsf{free}(\psi)$ and for all words $w \in \Sigma^\oplus$, and for all assignments $E : \mathsf{free}(\varphi) \to \mathsf{dom}(w)$, $w, E \models \psi$ if and only if $\sigma(w), E \models \varphi$. The base case when φ is a letter-predicate is the most interesting. Let $\varphi = (m, a, n)(x)$. By induction hypothesis there are sentences, for each $m \in M$, φ_m such that $h_1(w) = m$ if and only if $w \models \varphi_m$ We define $\psi = a(x) \wedge \varphi_m|_{<x} \wedge \varphi_n|_{>x}$, where $\phi|_{<x}$ (resp. $\phi|_{>x}$) denotes the relativisation of the variables in formula ϕ to positions less than (resp. greater than) x. Note that $\sigma(w), i \models \varphi$ if and only if $w, i \models \psi$. Inductively applying this translation for other formulas gives the required formula ψ. ■

C. Weakly iterated block product

Now we introduce the notion of weakly iterated block products. For any set R of algebras, it is defined recursively as follows:

1) N is a weakly iterated block product for any $N \in R$
2) If M is an weakly iterated block product, then $N\Box M$ is a weakly iterated block product for any $N \in R$

For example, in Lemma 9 the LHS is a weakly iterated product over $\{M, N, P\}$.The set of all weakly iterated block products of a set R is denoted by $\Box^*_{\mathsf{w}} R$. For a singleton set, we drop the set notation for convenience.

The following lemma or rather its generalization will be used later.

Lemma 10. *For any $\oplus$-algebras M, N and P,*

$$(M \times N)\Box P \preceq M\Box(N\Box P)$$

This can be generalized to $(M_1 \times \ldots \times M_k)\Box N$ as expected.

Theorem 3. $\mathrm{L}(\mathrm{FO}^2[<]) = \mathrm{L}(\Box^*_{\mathsf{w}}\mathrm{U}_1)$

Proof: The right to left inclusion is via induction on the number of blocks of U_1s. First, observe that languages recognized by a single U_1 can be defined in $\mathrm{FO}^2[<]$. For the induction step, we follow the proof in Theorem 2 closely. Let the hypothesis hold for algebra $M \in \Box_{\mathsf{w}}^* \mathrm{U}_1$. We show that a language L recognized by some morphism $h : \Sigma \to \mathrm{U}_1 \Box M$ can be defined in $\mathrm{FO}^2[<]$. Let $\sigma : \Sigma^{\oplus} \to (\mathrm{U}_1 \times \Sigma \times \mathrm{U}_1)^{\oplus}$ be the state-based transducer associated with the induced morphism $h_1 : \Sigma \to U_1$. From the block product principle, L can be expressed as a finite boolean combination of languages of the form L_1 and $\sigma^{-1}(L_2)$ where L_1 and L_2 are recognized by U_1 and M respectively. By the induction hypothesis both L_1 and L_2 are $\mathrm{FO}^2[<]$ definable. So it suffices to show that for an $\mathrm{FO}^2[<]$ language L_2 over the alphabet $(\mathrm{U}_1 \times \Sigma \times \mathrm{U}_1)$ the language $\sigma^{-1}(L_2)$ is also $\mathrm{FO}^2[<]$ definable. As observed in Theorem 2 the base case is the non-trivial case. The following formula accepts $\sigma^{-1}(L_2)$ if L_2 is defined by the formula $(0, a, 1)(x)$.

$$a(x) \wedge \big(\exists y < x \bigvee_{h_1(b)=0} b(y)\big) \wedge \big(\forall y > x \bigvee_{h_1(c)=1} c(y)\big)$$

Note that we used only two variables for the above translation. The other base cases are similar. We apply this translation inductively for other formulas.

Now we show the other inclusion of the proof. $\mathrm{FO}^2[<]$ has a "normal form" [19] where the quantifier at the maximum depth along with its scope is of the form $\exists x(a(x) \wedge x < y)$ or $\exists x\ (a(x) \wedge x > y)$. Let us call these base formulas. Note that base formulas have a free variable y. $\mathrm{FO}^2[<]$ sentences can be inductively built by replacing letter-predicates $c(y)$ in a formula by one of the base formulas. Let us assume that languages definable by $\mathrm{FO}^2[<]$ formulas of quantifier depth k can be recognized by weak block product of U_1s. Consider a formula α over the alphabet Σ of quantifier depth $k+1$. From our observation, there exists a set $\Gamma = \{\gamma_1, \ldots, \gamma_l\}$ and a formula ϕ over $\mathcal{P}(\Gamma)$ such that replacing every occurrence of $\gamma_i(y)$ by a base formula $\psi_i(y)$ over the alphabet Σ gives you the formula α. Here ϕ is a formula of quantifier depth k. By our assumption, we know that there is an algebra M and a morphism $h : \mathcal{P}(\Gamma) \to M$ which recognizes $\mathrm{L}(\phi)$. We now show how to get, for any word $w \in \Sigma^{\oplus}$, the 'corresponding' word over $\mathcal{P}(\Gamma)$ which keeps track of the truth values the formulas in $\{\psi_1(y), \ldots, \psi_l(y)\}$ at *every* position in w.

Consider $\mathcal{P}(\Sigma)$ as a $\circledast$-monoid where the binary product $\cdot$ and shuffle operator κ are unions of sets, and the omega-operator τ and omega*-operator τ^* are identity maps. So every element is a shuffle idempotent. Notice that $\mathcal{P}(\Sigma)$ is essentially the cartesian product of $|\Sigma|$-many U_1s. Now there exists a morphism $g : \Sigma^{\oplus} \to \mathcal{P}(\Sigma)$ such that $g(w) = \{a \mid \text{the letter } a \text{ occurs in } w\}$. The transducer associated with g is $\sigma : \Sigma^{\oplus} \to (\mathcal{P}(\Sigma) \times \Sigma \times \mathcal{P}(\Sigma))^{\oplus}$ where, for a word w, $\sigma(w)[y] = (g(w_{<y}), w[y], g(w_{>y}))$ for every position y in w. Observe that the word $\sigma(w)$ carries, at every position y, the information about the set of letters which occur to the left (as well as right) of y in w. Clearly, this information is enough to decide the truth values of $\{\psi_1(y), \ldots, \psi_l(y)\}$ at position y.

So, we construct the map $f : (\mathcal{P}(\Sigma)) \times \Sigma \times \mathcal{P}(\Sigma))) \to \mathcal{P}(\Gamma)$ such that, for every word $w \in \Sigma^{\oplus}$, $f(\sigma(w))$ has the 'correct' information. Now the morphism $h : \mathcal{P}(\Gamma) \to M$ can make use of this information to further decide the truth of ϕ. Thus $\mathcal{P}(\Sigma) \Box M$ can recognize the language of the $\mathrm{FO}^2[<]$ sentence α, and by Lemma 10 this algebra is in $\Box_{\mathsf{w}}^* \mathrm{U}_1$. ∎

D. Schützenberger-McNaughton-Papert Theorem

The marked star-free regular expressions over Σ are defined by the following grammar

$$r = \emptyset \mid a \mid \neg r \mid r_1 + r_2 \mid r_1 \cap r_2 \mid r_1 a r_2$$

Each such expression r naturally corresponds to a language $L(r) \subseteq \Sigma^{\oplus}$ in a straightforward manner: $L(\emptyset) = \emptyset, L(a) = \{a\}, L(\neg r) = \Sigma^{\oplus} \setminus L(r), L(r_1 + r_2) = L(r_1) \cup L(r_2)$ and $L(r_1 a r_2) = L(r_1) \cdot a \cdot L(r_2)$ etc. Note that we are simply extending the marked star-free regular expressions defined in [2] from *scattered* words to countable words. A language L is said to be marked star-free iff there exists a marked star-free regular expression r such that $L = L(r)$.

Theorem 4. *A language $L \subseteq \Sigma^{\oplus}$ is marked star-free iff L is* $\mathrm{FO}[<]$-*definable.*

Proof: The left to right direction is by structural induction: an expression r is translated into a $\mathrm{FO}[<]$-sentence ϕ_r such that $L(r) = \{w \in \Sigma^{\oplus} \mid w \models \phi_r\}$. The only non-trivial case being $\phi_{r_1 a r_2} = \exists x \phi_{r_1}|_{<x} \wedge a(x) \wedge \phi_{r_2}|_{>x}$. We skip the details.

Towards the right to left direction, in view of Theorem 2, it suffices to show that languages recognized by iterated block-products of U_1 admit marked star-free regular expressions. We proceed by induction on the number of iterated blocks of U_1 in the recognizing $\circledast$-algebra. The base case is easy.

Let $h : \Sigma^{\oplus} \to M \Box \mathrm{U}_1$ recognize a language L. By the block product principle (see Proposition 1), L is a boolean combination of languages $L' = h_1^{-1}(m)$ and $L'' = \sigma^{-1}(h_2^{-1}(u))$, for some $m \in M, u \in \mathrm{U}_1$. Since marked star-free expressions are closed under boolean operations, it suffices to show that L' and L'' have marked star-free expressions.

For $m' \in M$, we denote by $L_{m'}$, the language $h_1^{-1}(m')$, recognized by M and hence, by induction, has a marked star-free expression. As $L' = L_m$, L_m has a marked star-free expression over Σ. Consider $u = 0$, then it is easy to see that $\sigma^{-1}(h_2^{-1}(u)) = \bigcup_{(m_1, a, m_2) \in h_2^{-1}(0)} L_{m_1} \cdot a \cdot L_{m_2}$. The above 'decomposition' of $\sigma^{-1}(h_2^{-1}(u = 0))$ makes it evident that it too admits a marked star-free expression over Σ. The expression for the case with $u = 1$ is simply the complement expression. Thus L'' too has a marked star-free expression. ∎

Now we are ready to state the analogue of Schützenberger-McNaughton-Papert theorem for countable words.

Theorem 5. *[Schützenberger-McNaughton-Papert Theorem] Let $L \subseteq \Sigma^{\oplus}$. Then the following are equivalent.*

1) L is $\mathrm{FO}[<]$-definable.
2) L admits a marked star-free regular expression.
3) L is recognized by block products of U_1.

4) The syntactic algebra of L satisfies the equations[1]:
- $e^2 = e \Rightarrow e^{\tau} \cdot e^{\tau^*} = e$.
- $e^{\tau} = e^{\tau^*} = e \Rightarrow \{e\}^{\kappa} = e$.
- $\{e\}^{\kappa} = e \wedge eae = e \Rightarrow \{e, a\}^{\kappa} = e$.

Proof: The equivalence of the first three conditions follows from Theorems 2 and 4. The equivalence of the first and the last condition was shown in [6]. ∎

Over finite words, the above theorem holds with the last condition replaced by: 'the syntactic monoid of L is aperiodic'.

Over countable scattered words, it was shown in [2] that FO and marked star-free regular expressions have the same expressive power. It also gives an algebraic characterization (similar in spirit to the last condition in Theorem 5 invoked for $\Diamond$-algebras). They also asked if the 'scattered' hypothesis can be removed. Theorem 5 resolves this question satisfactorily.

Now we point out an interesting consequence of Theorem 5. Observe that both conditions 3 and 4 are algebraic in nature. The implication $3 \Rightarrow 4$ is rather easy to prove. The implication $4 \Rightarrow 3$ may be viewed as a suitable language-theoretic version of a special case of the Krohn-Rhodes theorem. As mentioned in the introduction, a special case of Krohn-Rhodes theorem asserts that an aperiodic monoids divides a block product of U_1. Roughly speaking, $4 \Rightarrow 3$ asserts that a $\circledast$-monoid satisfying the equations in 4 can be simulated by a block products of U_1!

V. Linear temporal logic

In this section we look at the subclass LTL[S,U] which is closed under boolean operations but only the strict until and strict since operators are allowed. It is known that LTL[S,U] is expressively less powerful than FO[<]. The language in Example 4 is not definable in LTL[S,U]. See [9] for more details.

We first begin with the notion of a marked word which will be useful later. A marked word $\hat{w}$ is a word in $\Sigma^{\circledast}.(\Sigma, \#).\Sigma^{\circledast}$. We denote by $\Sigma^{\circledast}_{\#}$ the set of all marked words over Σ. We will alternatively denote a marked word $\hat{w} \in \Sigma^{\circledast}_{\#}$ by (w, i) if $w \in \Sigma^{\circledast}$, $i \in \mathsf{dom}(w)$ and $\hat{w} = w_{<i}(w[i], \#)w_{>i}$.

LTL[S,U] formula over the alphabet Σ is defined recursively as

$$p \in \Sigma \mid \neg\alpha \mid \alpha \vee \beta \mid \alpha \ \mathbf{S} \ \beta \mid \alpha \ \mathbf{U} \ \beta$$

LTL[S,U] formulas are interpreted on marked words over Σ. Given a formula α and a marked word (w, i), we say that (w, i) satisfies α (denoted by $(w, i) \models \alpha$) if α is true at position i in the word w. We denote by $(w, i) \not\models \alpha$ if (w, i) does not satisfy α. The semantics is defined by structural induction as follows

$$(w, i) \models p \text{ if } w[i] = p$$
$$(w, i) \models \neg\alpha \text{ if } w[i] \not\models \alpha$$
$$(w, i) \models \alpha \vee \beta \text{ if } w[i] \models \alpha \text{ or } w[i] \models \beta$$

We say that $(w, i) \models \alpha \ \mathbf{S} \ \beta$ if

$$\exists j < i \ (w, j) \models \beta \text{ and } \forall k \in (j, i) \ (w, k) \models \alpha$$

[1]We refer the reader to [6] for details

Similarly $(w, i) \models \alpha \ \mathbf{U} \ \beta$ if

$$\exists j > i \ (w, j) \models \beta \text{ and } \forall k \in (i, j) \ (w, k) \models \alpha$$

The derived temporal operators (In the future) $\mathbf{F}\alpha$, (Globally in the future) $\mathbf{G}\alpha$, (In the past) $\mathbf{P}\alpha$, and (Historically or Globally in the past) $\mathbf{H}\alpha$ stand for $\top \ \mathbf{U} \ \alpha$, $\neg(\mathbf{F} \ \neg\alpha)$, $\top \ \mathbf{S} \ \alpha$ and $\neg(\mathbf{P} \ \neg\alpha)$ respectively. We introduce two new operators: $\alpha \ \dot{\mathbf{S}} \ \beta$ and $\alpha \ \dot{\mathbf{U}} \ \beta$. The semantics follows: $(w, i) \models \alpha \ \dot{\mathbf{S}} \ \beta$ if

$$\exists j > i \ (w, j) \models \beta \text{ and } \forall k > j \ (w, k) \models \alpha$$

Similarly, we say that $(w, i) \models \alpha \ \dot{\mathbf{U}} \ \beta$ if

$$\exists j < i \ (w, j) \models \beta \text{ and } \forall k < j \ (w, k) \models \alpha$$

Note that both the above operators are definable using the until and since operators. Henceforth we will denote by LTL[S,U] to mean the logic closed under boolean operations and the four temporal operators we defined above: $\{\mathbf{S}, \mathbf{U}, \dot{\mathbf{S}}, \dot{\mathbf{U}}\}$. Note that both the logics are expressively equivalent but this new logic helps characterizing LTL[S,U] algebraically. Let $A, B \subseteq \Sigma$. Then, we use the following shorthand notation: A stands for the formula $\bigvee_{a \in A} a$. For example, $A \ \mathbf{S} \ B$ denotes the formula $(\bigvee_{a \in A} a) \ \mathbf{S} \ (\bigvee_{b \in B} b)$. We also define the operator depth of a formula to be the maximum number of operators in a path of the parse tree of the formula. It is inductively defined as follows - the operator depth of a formula with no temporal operator is zero, and the operator depth of any formula of type $\alpha \ X \ \beta$, where X is one of the four operators, is one plus the maximum of the operator depth of α and β.

For a formula α, the marked language of α (denoted by $\mathcal{L}_{\#}(\alpha)$) is the set of all marked words (w, i) which satisfy α. For languages defined by formulas, we only consider words with a minimal position. So the language of α (denoted by $\mathcal{L}(\alpha)$) is defined as follows

$$\{w \in \Sigma\Sigma^{\circledast} \mid (w, 0) \models \alpha\}$$

Here 0 denotes the minimal position of the word. For a set of formulas Φ, we denote by $\mathcal{L}(\Phi)$, $\mathcal{L}_{\#}(\Phi)$ the set of all languages, marked languages defined by formulas in Φ.

For a morphism $h : \Sigma^{\circledast} \to M$, we define the function $\hat{h} : \Sigma^{\circledast}_{\#} \to (M \times \Sigma \times M)$ by $\hat{h}(u(a, \#)v) = (h(u), a, h(v))$. We say that a marked language L can be recognized by the morphism h, if there exists a set $S \subseteq (M \times \Sigma \times M)$ such that $L = \hat{h}^{-1}(S)$. For a class of $\circledast$-algebras $\mathbf{M}$, we denote by $\mathcal{L}_{\#}(\mathbf{M})$ the set of all marked languages defined by some morphism from $\Sigma^{\circledast}$ to a $\circledast$-algebra $M \in \mathbf{M}$ and a set $S \subseteq (M \times \Sigma \times M)$.

Consider M^l and M^r shown in Figs. 5 and 6. Our aim is to show that languages recognized by weak block products of M^r and M^l, restricted to words with minimal positions, are exactly those definable in LTL[S,U]. First the base case. The following lemma shows that the marked language of $A \ \mathbf{S} \ B$ is recognizable by M^r.

Lemma 11. *Let $A, B \subseteq \Sigma$. The language $\mathcal{L}_{\#}(A \ \mathbf{S} \ B)$ can be recognized by a morphism, $h : \Sigma^{\circledast} \to \mathrm{M}^r$. Moreover $\mathcal{L}_{\#}(A \ \dot{\mathbf{S}} \ B)$ can be recognized by a morphism, $h : \Sigma^{\circledast} \to \mathrm{M}^r$.*

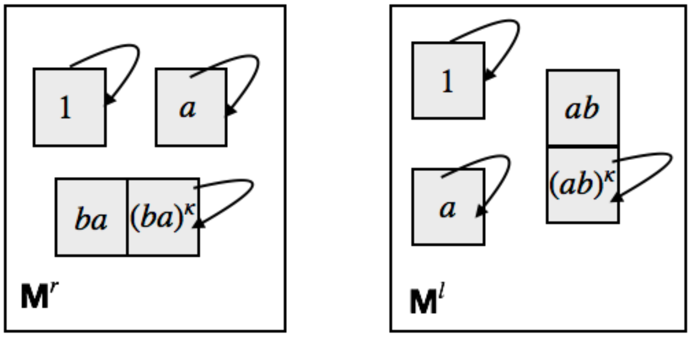

Fig. 5. The algebras M^r and M^l: We have denoted the algebras using the eggbox diagram. Additionally shuffle idempotents are denoted by a self loop. For example in M^l and M^r, the elements $1, a, (ba)^\kappa$ are shuffle idempotents.

	1	a	ba	$(ba)^\kappa$	τ	τ^*
1	1	a	ba	$(ba)^\kappa$	1	1
a	a	a	ba	$(ba)^\kappa$	a	a
ba	ba	ba	ba	$(ba)^\kappa$	$(ba)^\kappa$	ba
$(ba)^\kappa$	$(ba)^\kappa$	ba	ba	$(ba)^\kappa$	$(ba)^\kappa$	$(ba)^\kappa$

Fig. 6. The algebra M^r: Note that all elements are τ^* idempotents and the elements $ba, (ba)^\kappa$ are right zeros. The κ function is defined as follows: $1^\kappa = 1$, $\{1, a\}^\kappa = a$ and $X^\kappa = (ba)^\kappa$ if X contains any right zero.

Similarly $\mathcal{L}_\#(A \mathbf{U} B)$ and $\mathcal{L}_\#(A \dot{\mathbf{U}} B)$ can be recognized by morphisms $h : \Sigma^\circledast \to \mathrm{M}^l$.

Proof: We show that $L = \mathcal{L}_\#(A \mathbf{S} B)$ can be recognized by M^r and leave the rest of the proofs since it is of similar flavour. Note that if $(w, i) \models A \mathbf{S} B$, then $w_{<i} \in \Sigma^\circledast B A^\circledast$. We claim that the morphism, $h : \Sigma^\circledast \to \mathrm{M}^r$ which extends

$$h(s) = \begin{cases} a, \text{ if } s \in A \cap B \\ 1, \text{ if } s \in A \backslash B \\ ba, \text{ if } s \in B \backslash A \\ (ba)^\kappa, \text{ otherwise} \end{cases}$$

can recognize the language L. It is sufficient if we show: for all words $u \in \Sigma^\circledast$, $u \in \Sigma^\circledast B A^\circledast$ if and only if $h(u) \in \{a, ba\}$.

First we show the forward direction. If $u \in \Sigma^\circledast B A^\circledast$, then there exists a position i such that $u[i] \in B$ and for all $j > i$, we have $u[j] \in A$. In other words $h(u[i]) \in \{a, ba\}$ and $h(u_{>i}) \in \{1, a\}$. In all cases $h(u) \in \{a, ba\}$.

For the other direction, let us assume $h(u) \in \{a, ba\}$. First let us assume $h(u) = a$. Clearly $u \in A^\circledast (A \cap B) A^\circledast$ and therefore $u \in L$. So, let $h(u) = ba$. Then two cases can happen - either $u \in \Sigma^\circledast (B \backslash A) A^\circledast$ or $u = v_1 v_2$ where $h(v_1) = (ba)^\kappa$ and $h(v_2) = a$. In both cases $u \in L$. ∎

Below we show that the other direction is also true. That is, the marked languages definable by M^r can be recognized by boolean combinations of depth one formulas.

Lemma 12. *Consider the morphism $h : \Sigma^\circledast \to \mathrm{M}^r$. All marked languages definable using h can be recognized by a boolean combination of formulas of the form $A \mathbf{S} B$ and $A \dot{\mathbf{S}} B$.*

Similarly marked languages definable using $h : \Sigma^\circledast \to \mathrm{M}^l$ can be recognized by a boolean combination of formulas of the form $A \mathbf{U} B$ and $A \dot{\mathbf{U}} B$.

Proof: We show that marked languages definable using $h : \Sigma^\circledast \to \mathrm{M}^r$ can be recognized by a boolean combination of formulas of the form $A \mathbf{S} B$ and $A \dot{\mathbf{S}} B$. The case of M^l can be similarly proved. Let L be a marked language recognized by h. Let $\hat{h}$ the associated function, $\hat{h} : \Sigma_\#^\circledast \to (\mathrm{M}^r, \Sigma, \mathrm{M}^r)$ and $S \subseteq (\mathrm{M}^r, \Sigma, \mathrm{M}^r)$ such that $L = \hat{h}^{-1}(S)$. We will show that for all element $(m, s, n) \in S$, the marked language $\hat{h}^{-1}(m, s, n)$ is recognized by a boolean combination of since and until. Let $\hat{h}(w, i) = (m, s, n)$. It is easy to check that the marked position (i.e. $w[i]$) contains letter $s \in \Sigma$.

First, for each $m \in \mathrm{M}^r$, we give a formula which is true at position i if and only if $h(w_{<i}) = m$. For a set $S \subseteq \mathrm{M}^r$, we define the alphabet $\Sigma_S = \{s \in \Sigma \mid h(s) \in S\}$. Then $h^{-1}(m)$ for each $m \in \mathrm{M}^r$ can be recognized as follows:

$$\begin{aligned} &h^{-1}(1) = \mathbf{H}\Sigma_1 \\ &h^{-1}(a) = \mathbf{H}\Sigma_{\{1,a\}} \wedge \mathbf{P}\Sigma_a \\ &h^{-1}(ba) = \Sigma_1 \mathbf{S} \Sigma_{ba} \vee \left(\Sigma_{\{1,a\}} \mathbf{S} \Sigma_a \wedge \mathbf{P}\Sigma_{\{ba,(ba)^\kappa\}}\right) \end{aligned}$$

$h^{-1}((ba)^\kappa)$ is the set of all words not in $h^{-1}(x)$ for an $x \neq (ba)^\kappa$ and hence it is also definable: complement of the union of all the above languages.

Now, for each $n \in \mathrm{M}^r$, we give a formula which is true at position i if and only if $h(w_{>i}) = n$. Again, consider the sets Σ_S which were defined above. Note that if $n = 1$, then for all $k > i$ we have $w[k] \in \Sigma_1$. The following formula recognizes this: $\neg(\Sigma \dot{\mathbf{S}} \Sigma_{\{a,ba,(ba)^\kappa\}})$. Similarly if $n = a$, then for all $k > i$, we have $w[k] \in \Sigma_{\{1,a\}}$. This is also definable as shown before. Now consider the other two elements which are right zeros. If $n = ba$, then it could be because of two cases.

1) There is a position $k > i$ such that $w[k] \in \Sigma_{ba}$ and all future positions contains only letters from Σ_1.
2) There are two positions $k_1 > k_2 > i$ such that $w[k_2] \in \Sigma_{ba}$, $w[k_1] \in \Sigma_a$ and $w_{>k_1} \in \Sigma_{\{1,a\}}^\circledast$.

The following formula recognizes the above two conditions.

$$\Sigma_1 \dot{\mathbf{S}} \Sigma_{ba} \bigvee \left(\Sigma_{\{1,a\}} \dot{\mathbf{S}} \Sigma_a \wedge \mathbf{F}\Sigma_{\{ba,(ba)^\kappa\}}\right)$$

Note that the future operator ($\mathbf{F}$) can defined using $\dot{\mathbf{S}}$. Finally if $n = (ba)^\kappa$, the formula can be defined by a boolean combinations of the other formulas.

The three formulas in conjunction give a formula ϕ such that $\mathcal{L}_\#(\phi) = \hat{h}^{-1}(m, s, n)$. ∎

Until now we saw that languages definable by simple formulas (formulas of operator depth 1) are expressively equivalent to those definable by cartesian products of M^r and M^l. *Substitution* is a standard way to build temporal logic formulas of greater operator depths [22], [21]. Let ϕ be a formula over the alphabet Γ. Let $\{\psi_a\}_{a \in \Gamma}$ be a set of formulas over the alphabet Σ. Then $\phi[a \mapsto \psi_a]$ stands for the formula over Σ that we get from ϕ by replacing all occurrence of letters $a \in \Sigma$ by ψ_a. Substitution is inductively defined as follows.

1) $a[a \mapsto \psi_a] = \psi_a$
2) $(\alpha \vee \beta)[a \mapsto \psi_a] = \alpha[a \mapsto \psi_a] \vee \beta[a \mapsto \psi_a]$
3) $\neg\alpha[a \mapsto \psi_a] = \neg(\alpha[a \mapsto \psi_a])$
4) $(\alpha X \beta)[a \mapsto \psi_a] = (\alpha[a \mapsto \psi_a]) X (\beta[a \mapsto \psi_a])$

where X is one of the operators $\{\mathbf{S}, \mathbf{U}, \dot{\mathbf{S}}, \dot{\mathbf{U}}\}$. In the rest of the section we use substitution to show that LTL[S,U] is expressively equivalent to the weak block product of M^r and M^l. Let σ be a state-based transducer. The next lemma shows that under special circumstance LTL[S,U] is closed under σ^{-1}. That is, for any formula α we can find another formula β also in LTL[S,U] such that the marked word $(\sigma(w), i)$ satisfies α if and only if (w, i) satisfies β.

Lemma 13. *Let* $\Gamma = (\mathrm{M}^r \times \Sigma \times \mathrm{M}^r)$ *and* $\sigma : \Sigma^{\circledast} \to \Gamma^{\circledast}$ *be a state-based transducer. Then, there is a function* f : LTL[S,U] $\to$ LTL[S,U] *such that for all* $w \in \Sigma^{\circledast}$ *and* $\alpha \in$ LTL[S,U] *over* Γ

$$(\sigma(w), i) \models \alpha \Leftrightarrow (w, i) \models f(\alpha)$$

Proof: The proof is by structural induction. Lemma 12 gives $f(\alpha)$ for a letter $\alpha = (m, a, n) \in (\mathrm{M}^r \times \Sigma \times \mathrm{M}^r)$. Other formulas are inductively given as follows: $f(\alpha_1 \vee \alpha_2) = f(\alpha_1) \vee f(\alpha_2)$, $f(\neg\alpha) = \neg f(\alpha)$, $f(\alpha_1 X \alpha_2) = f(\alpha_1) X f(\alpha_2)$ where $X \in \{\mathbf{S}, \mathbf{U}, \dot{\mathbf{S}}, \dot{\mathbf{U}}\}$. It is easy to check that $(\sigma(w), i) \models \alpha \Leftrightarrow (w, i) \models f(\alpha)$ holds for the substitution f. ∎

We have built all the tools necessary to show that weak block products of M^r and M^l can be "simulated" by LTL[S,U].

Lemma 14. *Let N be an algebra such that* $\mathcal{L}(N) \subseteq$ LTL[S,U]. *Then,* $\mathcal{L}(\mathrm{M}^r \Box N) \subseteq$ LTL[S,U]

Proof: Let $\hat{N} = (\mathrm{M}^r \Box N)$ and consider the morphism $h : \Sigma^{\circledast} \to \hat{N}$. Our aim is to show that the languages recognized by h can be defined in LTL[S,U]. It is enough to show that for an arbitrary $\tau = (s, f) \in \hat{N}$, the language $h^{-1}(\tau)$ can be defined. Let $w \in \Sigma^{\circledast}$. By the block product principle, $w \in h^{-1}(\tau)$ if and only if $w \in L_1$ and $\sigma(w) \in L_2$ where L_1 and L_2 are languages definable in M^r and N respectively. From Lemma 12 there is a formula ϕ which defines L_1. From the assumption of this lemma, there is a formula ψ which defines the language L_2. Lemma 13 gives a formula $f(\psi)$ such that $\sigma(w) \in L_2$ if and only if $w \in \mathcal{L}(f(\psi))$. Thus the formula $\phi \wedge f(\psi)$ recognize the language $h^{-1}(\tau)$. ∎

Finally, we show the other direction of the above lemma. We observe that any LTL[S,U] formula can be built by iteratively substituting formulas of operator depth one [22].

Lemma 15. *Let α be an* LTL[S,U] *formula. Then $\mathcal{L}(\alpha)$ is recognized by weak block products of* M^l *and* M^r.

Proof: The proof is by induction on the operator depth of the formula α (see Proposition 5. in [21] for a similar proof). Let k be the operator depth of α. Lemma 11 gives the base case of operator depth one formulas. We will now prove the lemma when $k > 1$. Let us assume Σ to be the alphabet of α. Let $\Gamma = \{\gamma_1, \ldots, \gamma_n\}$ be all the subformulas of α of operator depth less than or equal to one (γ_is are formulas over Σ). Let $\Delta = \{a_1, \ldots, a_n\}$ be a set. Then, there exists a formula β over $\mathcal{P}(\Delta)$ whose operator depth is $k-1$ and such that $\beta[a_i \to \gamma_i]$ is the formula α. We now show that α can be recognized by weak block products of M^l and M^r. By induction hypothesis, β is recognized by an algebra $N \in \Box_{\mathsf{w}}^{*}\{\mathrm{M}^l, \mathrm{M}^r\}$. That is, there exists a morphism $g : \mathcal{P}(\Delta)^{\circledast} \to N$ and $T \subseteq N$ such that $g^{-1}(T) = \mathcal{L}(\beta)$.

From Lemma 11 we know there exists an algebra $M \in \prod \mathrm{M}^l \times \prod \mathrm{M}^r$ and a morphism $h : \Sigma^{\circledast} \to M$ and $S_i \subseteq M \times \Sigma \times M$ for $i \leq |\Gamma|$ such that $\hat{h}^{-1}(S_i) = \mathcal{L}_{\#}(\gamma_i)$. Consider the morphism which extends $f : M \times \Sigma \times M \to \mathcal{P}(\Delta)$ where $f(m, b, n) = \{a_i | (m, b, n) \in S_i\}$. Let the state-based transducer be $\sigma : \Sigma^{\oplus} \to (M \times \Sigma \times M)^{\oplus}$. Now for a word $w \in \Sigma^{\oplus}$ we claim the following

$$(f(\sigma(w)), x) \models \beta \Leftrightarrow (w, x) \models \beta[a_i \mapsto \gamma_i]$$

This can be proved by induction on the formula β. It is clear that the claim holds when β is a_i. The induction hypothesis goes through easily. So, let us now consider the composition of functions $g \circ f : M \times \Sigma \times M \to N$ where $g(f(m, b, n)) \in N$. From the above claim we get that

$$w \in \mathcal{L}(\alpha) \Leftrightarrow g(f(\sigma(w))) \in T$$

Now, using the morphisms h and $g \circ f$, we can construct a morphism into $M \Box N$ that recognizes $\mathcal{L}(\alpha)$. Note that, by Lemma 10, $M \Box N \in \Box_{\mathsf{w}}^{*}\{\mathrm{M}^l, \mathrm{M}^r\}$ This completes the proof of the lemma. ∎

We now state the main result of this section. Its proof follows from the previous two lemmas.

Theorem 6. $\mathcal{L}(\text{LTL[S,U]}) = \mathcal{L}(\Box_{\mathsf{w}}^{*}\{\mathrm{M}^r, \mathrm{M}^l\})$.

We remark that, in our analysis, to simulate every substitution we used one block product operation and vice versa. We believe that, with little more work, one should be able to algebraically characterize each class of k-operator depth formulas.

VI. Conclusion

We have incorporated block products into the recently developed rich algebraic framework for regular languages of countable words and provided a suitable block product principle. Our applications to logic demonstrate the uses of these constructs. More specificially, we have crucially used them to arrive at a Schützenberger-McNaughton-Papert theorem over countable words. We have also obtained a block-product based algebraic characterization of LTL with until-since temporal modalities.

We strongly believe that the block product construction presented in this work is well-suited for classifying several natural logics and the fragments thereof. Our work also exposes the possibility of Krohn-Rhodes theorem for finite monoids satisfying generalized associativity.

References

[1] Nicolas Bedon, Alexis Bès, Olivier Carton, and Chloe Rispal. Logic and rational languages of words indexed by linear orderings. *Theory Comput. Syst.*, 46(4):737–760, 2010.

[2] Alexis Bès and Olivier Carton. Algebraic characterization of FO for scattered linear orderings. In *Computer Science Logic, 25th International Workshop / 20th Annual Conference of the EACSL, CSL 2011*, pages 67–81, 2011.

[3] Véronique Bruyère and Olivier Carton. Automata on linear orderings. *J. Comput. Syst. Sci.*, 73(1):1–24, 2007.

[4] Olivier Carton. Wreath product and infinite words. *Journal of Pure and Applied Algebra*, 153(2):129 – 150, 2000.

[5] Olivier Carton, Thomas Colcombet, and Gabriele Puppis. An algebraic approach to MSO-definability on countable linear orderings. *J. Symb. Log.*, 83(3):1147–1189, 2018.

[6] Thomas Colcombet and A. V. Sreejith. Limited set quantifiers over countable linear orderings. In *Automata, Languages, and Programming - 42nd International Colloquium, ICALP 2015, Proceedings, Part II*, pages 146–158, 2015.

[7] Volker Diekert, Paul Gastin, and Manfred Kufleitner. A survey on small fragments of first-order logic over finite words. *Int. J. Found. Comput. Sci.*, 19(3):513–548, 2008.

[8] E. Allen Emerson. Modal and temporal logics. *Handbook of theoretical computer science*, B:995–1072, 1990.

[9] Dov M. Gabbay, Ian Hodkinson, and Mark Reynolds. *Temporal Logic: Mathematical Foundations and Computational Aspects, Volume 1*. Oxford University Press, Oxford, 1994.

[10] Amaldev Manuel and A. V. Sreejith. Two-variable logic over countable linear orderings. In *41st International Symposium on Mathematical Foundations of Computer Science, MFCS 2016*, pages 66:1–66:13, 2016.

[11] Robert McNaughton and Seymour A. Papert. *Counter-Free Automata (M.I.T. Research Monograph No. 65)*. The MIT Press, 1971.

[12] Jean-Eric Pin. Handbook of formal languages, vol. 1. chapter Syntactic Semigroups, pages 679–746. Springer-Verlag, Berlin, Heidelberg, 1997.

[13] Jean-Éric Pin. Mathematical foundations of automata theory. 2018.

[14] Michael O. Rabin. Decidability of second-order theories and automata on infinite trees. *Transactions of the American Mathematical Society*, 141:1–35, 1969.

[15] Chloe Rispal and Olivier Carton. Complementation of rational sets on countable scattered linear orderings. *Int. J. Found. Comput. Sci.*, 16(4):767–786, 2005.

[16] Joseph G. Rosenstein. *Linear orderings*. Academic Press New York, 1981.

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

[18] Howard Straubing. *Finite automata, formal logic, and circuit complexity*. Birkhauser Verlag, Basel, Switzerland, 1994.

[19] Howard Straubing and Denis Thérien. Weakly iterated block products of finite monoids. In *LATIN 2002: Theoretical Informatics, 5th Latin American Symposium, Proceedings*, pages 91–104, 2002.

[20] Pascal Tesson and Denis Thérien. Logic meets algebra: the case of regular languages. *Logical Methods in Computer Science*, 3(1), 2007.

[21] Denis Thérien and Thomas Wilke. Temporal logic and semidirect products: An effective characterization of the until hierarchy. *SIAM J. Comput*, 31(3):777–798, 2001.

[22] Denis Thérien and Thomas Wilke. Nesting until and since in linear temporal logic. *Theory Comput. Syst*, 37(1):111–131, 2004.

[23] Wolfgang Thomas. Handbook of formal languages, vol. 3. chapter Languages, Automata, and Logic, pages 389–455. Springer-Verlag New York, Inc., New York, NY, USA, 1997.

[24] Philipp Weis and Neil Immerman. Structure theorem and strict alternation hierarchy for FOˆ2 on words. *Logical Methods in Computer Science*, 5(3), 2009.

[25] Thomas Wilke. An algebraic theory for regular languages of finite and infinite words. *IJAC*, 3(4):447–490, 1993.