Stone Duality and the Substitution Principle*

Célia Borlido[1], Silke Czarnetzki[2], Mai Gehrke[3], and Andreas Krebs[4]

1 IRIF, CNRS and Université Paris Diderot, Paris, France
2 Wilhelm-Schickard Institut, Universität Tübingen, Tübingen, Germany
3 IRIF, CNRS and Université Paris Diderot, Paris, France
4 Wilhelm-Schickard Institut, Universität Tübingen, Tübingen, Germany

Abstract

In this paper we relate two generalisations of the finite monoid recognisers of automata theory for the study of circuit complexity classes: Boolean spaces with internal monoids and typed monoids. Using the setting of stamps, this allows us to generalise a number of results from algebraic automata theory as it relates to Büchi's logic on words. We obtain an Eilenberg theorem, a substitution principle based on Stone duality, a block product principle for typed stamps and, as our main result, a topological semidirect product construction, which corresponds to the application of a general form of quantification. These results provide tools for the study of language classes given by logic fragments such as the Boolean circuit complexity classes.

1998 ACM Subject Classification F.4.3 Formal Languages

Keywords and phrases $\mathcal{C}$-variety of languages, typed monoid, Boolean space with an internal monoid, substitution principle, semidirect product.

Digital Object Identifier 10.4230/LIPIcs.CSL.2017.13

1 Introduction

Complexity theory and the theory of regular languages are intimately connected through logic. As with classes of regular languages, many computational complexity classes are model classes of appropriate logic fragments on finite words [12]. For example, $\mathrm{AC}^0 = FO[\mathrm{arb}]$, $\mathrm{ACC}^0 = (FO + MOD)[\mathrm{arb}]$, and $\mathrm{TC}^0 = MAJ[\mathrm{arb}]$ where arb is the set of all predicates on the positions of a word, FO is first-order logic, and MOD and MAJ stand for the *modular* and *majority* quantifiers, respectively. On the one hand, the presence of *arbitrary (numerical) predicates*, and on the other hand, the presence of the *majority quantifier* is what brings one far beyond the scope of the profinite algebraic theory of regular languages.

Most results in complexity theory are proved with combinatorial, probabilistic, and algorithmic methods [18]. However, there are a few connections with the topo-algebraic tools for regular languages. A famous result of Barrington, Compton, Straubing, and Thérien [2] states that a regular language is in AC^0 if and only if its syntactic homomorphism is quasi-aperiodic. Although this result relies on [5] and no purely algebraic proof is known, being able to characterise the class of regular languages in AC^0 gives some hope that the non-uniform classes might be amenable to treatment by the generalised topo-algebraic methods.

Indeed, the hope is that one can generalise the tools of algebraic automata theory. In the paper *Logic Meets Algebra: the case of regular languages* [17], Thérien and Tesson lay

* This work has received funding from the European Research Council (ERC) under the European Union's Horizon 2020 research and innovation programme (grant agreement No.670624) and from the DFG Emmy Noether program (KR 4042/2).

26th EACSL Annual Conference on Computer Science Logic (CSL 2017).
Editors: Valentin Goranko and Mads Dam; Article No. 13; pp. 13:1–13:20
Leibniz International Proceedings in Informatics
LIPICS Schloss Dagstuhl – Leibniz-Zentrum für Informatik, Dagstuhl Publishing, Germany

out the theory used to characterise logic classes in the setting of regular languages in terms of their recognisers. Here we add topology to the picture, using Stone duality, to obtain corresponding tools that apply beyond the setting of regular languages.

Two ways of breaking the regular barrier already exist. Typed monoids [14], which still have a finite component, must be studied in families, while Boolean spaces with internal monoids [7, 10] generalise the profinite monoids of the classical theory and provide single topological objects of study relative to each alphabet. In Section 3 we relate these via an Eilenberg theorem improving on [3] by identifying tighter closure properties.

Using typed stamps, i.e. morphisms from finitely generated free monoids to typed monoids, we show that the Stone dual of predicate substitution is given by a transduction (Section 4), and that transduced languages are the ones recognised by a generalised version of the block product (Section 5). Thus we identify this important connection between logical generation and algebraic recognition as a case of Stone duality in a very general setting.

The topological recognisers provide access to equations as in Eilenberg-Reiterman theory [9], and thus to tools for separation and, in the finite case, decidability. The main result of the paper, Theorem 23, is a general form of the classical result by Almeida and Weil [1], which provides a semidirect product construction for Boolean spaces with dense monoids characterising the block product of varieties of typed stamps. Finally, in Section 7, we illustrate how these tools may be applied in the study of Boolean circuit classes.

2 Preliminaries on Stone duality

In this section we present the basics of Stone duality as used in the rest of the paper. See [6, 8] for an adapted introduction or [13] for further details.

The most basic duality we use, also known as *discrete duality*, provides a correspondence between powerset Boolean algebras (these are the complete and atomic Boolean algebras) and sets. Given such a Boolean algebra $\mathcal{B}$, its dual is its set of atoms, denoted $\mathsf{At}(\mathcal{B})$ and, given a set X, its dual is the Boolean algebra $\mathcal{P}(X)$. Clearly going back and forth yields isomorphic objects. If $h\colon \mathcal{B} \to \mathcal{A}$ preserves arbitrary meets and joins, the dual of h, denoted $\mathsf{At}(h)\colon \mathsf{At}(\mathcal{A}) \to \mathsf{At}(\mathcal{B})$ is given by the adjunction:

$$\forall\, a \in \mathcal{A} \ \text{ and } \ \forall\, x \in \mathsf{At}(\mathcal{B}) \quad (\ \mathsf{At}(h)(x) \leq a \iff x \leq h(a)\ )\,.$$

For example, if $\iota\colon \mathcal{B} \hookrightarrow \mathcal{P}(X)$ is the inclusion of a finite Boolean subalgebra of a powerset, then $\mathsf{At}(\iota)\colon X \twoheadrightarrow \mathsf{At}(\mathcal{B})$ is the quotient map corresponding to the finite partition of X given by the atoms of $\mathcal{B}$. Conversely, given a function $f\colon X \to Y$, the dual is just $\mathcal{P}(f) = f^{-1}\colon \mathcal{P}(Y) \to \mathcal{P}(X)$.

Generally Boolean algebras do not have enough atoms, and we have to consider *ultrafilters* instead (which may be seen as 'searches downwards' for atoms). Given an arbitrary Boolean algebra $\mathcal{B}$, an ultrafilter of $\mathcal{B}$ is a non-empty subset μ of $\mathcal{B}$ satisfying:

- μ is an upset, i.e., $a \in \mu$ and $a \leq b$ implies $b \in \mu$;
- μ is closed under finite meets, i.e., $a, b \in \mu$ implies $a \wedge b \in \mu$;
- for all $a \in \mathcal{B}$ exactly one of a and $\neg a$ is in μ.

Here, we will denote the set of ultrafilters of $\mathcal{B}$ by $X_{\mathcal{B}}$, and we will consider it as a topological space equipped with the topology generated by the sets $\widehat{a} = \{\mu \in X_{\mathcal{B}} \mid a \in \mu\}$ for $a \in \mathcal{B}$. The last property in the definition of ultrafilters implies that these basic open sets are also closed (and thus *clopen*). The resulting spaces are compact, Hausdorff, and have a basis of clopens. Such spaces are called *Boolean spaces*. Conversely, given a Boolean space, its clopen subsets form a Boolean algebra and one can show that going back and forth results in isomorphic objects. Given a homomorphism $h\colon \mathcal{A} \to \mathcal{B}$ between Boolean algebras, one can show that the inverse image of an ultrafilter is an ultrafilter and thus h^{-1} induces a continuous map $X_{\mathcal{B}} \to X_{\mathcal{A}}$. Finally, given a continuous function $f\colon X \to Y$, the inverse map restricts to clopens and yields a homomorphism of Boolean algebras.

Boolean spaces are also the *profinite sets*, see Appendix A. This is fundamental for the toggling between the two generalisations of finite monoids given below. Finally, for a set S, the dual space of $\mathcal{P}(S)$, denoted $\beta(S)$, is the *Stone-Čech compactification* of S. For each $s \in S$, the set $\iota(s) = \{P \in \mathcal{P}(S) \mid s \in P\}$ is the *principal ultrafilter generated by* s, and the induced map $\iota : S \hookrightarrow \beta(S)$ is an injection with dense image.

3 Recognition of languages

The notion of recognition originates in automata theory where the classical recognisers are monoid morphisms into finite monoids. Many important classes of regular languages correspond to *pseudovarieties of finite monoids*, that is, classes closed under homomorphic images, subalgebras, and finite products. Eilenberg's theorem characterises the corresponding classes of regular languages, called *varieties of regular languages.* All the languages recognised by a monoid of a given pseudovariety, no matter the recognising morphism used, belong to the corresponding variety of languages. However, not all language classes of interest form varieties. In fact, many classes corresponding to fragments of logic contain all the languages recognised by one morphism into a finite monoid but not the ones recognised via another morphism into the same monoid. It follows that such classes are not closed under inverse images for arbitrary morphisms between finitely generated free monoids, and one has to keep track of classes of morphisms from free monoids to finite ones, so-called *stamps*. Stamps, and a notion of pseudovariety of stamps, were introduced by Straubing [16] to study language classes coming from logic on words.

If one is interested in classes that contain non-regular languages, one needs more complex recognisers based on infinite monoids. In [7] a notion of compact topological recognisers was introduced and in [10] a more duality-friendly variant, so-called Boolean spaces with internal monoids, was given. These recognisers, based on Boolean spaces, provide a notion of recognition appropriate also for non-regular languages, but the finiteness is lost. Independently [3] introduced typed monoids, which are infinite monoids equipped with a finite set-quotient. These have a finite component but must be studied in families. Here we show how these two notions fit together and provide a variant of Eilenberg's variety theorem in the setting of stamps for languages that are not necessarily regular.

3.1 Generalising finite monoids

The basic recognisers of automata theory are finite monoids. Here, we consider recognisers which separate the attributes of being finite and of being a monoid.

▶ **Definition 1.** A *typed monoid* is a tuple $\mathsf{R} = (M, p, X)$, where X is a finite set, M is a monoid and $p\colon M \twoheadrightarrow X$ is a surjective set function. A *typed submonoid* of R is given by a submonoid N of M by restriction of p and its image, $\mathsf{R}|_N = (N, p|_N, p[N])$. A *morphism of typed monoids*, $\Phi\colon (M, p, X) \to (N, q, Y)$, is a pair $\Phi = (g, \varphi)$ where $g\colon M \to N$ is a monoid morphism and $\varphi\colon X \to Y$ is a set function, so that $\varphi \circ p = q \circ g$. The *image* of Φ is the typed submonoid of (N, q, Y) given by $g[N]$. We say that (M, p, X) *recognises* the language $L \subseteq A^*$ when there is a monoid morphism $\mu : A^* \to M$ and $C \subseteq X$ such that $L = (p \circ \mu)^{-1}(C)$.

For instance, $L_{\mathrm{Eq}} = \{w \in \{a,b\}^* \mid |w|_a = |w|_b\}$, where $|w|_a$ and $|w|_b$ stand, respectively, for the number of a's and of b's in w, is recognised by $(\mathbb{Z}, p, \{0,1\})$, where $p(n) = 1$ if and only if $n = 0$. The recognising morphism $h_{\mathrm{E}q}$ sends a to 1 and b to -1 and $L_{\mathrm{Eq}} = h_{\mathrm{E}q}^{-1}(\{1\})$.

Finite monoids provide invariants that are useful in describing and understanding classes of regular languages. Their pertinence arises from the fact that many classes of languages of interest are closed under *quotients* by words. That is, if $L \subseteq A^*$ is in the class and $u \in A^*$ then $u^{-1}L = \{w \in A^* \mid uw \in L\}$ and $Lu^{-1} = \{w \in A^* \mid wu \in L\}$ are also in the class. Note that the Boolean algebra of all languages recognised by a morphism $\mu \colon A^* \to M$ is closed under quotients. However, this is not true for typed monoids in general and, given a non-regular language, the closure under quotienting is necessarily infinite, so we need to capture infinite Boolean subalgebras of $\mathcal{P}(M)$ for M an infinite monoid. For this purpose, we recall that a *biaction* of the monoid M on a set X is given by a two-sided action map $\alpha \colon M \times X \times M \to M$ satisfying $\alpha(1, x, 1) = x$ where 1 is the identity element of M and $\alpha(m_1, \alpha(m_1', x, m_2'), m_2) = \alpha(m_1 m_1', x, m_2' m_2)$ for all $m_1, m_1', m_2', m_2 \in M$. The *left component* of the action at m is the map $\lambda_m \colon X \to X$ given by $x \mapsto \alpha(m, x, 1)$, while the *right component* at m is $\rho_m \colon x \mapsto \alpha(1, x, m)$. As derived in [10], the following topological notion captures Boolean subalgebras of a monoid which are closed under the quotient operations.

▶ **Definition 2.** A *Boolean space with an internal monoid* (BiM) is a triple (M, p, X) where X is a Boolean space equipped with a biaction of a monoid M whose right and left components at each $m \in M$ are continuous and a map $p \colon M \to X$ which has dense image and is a morphism of sets with M-biactions. That is, for each $m \in M$, the following diagrams commute:

$$\begin{tikzcd}
M \arrow[r, "p"] \arrow[d, "\ell_m"'] & X \arrow[d, "\lambda_m"] & & M \arrow[r, "p"] \arrow[d, "r_m"'] & X \arrow[d, "\rho_m"] \\
M \arrow[r, "p"] & X & & M \arrow[r, "p"] & X
\end{tikzcd}$$

with $\ell_m \colon n \mapsto m \cdot n$ and $r_m \colon n \mapsto n \cdot m$ the components at m of the biaction of M on itself.

▶ **Example 3.** The Boolean algebra closed under quotients that is generated by L_{Eq} defined above is recognised by the BiM $(\mathbb{Z}, p, \mathbb{Z} \cup \{\infty\})$ via the morphism h_{Eq}. Here, $\mathbb{Z} \cup \{\infty\}$ is the one-point compactification of $\mathbb{Z}$, p is just the inclusion map, and the left and right actions are given by the usual addition on $\mathbb{Z}$ augmented by $n + \infty = \infty = \infty + n$, for every $n \in \mathbb{Z}$.

The weaker structure (M, p, X) where M is a monoid, X is a Boolean space, and p has dense image we will call a *Boolean space with a dense monoid*, or simply a *B-monoid.* Note that typed monoids are precisely the finite B-monoids (those for which X is finite). Moreover, a *B-submonoid* of (M, p, X) is given by a submonoid N of M by restricting p and viewing it as a map into the topological closure of its image.

Alternatively, we can capture infinite Boolean subalgebras of $\mathcal{P}(M)$ closed under the quotients using subfamilies $\mathcal{S} \subseteq \mathcal{T}_M = \{(M, p, X) \mid (M, p, X) \text{ is a typed monoid}\}$. For this purpose, we define a quasiorder on $\mathcal{T}_M$ given by $(M, p, X) \geq (M, q, Y)$ provided (M, q, Y) is a quotient of (M, p, X) for which the map g is the identity. The family $\mathcal{S}$ is then a *downset* of $\mathcal{T}_M$ provided it is closed under such quotients, and $\mathcal{S}$ is *directed* provided $(M, p, X), (M, q, Y) \in \mathcal{S}$ implies the existence of $(M, r, Z) \in \mathcal{S}$ with quotient maps $p' \colon Z \to X$ and $q' \colon Z \to Y$ so that $p = p' \circ r$ and $q = q' \circ r$. Also, we say that $\mathcal{S}$ is *multiplicative* provided for all $(M, p, X) \in \mathcal{S}$ and $m \in M$, there are $(M, q, Y), (M, r, Z) \in \mathcal{S}$ and maps $\lambda_m \colon Y \to X$ and $\rho_m \colon Z \to X$ so that $\lambda_m \circ q(m') = p(mm')$ and $\rho_m \circ r(m') = p(m'm)$, for every $m' \in M$.

Stone duality yields the following proposition.

▶ **Proposition 4.** *For a monoid M, there are bijections between each of the following:*

1. *the set of Boolean subalgebras closed under quotients $\mathcal{B} \subseteq \mathcal{P}(M)$;*
2. *the set of directed and multiplicative downsets $\mathcal{S} \subseteq \mathcal{T}_M$ of typed monoids based on M;*
3. *the set of Boolean spaces with internal monoids (M, p, X) based on M.*

Proof Sketch. The bijections are given as follows. Given a Boolean subalgebra $\mathcal{B}$ of $\mathcal{P}(M)$ closed under quotients, the corresponding directed and multiplicative downset is

$$\mathcal{S}_{\mathcal{B}} = \{(M, \mathsf{At}(\iota), \mathsf{At}(\mathcal{B}')) \mid \mathcal{B}' \subseteq \mathcal{B} \text{ is a finite Boolean subalgebra and } \iota\colon \mathcal{B}' \hookrightarrow \mathcal{P}(M)\}.$$

In turn, given a directed and multiplicative downset of typed monoids $\mathcal{S} \subseteq \mathcal{T}_M$, the limit of the projective system of maps $p_i : M \twoheadrightarrow X_i$ with $(M, p_i, X_i) \in \mathcal{S}$ defines a BiM, see Appendix A. Finally, given a BiM (M, p, X), the corresponding Boolean algebra is

$$\mathcal{B}_{(M,p,X)} = \{p^{-1}(K) \mid K \subseteq X \text{ is clopen}\}.$$ ◀

3.2 Stamps for non-regular languages

▶ **Definition 5.** A *Boolean space with a dense stamp* (or *B-stamp* for short) is a tuple $\mathsf{R} = (A, \mu, M, p, X)$ where $\mu : A^* \twoheadrightarrow M$ is a monoid quotient and (M, p, X) is a B-monoid. A B-stamp is called a BiM *presentation* when (M, p, X) is a BiM and a *typed stamp* when (M, p, X) is a typed monoid.

A morphism between B-stamps $\mathsf{R} = (A, \mu, M, p, X)$ and $\mathsf{S} = (B, \nu, N, q, Y)$ is a triple $\Phi = (h, g, \varphi)$, where $h : A^* \to B^*$ and $g : M \to N$ are monoid morphisms, $\varphi : X \to Y$ is a continuous function, and the following diagram commutes:

$$\begin{tikzcd}
A^* \arrow[r, "\mu"] \arrow[d, "h"'] & M \arrow[r, "p"] \arrow[d, "g"] & X \arrow[d, "\varphi"] \\
B^* \arrow[r, "\nu"] & N \arrow[r, "q"] & Y
\end{tikzcd}$$

When h is the identity on A^*, we say that S *factors through* R. Further, each morphism $h : A^* \to B^*$ defines a *substamp* $(h[A], \nu', N', q', Y')$ of S, where the elements of $h[A]$ are regarded as letters, $\nu' : h[A]^* \to N'$ is the surjective co-restriction of the monoid morphism sending $u \in h[A]$ to $\nu(u)$, and (N', q', Y') is the sub of (N, q, Y) given by N'.

B-stamps R encode two types of behaviour: algebraic behaviour given by the *monoid presentation* (A, μ, M) and topological behaviour given by the B-monoid (M, p, X). The interplay between these two is a key ingredient in this paper. We say that a language L is *recognised* by R provided there exists a clopen subset $C \subseteq X$ such that $L = (p \circ \mu)^{-1}(C)$. Notice that the set of all languages recognised by a B-stamp always forms a Boolean algebra.

▶ **Definition 6.** Let $\mathcal{B} \subseteq \mathcal{P}(A^*)$ be a Boolean algebra of languages. Then the *syntactic B-stamp* of $\mathcal{B}$ is $\mathsf{R}_{\mathcal{B}} = (A, \mu_{\mathcal{B}}, M_{\mathcal{B}}, p_{\mathcal{B}}, X_{\mathcal{B}})$ where $X_{\mathcal{B}}$ is the dual space of $\mathcal{B}$ and $M_{\mathcal{B}}{=}A^*/{\sim_{\mathcal{B}}}$ where the *syntactic congruence* $\sim_{\mathcal{B}}$ is given by

$$w \sim_{\mathcal{B}} w' \quad \iff \quad \forall L \in \mathcal{B},\ \forall u, v \in A^* \quad (uwv \in L \iff uw'v \in L).$$

The quotient map $\mu_{\mathcal{B}}\colon A^* \twoheadrightarrow A^*/{\sim_{\mathcal{B}}}$ is the *syntactic morphism* of $\mathcal{B}$. The restriction to A^* of the dual of the inclusion $\iota\colon \mathcal{B} \hookrightarrow \mathcal{P}(A^*)$ is a map $\tilde{\iota} : A^* \to X_{\mathcal{B}}$ with dense image. Since $\sim_{\mathcal{B}}$ is the least monoid congruence containing the kernel of $\tilde{\iota}$, it factors through $\mu_{\mathcal{B}}$. That is, there is a map $p_{\mathcal{B}}\colon M_{\mathcal{B}} \to X_{\mathcal{B}}$ so that $\tilde{\iota} = p_{\mathcal{B}} \circ \mu_{\mathcal{B}}$. Clearly, the image of $p_{\mathcal{B}}$ is dense in X.

▶ **Proposition 7.** *A B-stamp* R *recognises a Boolean algebra closed under quotients* $\mathcal{B}$ *if and only if the syntactic B-stamp of* $\mathcal{B}$ *factors through* R.

3.3 $\mathcal{C}$-varieties of languages and an Eilenberg theorem

As mentioned earlier, we are interested in classes of languages that may not be closed under preimages of arbitrary morphisms. In what follows, we fix a class $\mathcal{C}$ of morphisms between finitely generated free monoids which is closed under composition and contains all length-preserving morphisms (i.e. the ones sending generators to generators), written *lp-morphism* (we will see why *lp*-morphisms are important in the treatment of logic on words in Section 4.2). A *$\mathcal{C}$-variety of languages* is an assignment, for each finite alphabet A, of a Boolean algebra closed under quotients $\mathcal{V}(A)$ of languages over A such that, for every morphism $h : B^* \to A^*$ in $\mathcal{C}$, if $L \in \mathcal{V}(A)$ then $h^{-1}(L) \in \mathcal{V}(B)$. In this subsection, we identify the classes of typed stamps that correspond to $\mathcal{C}$-varieties of languages. For this, we need some definitions.

A morphism $\Phi = (h, g, \varphi)$ of typed stamps is a *$\mathcal{C}$-morphism* provided $h \in \mathcal{C}$ and a *$\mathcal{C}$-substamp* is a typed substamp given by $h \in \mathcal{C}$. Let $\{\mathsf{R}_i{=}(A, \mu_i, M_i, p_i, X_i)\}_{i\in I}$ be a family of B-stamps. Their *product* is the B-stamp $\odot_{i\in I}\mathsf{R}_i = (A, \mu, M, p, X)$ where $\mu\colon w \mapsto (\mu_i(w))_{i\in I}$ is the surjective co-restriction of the product map and (M, p, X) is the B-submonoid given by the image of μ in the product of the (M_i, p_i, X_i)'s. This construction yields a B-stamp, the product of BiM presentations is a BiM presentation, but the product of typed stamps may not be a typed stamp. Thus we define the *restricted product with respect to a finite subset* $F \subseteq I$ to be the B-stamp obtained from the product by composing $p\colon M \to X$ with the restriction of $\prod_{j\in F} \pi_j$ where $\pi_j\colon \Pi X_i \to X_j$ are the projections. Note that every restricted product of typed stamps is a typed stamp.

Let M be a monoid and $\mathcal{S} \subseteq \mathcal{T}_M$. We say $\mathcal{S}$ is *separating* provided whenever $m_1, m_2 \in M$ are distinct, there is $(M, p, X) \in \mathcal{S}$ such that $p(m_1) \neq p(m_2)$. Finally, for a monoid presentation (A, μ, M) and a family of typed stamp $\mathcal{F}$, denote by $\mathsf{Typ}_{\mathcal{F}}(A, \mu, M)$ the set of all typed monoids (M, p, X) such that (A, μ, M, p, X) belongs to $\mathcal{F}$. A monoid quotient $g\colon M \twoheadrightarrow N$ is said to be *$\mathcal{F}$-compatible* provided $\{(N, q, Y) \in \mathcal{T}_N \mid (M, q \circ g, Y) \in \mathsf{Typ}_{\mathcal{F}}(A, \mu, M)\}$ is separating. A stamp $\mathsf{S}{=}(A, \nu, N, q, Y)$ is an *$\mathcal{F}$-compatible quotient* of $\mathsf{R}{=}(A, \mu, M, p, X)$ provided S factors through R and the corresponding quotient map $M \twoheadrightarrow N$ is $\mathcal{F}$-compatible.

▶ **Definition 8.** A family of typed stamps V is called a *$\mathcal{C}$-pseudovariety* provided

(V.1) V is closed under taking $\mathcal{C}$-substamps and V-compatible quotients;

(V.2) V is closed under taking arbitrary restricted products;

(V.3) $\mathsf{Typ}_V(A, \mu, M)$ is a multiplicative and separating downset whenever (A, μ, M) is a monoid presentation of a typed stamp in V.

We are now able to state the main result of this section.

▶ **Theorem 9.** *There is a one-to-one correspondence between $\mathcal{C}$-varieties of languages and $\mathcal{C}$-pseudovarieties of typed stamps.*

Proof Sketch. The correspondence is given as follows: for each $\mathcal{C}$-pseudovariety of typed stamps V, we define $\mathcal{V}(A)$ to be the set of all languages over A that are recognised by some element of V. Conversely, given a $\mathcal{C}$-variety of languages $\mathcal{V}$, for each finite alphabet A, the syntactic BiM of $\mathcal{V}(A)$ is a BiM presentation R_A. In turn, the BiM presentations that factor through R_A are given by projective limit systems of typed stamps. Altogether, these form a $\mathcal{C}$-pseudovariety. Finally one can show that these two constructions are inverse to each other. ◀

Observe that, unlike what happens for classical stamps on finite monoids, not all classes of typed stamps generate $\mathcal{C}$-pseudovarieties. For this to be the case, the multiplicative closure of the class has to be separating for each of the monoid presentations of the class.

3.4 Using profinite alphabets

Let V be a $\mathcal{C}$-pseudovariety of typed stamps, and $\mathcal{V}$ the corresponding $\mathcal{C}$-variety of languages. For each finite alphabet A, the class of typed stamps in V based on A forms a projective limit system whose projective limit is a BiM presentation, which we will denote by $\Sigma_A(V)$. As indicated by the proof sketch for Theorem 9, the point is that $\Sigma_A(V)$ is also the syntactic B-stamp of $\mathcal{V}(A)$. In case $\mathcal{V}$ consists entirely of regular languages, $\Sigma_A(V)$ is essentially what is usually denoted by $\overline{\Omega}_A(V_{\text{mon}})$ or $\hat{F}_A(V_{\text{mon}})$ and is called the *free A-generated pro-*V_{mon} *monoid*, where V_{mon} is the pseudovariety of all finite monoids M for which (M, id, M) is the B-monoid component of a stamp in V.

As in the setting of regular languages, it is sometimes useful to extend varieties to profinite alphabets. This will be crucial for our main result, Theorem 23. Let Y be a profinite alphabet and let $\{\pi_i : Y \twoheadrightarrow Y_i\}_{i\in I}$ be the set of all finite continuous quotients of Y. Further let $\{h_{i,j} : Y_i \twoheadrightarrow Y_j \mid \pi_j = h_{i,j} \circ \pi_i\}$ be the diagram of all quotient maps commuting with the projections. Then this is a projective limit system, and it is a well-known fact that Y is the projective limit of this system (see Appendix A). On the other hand, since $\mathcal{V}$ is a $\mathcal{C}$-variety, each of the maps $h_{i,j}$ defines a unique monoid quotient map $h^*_{i,j} : Y_i^* \twoheadrightarrow Y_j^*$, which dually defines an embedding of Boolean algebras $(h^*_{i,j})^{-1} : \mathcal{V}(Y_j) \hookrightarrow \mathcal{V}(Y_i)$. Therefore, using a slight adaptation of Proposition 7, we have that there exists a morphism of the form $\Phi_{i,j} = (h_{i,j}, g_{i,j}, \varphi_{i,j})$ from $\Sigma_{Y_i}(V)$ to $\Sigma_{Y_j}(V)$. Thus, the family $\{\Sigma_{Y_i}(V)\}_{i\in I}$ defines a projective system with connecting morphisms $\Phi_{i,j}$.

▶ **Proposition 10.** *The projective limit of the family $\{\Sigma_{Y_i}(\mathrm{V})\}_{i\in I}$ exists and it is a* BiM *presentation (Y, μ, M, p, X) on the profinite alphabet Y, which we denote by $\Sigma_Y(\mathrm{V})$.*

4 Logic on Words

Logic on words (see, e.g. [15]), is a logical language whose intended models are *words*, that is, elements of the free monoid A^*, for a fixed finite alphabet A. We shall consider formulas that are recursively built as follows:

- *letter predicates* are formulas: for each $a \in A$, we have a letter predicate $\mathsf{P}_a(x)$;
- *numerical predicates* are formulas: given $k \geq 0$, a k-ary numerical predicate is given by a relation R that assigns to each element n of $\mathbb{N}$ a subset R_n of $\{1, \ldots, n\}^k$ (for instance, the (binary) numerical predicate $x < y$ assigns to each n the set $\{(i, j) \mid 1 \leq i < j \leq n\}$);
- *Boolean combinations* of formulas are formulas: if φ and ψ are formulas, then so are $\varphi \wedge \psi$, $\varphi \vee \psi$, and $\neg\varphi$;
- *unary quantification* of a formula with respect to a variable is a formula: a quantifier is given by a map $\mathsf{Q} : \{0, 1\}^* \to \{0, 1\}$ (for instance, the existential quantifier $\exists$ maps the word $\varepsilon_1 \cdots \varepsilon_k \in \{0, 1\}^*$ to 1 if and only if there exists an index i such that $\varepsilon_i = 1$).

Sentences with letter predicates from A have words of A^* as models. More generally, models of formulas with free variables in $\Upsilon = \{x_1, \ldots, x_k\}$ are given by Υ-*structures*, $w_{x_1=i_1,\ldots,x_k=i_k}$, where $w \in A^*$ and $i_1, \ldots, i_k \in \{1, \ldots, |w|\}$ (cf. [15, Chapter II]). We denote by $A^* \otimes \Upsilon$ the set of all Υ-structures. The semantics of a formula can then be defined inductively as follows. The $\{x\}$-structure $w_{x=i}$ satisfies $\mathsf{P}_a(x)$ if and only if its i-th letter is an a, and $w_{x_1=i_1,\ldots,x_k=i_k}$ satisfies the atomic formula $R(x_1, \ldots, x_k)$, where R is a k-ary numerical predicate, if and only if $(i_1, \ldots, i_k)$ belongs to $R_{|w|}$. The Boolean connectives *and* $\wedge$, *or* $\vee$, and *not* $\neg$ are interpreted classically. A *quantifier* $\mathsf{Q} : \{0, 1\}^* \to \{0, 1\}$ is interpreted as follows: w satisfies $\mathsf{Q}x\ \varphi(x)$ if and only if $\mathsf{Q}(\delta_{\varphi(x)}(w)) = 1$, where $\delta_{\varphi(x)} : A^* \to \{0, 1\}^*$ sends $w = a_1 \cdots a_k$ to $\varepsilon_1 \cdots \varepsilon_k$ where $\varepsilon_i = 1$ if and only if $w_{x=i}$ satisfies $\varphi(x)$. Important examples of quantifiers are given by the *existential quantifier* $\exists$ mentioned above, *modular quantifiers* $\exists_q^r$, for each $q \in \mathbb{N}$ and $0 \leq r < q$, mapping a word of $\{0,1\}^*$ to 1 if and only if the number of 1's is congruent to r modulo q, and the *majority quantifier* Maj which sends an element of $\{0,1\}^*$ to 1 if and only if it has strictly more occurrences of 1 than of 0. Finally, for a fixed set of free variables Υ, given a formula φ with free variables in Υ, we denote by L_φ the set of all Υ-structures satisfying φ.

In what follows, we fix a logical signature with set of numerical predicates $\mathcal{N}$ and set of unary quantifiers $\mathcal{Q}$. For each finite alphabet A and each finite set of variables Υ, we denote by $\mathcal{Q}_A[\mathcal{N}](\Upsilon)$ the corresponding set of first-order formulas with free variables in Υ, and letter predicates for $a \in A$. For reasons which will become apparent in Section 4.1, we are interested in studying fragments of logic allowing the alphabet to vary.

▶ **Definition 11.** A *logic class* Γ is a map that associates to each finite alphabet A and finite set Υ of first-order variables, a set of formulas $\Gamma_A(\Upsilon) \subseteq \mathcal{Q}_A[\mathcal{N}](\Upsilon)$ which satisfies the following properties:

(LC.1) If $\Upsilon \subseteq \Upsilon'$, then $\Gamma_A(\Upsilon) \subseteq \Gamma_A(\Upsilon')$;

(LC.2) Each set $\Gamma_A(\Upsilon)$ is closed under Boolean connectives $\wedge$ and $\neg$ (and thus, under $\vee$);

(LC.3) Every map $\zeta \colon A \to B$ between finite alphabets induces a map $\zeta_{\Gamma,\Upsilon} : \Gamma_B(\Upsilon) \to \Gamma_A(\Upsilon)$ which sends $\varphi \in \Gamma_B(\Upsilon)$ to the formula obtained by substituting, for every occurrence in φ of a predicate $\mathsf{P}_b(x)$ with $b \in B$, the formula $\bigvee_{\zeta(a)=b} \mathsf{P}_a(x)$. Note that if b is not in the image of ζ, then $\mathsf{P}_b(x)$ is replaced by the empty join, which is logically equivalent to the *always-false* proposition.

The intuitive idea behind the definition of a logic class is to model what is usually referred to as a *fragment of logic*. For instance, considering all formulas of quantifier depth less than k in a given logic defines a logic class.

We consider formulas up to semantic equivalence, and thus, by (LC.2), each $\Gamma_A(\Upsilon)$ is a Boolean algebra. The associated partial order relation is given by semantic implication: $\varphi \leq \psi$ if and only if $L_\varphi \subseteq L_\psi$. That is, $\Gamma_A(\Upsilon)$ is isomorphic to the Boolean algebra of languages $\{L_\varphi \mid \varphi \in \Gamma_A(\Upsilon)\}$.

Restricting to sentences, we obtain a language class $A \mapsto \mathcal{L}_A(\Gamma) = \{L_\varphi \mid \varphi \in \Gamma_A(\emptyset)\}$. Note that property (LC.3) of logic classes implies that the associated class of languages is closed under inverse images of *lp*-morphisms. Therefore, if each $\mathcal{L}_A(\Gamma)$ is closed under quotienting by words, then the language class is (at least) an *lp*-variety.

4.1 Substitution

The concept of substitution for the study of logic on words, as in [17], is quite different from substitution in predicate logic. Substitution in predicate logic works on terms, whereas the notion of substitution in [17] works at the propositional level of the predicate logic. As such it provides a method for decomposing complex formulas into simpler ones. The core idea is to enrich the alphabet over which the logic is defined in order to be able to substitute large subformulas through letter predicates.

Roughly speaking, *substitution* is a tool for decomposing formulas into simpler ones. For instance, the sentence $\psi = \exists x\ \varphi(x)$ may be obtained from the sentence $\exists x\ \mathsf{P}_b(x)$ by replacing $\mathsf{P}_b(x)$ by $\varphi(x)$. Then, understanding ψ amounts to understanding both the sentence $\exists x\ \mathsf{P}_b(x)$ and the formula $\varphi(x)$. If we want to substitute away several subformulas in this way, we must account for their logical relations. The idea of an alphabet is that the corresponding predicates $\{\mathsf{P}_a(x)\}_{a \in A}$ interpret in any word as a finite set $\mathcal{F}$ of formulas satisfying:

(A.1) $\vee_{\varphi \in \mathcal{F}}\ \varphi$ is the *always-true* proposition;

(A.2) for every $\varphi_1, \varphi_2 \in \mathcal{F}$ distinct, $\varphi_1 \wedge \varphi_2$ is the *always-false* proposition.

We formalise this concept of substitution. Suppose we are given two logic classes Λ and Γ.[1] Further let C be a finite alphabet and $\xi : C \to \Lambda_A(\Upsilon \cup \{x\})$, where $x \notin \Upsilon$, a map whose image satisfies (A.1) and (A.2). Note that, this is equivalent to requiring that the formulas in the image of ξ are precisely the atoms of the Boolean algebra $\mathcal{B}$ generated by the image of ξ. Now we may define the *substitution* given by ξ to be the map $\sigma_\xi : \Gamma_C(\emptyset) \to \mathcal{Q}_A[\mathcal{N}](\Upsilon)$ defined by substituting for any occurrence of a letter predicate $\mathsf{P}_c(z)$ in a sentence $\psi \in \Gamma_C(\emptyset)$, the formula $\xi(c)(x/z)$ (that is, the formula obtained by substituting z for x in the formula $\xi(c) \in \Lambda_A(\Upsilon \cup \{x\})$). Note that here we assume (without loss of generality) that things have been arranged so that the variables occurring in ψ, such as z, do not occur in the formulas in the image of ξ. Since the only constraints on the interpretation of letters in a word are given by the properties (A.1) and (A.2), it follows by a simple structural induction that σ_ξ is a morphism of Boolean algebras. We denote the image of this morphism by $\Gamma \circ \mathcal{B}$.

Note this name makes sense as $\Gamma \circ \mathcal{B}$ is uniquely determined by $\mathcal{B}$. Indeed, instead of starting from a map ξ whose image satisfies (A.1) and (A.2), we could start with a finite Boolean subalgebra $\mathcal{B}$ of $\Lambda_A(\Upsilon \cup \{x\})$. Then we obtain a finite alphabet by letting $C_\mathcal{B} = \mathsf{At}(\mathcal{B})$ and a map $\xi_\mathcal{B}$ given by the inclusion of $\mathsf{At}(\mathcal{B})$ in $\Lambda_A(\Upsilon \cup \{x\})$. The resulting substitution, which we will denote by $\sigma_\mathcal{B}$ (instead of $\sigma_{\xi_\mathcal{B}}$), has $\Gamma \circ \mathcal{B}$ as its image. With this notation in place, we can now compare the substitution maps obtained for different finite Boolean subalgebras of $\Lambda_A(\Upsilon \cup \{x\})$. In fact, in what follows, we shall argue that the concept of substitution is naturally extendable to infinite Boolean algebras.

Let $\mathcal{B}_1 \subseteq \mathcal{B}_2$ be finite Boolean subalgebras of $\Gamma_A(\Upsilon \cup \{x\})$. Then the dual of the inclusion map $\mathcal{B}_1 \hookrightarrow \mathcal{B}_2$ sends each atom of $\mathcal{B}_2$ to the unique atom of $\mathcal{B}_1$ that is above it (in the order on $\mathcal{B}_2$). This yields a map $\xi_{1,2} : C_2 \twoheadrightarrow C_1$, where $C_i = \mathsf{At}(\mathcal{B}_i)$, for $i = 1, 2$, are the corresponding alphabets. Now since Γ is a language class, by property (LC.3), $\xi_{1,2}$ yields a morphism $\iota_{1,2} \colon \Gamma_{C_1}(\emptyset) \to \Gamma_{C_2}(\emptyset)$. Moreover, since every element of $\mathcal{B}_1$ is logically equivalent to the disjunction of the atoms of $\mathcal{B}_2$ below it, the morphism $\iota_{1,2}$ is in fact an embedding and the following diagram commutes:

$$\begin{tikzcd}
\Gamma_{C_1}(\emptyset) \arrow[dr, "\sigma_{\mathcal{B}_1}"] \arrow[dd, tail, "\iota_{1,2}"'] & \\
& \mathcal{Q}_A[\mathcal{N}](\Upsilon) \\
\Gamma_{C_2}(\emptyset) \arrow[ur, "\sigma_{\mathcal{B}_2}"'] &
\end{tikzcd}$$

That is, $\mathcal{B}_1 \subseteq \mathcal{B}_2$ yields $\Gamma \circ \mathcal{B}_1 \subseteq \Gamma \circ \mathcal{B}_2$. Thus we have a direct system of subalgebras of $\mathcal{Q}_A[\mathcal{N}](\Upsilon)$ (see Appendix A). This allows us to extend the composition to infinite Boolean algebras of formulas and thereby also to language classes.

▶ **Definition 12.** Let Λ and Γ be logic classes. For each finite set Υ of variables, and each finite alphabet A, we define

$$(\Gamma \circ \Lambda)_A(\Upsilon) = \Big\langle \varinjlim \{\Gamma \circ \mathcal{B} \mid \mathcal{B} \subseteq \Lambda_A(\Upsilon \cup \{x\}) \text{ is a finite Boolean algebra}\} \cup \Lambda_A(\Upsilon) \Big\rangle_{\mathsf{BA}},$$

where the connecting morphisms are the inclusions $\Gamma \circ \mathcal{B}_1 \subseteq \Gamma \circ \mathcal{B}_2$ whenever $\mathcal{B}_1 \subseteq \mathcal{B}_2$.

▶ **Proposition 13.** *Let Λ and Γ be logic classes. Then, $\Gamma \circ \Lambda$ is also a logic class.*

[1] We will only use the sentences of Γ, so we may assume that Γ consists entirely of sentences. It may for example be all the depth one sentences using some quantifier of interest.

4.2 Substitution and Duality: The Substitution Principle

Substitution allows us to describe logic classes as compositions of simpler ones. In Section 5 we treat recognition for languages given by formulas obtained by substitution in terms of recognisers of the components. Key ingredients for this are the dual of the substitution map, which we describe in Proposition 14, and the concrete description of the languages given by formulas obtained by substitution (Corollary 15) that it provides.

Let Γ and Λ be logic classes and A a finite alphabet. In this subsection, our goal is to describe the languages in $\mathcal{L}_A(\Gamma \circ \Lambda)$, or equivalently, the languages defined by sentences of $(\Gamma \circ \Lambda)_A(\emptyset)$. If $\mathcal{B}$ is a finite Boolean subalgebra of $\Lambda_A(x)$ and $C_{\mathcal{B}}{=}\mathsf{At}(\mathcal{B})$, then $\mathcal{B}$ defines a substitution morphism $\sigma_{\mathcal{B}}\colon \Gamma_{C_{\mathcal{B}}}(\emptyset) \to (\Gamma \circ \Lambda)_A(\emptyset)$, which may be seen as a morphism $\sigma_{\mathcal{B}}\colon \mathcal{L}_{C_{\mathcal{B}}}(\Gamma) \to \mathcal{L}_A(\Gamma \circ \Lambda)$ between the corresponding Boolean algebras of languages given by $L_\psi \mapsto L_{\sigma_{\mathcal{B}}(\psi)}$. Let $\Sigma_{\mathcal{B}}$ be the Stone dual of $\sigma_{\mathcal{B}}$. Since $\mathcal{L}_A(\Gamma \circ \Lambda)$ and $\mathcal{L}_{C_{\mathcal{B}}}(\Gamma)$ are Boolean algebras of languages over A^* and $C_{\mathcal{B}}^*$, respectively, we have maps $p\colon A^* \to X_{\mathcal{L}_A(\Gamma\circ\Lambda)}$ and $q\colon C_{\mathcal{B}}^* \to X_{\mathcal{L}_{C_{\mathcal{B}}}(\Gamma)}$ with dense images obtained as the restrictions of the dual maps of the inclusions $\mathcal{L}_A(\Gamma \circ \Lambda) {\hookrightarrow} \mathcal{P}(A^*)$ and $\mathcal{L}_{C_{\mathcal{B}}}(\Gamma) {\hookrightarrow} \mathcal{P}(C_{\mathcal{B}}^*)$, respectively. A continuous map $X_{\mathcal{L}_A(\Gamma\circ\Lambda)} {\to} X_{\mathcal{L}_{C_{\mathcal{B}}}(\Gamma)}$ need not restrict to the dense images of the monoids, however this is indeed the case for the maps $\Sigma_{\mathcal{B}}$. This is a consequence of the following stronger result.

▶ **Proposition 14.** *Let $\mathcal{B}$ be a finite Boolean subalgebra of $\Lambda_A(x)$, $C_{\mathcal{B}} = \mathsf{At}(\mathcal{B})$, and let $\xi\colon A^* \otimes \{x\} \to C_{\mathcal{B}}$ be the dual to the embedding $\mathcal{B} \hookrightarrow \mathcal{P}(A^* \otimes \{x\})$. Then, the function $\tau_{\mathcal{B}} : A^* \to C_{\mathcal{B}}^*$ defined by $\tau(w) = \xi(w_{x=1}) \cdots \xi(w_{x=|w|})$ makes the diagram commute:*

$$\begin{tikzcd}
A^* \arrow[r, dashed, "\tau_{\mathcal{B}}"] \arrow[d, "rp"'] & C_{\mathcal{B}}^* \arrow[d, "q"] \\
X_{\mathcal{L}_A(\Gamma\circ\Lambda)} \arrow[r, "\Sigma_{\mathcal{B}}"] & X_{\mathcal{L}_{C_{\mathcal{B}}}(\Gamma)}
\end{tikzcd}$$

Recall, by Definition 12, that $(\Gamma \circ \Lambda)_A(\emptyset)$ is generated as a Boolean algebra by $\Lambda_A(\emptyset)$ and the sentences of the form $\sigma_{\mathcal{B}}(\psi)$ for $\psi \in \Gamma_{C_{\mathcal{B}}}(\emptyset)$ for $\mathcal{B}$ a finite Boolean subalgebra of $\Lambda_A(x)$. From the proof of Proposition 14 (see (4) in Appendix C.2), we obtain a concrete description of the languages corresponding to $\mathcal{L}_A(\Gamma \circ \Lambda)$.

▶ **Corollary 15** (Substitution Principle)**.** *The Boolean algebra $\mathcal{L}_A(\Gamma \circ \Lambda)$ is generated by the languages of $\mathcal{L}_A(\Lambda)$ together with the languages of the form $\tau_{\mathcal{B}}^{-1}(K)$, where $\mathcal{B} \subseteq \Lambda_A(x)$ is a finite Boolean subalgebra and $K \in \mathcal{L}_{C_{\mathcal{B}}}(\Gamma)$.*

Dualising the direct limit system $\{\Gamma \circ \mathcal{B}\}_{\mathcal{B}}$ discussed in Section 4.1, we obtain a projective system (see Appendix A):

▶ **Proposition 16.** *The class of morphisms*

$$\{\tau_{\mathcal{B}} : A^* \to C_{\mathcal{B}}^* \mid \mathcal{B} \subseteq \Lambda_A(x) \textit{ is a finite Boolean subalgebra}\}$$

forms a projective limit system, where the connecting morphisms are the morphisms of monoids $C_{\mathcal{B}_2}^* \to C_{\mathcal{B}_1}^*$ induced by the dual maps of inclusions $\mathcal{B}_1 \hookrightarrow \mathcal{B}_2$. The limit of this system is the map $\tau : A^* \to X_{\Lambda_A(x)}^*$ sending a word $w \in A^*$ to the word $\mu_1 \cdots \mu_{|w|}$ with $\mu_i = \{\varphi(x) \in \Lambda_A(x) \mid w_{x=i}$ satisfies $\varphi(x)\}$, and $X_{\Lambda_A(x)}$ is the dual of $\Lambda_A(x)$.

Note that the maps $\tau_{\mathcal{B}}$ are all length preserving, thus the above system factors into projective limit systems $\{\tau_{\mathcal{B}} : A^n \to (C_{\mathcal{B}})^n \mid \mathcal{B} \subseteq \Lambda_A(x)$ is a finite Boolean subalgebra$\}$ for each $n \in \mathbb{N}$, and each of these systems has a profinite limit in the usual sense. The space $X_{\Lambda_A(x)}^*$ obtained in Proposition 16 is then seen as the union over $n \in \mathbb{N}$ of the spaces $X_{\Lambda_A(x)}^n$, each one of those being a Boolean space when equipped with the product topology.

5 The Block Product Principle

It is well-known that when the logic classes considered define only regular languages, the decomposition of first-order formulas given by the Substitution Principle is modelled by the usual block product of finite monoids (see [17] for a survey). In this section, we consider a notion of block product that generalises the usual one from a logic perspective, thereby obtaining a Block Product Principle for *lp*-pseudovarieties of typed stamps (cf. Theorem 21).

The Substitution Principle justifies the interest in studying the family of maps $\tau_{\mathcal{B}}$ parameterised by finite Boolean subalgebras $\mathcal{B} \subseteq \Lambda_A(x)$. In turn, each language defined by a formula $\varphi(x)$ may be seen as a language over the extended alphabet $A \times 2^{\{x\}}$, and therefore it is recognised by some typed stamp over the alphabet $A \times 2^{\{x\}}$. The following notion of *transduction* encodes the maps $\tau_{\mathcal{B}}$ in terms of recognition (cf. Proposition 18).

▶ **Definition 17.** Let $\mathsf{S} = (A \times 2^{\{x\}}, \nu, N, q, Y)$ be a typed stamp. We let C_{S} be the alphabet contained in Y given by the image $q \circ \nu[A^* \otimes \{x\}]$. The *transduction determined by* S is the map $\tau_{\mathsf{S}} : A^* \to C_{\mathsf{S}}^*$ given by $\tau_{\mathsf{S}}(w) = q \circ \nu(w_{x=1}) \cdots q \circ \nu(w_{x=|w|})$.

▶ **Proposition 18.** *Let $\mathcal{B} \subseteq \Lambda_A(x)$ be a finite Boolean algebra, and $\mathsf{S} = (A \times 2^{\{x\}}, \nu, N, q, Y)$ be the syntactic typed stamp of the Boolean subalgebra of $\mathcal{P}((A \times 2^{\{x\}})^*)$ generated by the set of languages definable by a formula of $\mathcal{B}$. Then, C_{S} is isomorphic as a set to $C_{\mathcal{B}} = \mathsf{At}(\mathcal{B})$, and $\tau_{\mathsf{S}} = \tau_{\mathcal{B}}$.*

Now, in view of the Substitution Principle and of Proposition 18, our goal is to identify the recognisers of languages of the form $\tau_{\mathsf{S}}^{-1}(K)$ when $K \subseteq C_{\mathsf{S}}^*$ is recognised by a given typed stamp R. The block product $\mathsf{R} \Box \mathsf{S}$, that we introduce in Definition 19 below, has the property of recognising these languages and little more (cf. Proposition 20).

Recall that, given monoids M and N, their *block product*, $M \Box N$, is the monoid with underlying set $M^{N \times N} \times N$ and binary operation given by $(f, n)(f', n') = (h, n \cdot n')$, where $h(n_1, n_2) = f(n_1, n' \cdot n_2) + f'(n_1 \cdot n, n_2)$. Here, in order to improve readability, we are denoting the operation on M additively and that on N multiplicatively, although neither is assumed to be commutative.

▶ **Definition 19.** Let $\mathsf{R} = (C_{\mathsf{S}}, \mu, M, p, X)$ and $\mathsf{S} = (A \times 2^{\{x\}}, \nu, N, q, Y)$ be typed stamps. For each letter $a \in A$, we define the function $f_a : N \times N \to M$ by

$$f_a(n_1, n_2) = \mu \circ q(n_1\, \nu(a, \{x\})\, n_2),$$

and we let $\mathcal{F}_{\mathsf{R},\mathsf{S}}$ be the set of all such functions. Then, the *block product* of R and S, denoted $\mathsf{R} \Box \mathsf{S}$, is the typed stamp

$$(A, \mu \Box_q \nu, \langle \mathcal{F}_{\mathsf{R},\mathsf{S}} \times \nu(A) \rangle, p \Box q, X \times Y),$$

where $\langle \mathcal{F}_{\mathsf{R},\mathsf{S}} \times \nu(A) \rangle$ is the submonoid of the usual block product of monoids $M \Box N$ generated by $\mathcal{F}_{\mathsf{R},\mathsf{S}} \times \nu(A)$, $\mu \Box_q \nu$ is the unique homomorphism mapping $a \in A$ to $(f_a, \nu(a))$, and $p \Box q$ sends (f, n) to $(p \circ f(1, 1), q(n))$.

We now state the local version of the *Block Product Principle*.

▶ **Proposition 20.** *Let $\mathsf{R} = (C_{\mathsf{S}}, \mu, M, p, X)$ and $\mathsf{S} = (A \times 2^{\{x\}}, \nu, N, q, Y)$ be typed stamps. Then, a language $L \subseteq A^*$ is recognised by $\mathsf{R} \Box \mathsf{S}$ if and only if it is a Boolean combination of languages of the form $\tau_{\mathsf{S}}^{-1}(K_1) \cap K_2$ for some language $K_1 \subseteq C_{\mathsf{S}}^*$ recognised by R and some language $K_2 \subseteq A^*$ recognised by the lp-substamp of S defined by the injective map $A \hookrightarrow A \times 2^{\{x\}}$ sending a to $(a, \emptyset)$.*

Given *lp*-pseudovarieties V and W of typed stamps, we denote by $V\Box W$ the *lp*-pseudovariety generated by $\mathsf{R}\Box\mathsf{S}$ for $\mathsf{R}\in V$ and $\mathsf{S}\in W$ typed stamps over suitable alphabets. Note that, the multiplicative closure of this set of block products forms a separating set for each of the monoid presentations. We denote by $\mathcal{V}\boxplus\mathcal{W}$ the corresponding *lp*-variety of languages (recall Theorem 9). As a consequence of Proposition 20 we obtain:

▶ **Theorem 21** (Block Product Principle). *Let* V *and* W *be lp-pseudovarieties of typed stamps. Then, the Boolean algebra* $(\mathcal{V}\boxplus\mathcal{W})(A)$ *is generated by the languages of* $\mathcal{W}(A)$ *together with the languages of the form* $\tau^{-1}(K)$*, where* τ *is a transduction defined by an element of* W *and* K *belongs to* $\mathcal{V}(C)$ *for a suitable alphabet* C*.*

6 A generalisation of Semidirect Product

In Section 5, we showed that the block product of typed stamps is suitable for the recognition of languages defined by formulas obtained by substitution. In this section, we describe the structure of the syntactic B-stamp $\Sigma_A(V\Box W)$ of the Boolean algebra of languages recognised by some element of $V\Box W$, when V and W are two given *lp*-pseudovarieties (cf. Theorem 23). This may be considered as analogue to the result of Almeida and Weil [1] describing the A-generated free pro-$(V{*}{*}W)$ monoid as a two-sided semidirect product of the free pro-V and pro-W monoids over suitable alphabets. In their result, V and W are pseudovarieties of finite monoids in the usual sense (recall the beginning of Section 3.4) and $V{*}{*}W$ denotes the pseudovariety generated by the (usual) block product of monoids, $M\Box N$, for $M\in V$ and $N\in W$.

Let M and N be monoids. Again, we denote the operation on M additively and that on N multiplicatively.

A *monoid biaction of* N *on* M is a biaction of N on the underlying set of M, satisfying the following additional properties:

- $n_1\cdot(m_1+m_2)\cdot n_2 = n_1\cdot m_1\cdot n_2 + n_1\cdot m_2\cdot n_2$, for all $n_1,n_2\in N$, and $m_1,m_2\in M$;
- $n_1\cdot 0\cdot n_2 = 0$, for all $n_1,n_2\in N$.

Given such a biaction, we may define a new monoid, called the *two-sided semidirect product*, usually denoted by $M{*}{*}N$. It has underlying set $M\times N$, and operation defined by

$$(m_1,n_1)(m_2,n_2) = (m_1\cdot n_2 + n_1\cdot m_2, n_1n_2).$$

An example of this construction is given by the usual block product of monoids.

More generally, if $\mathsf{R} = (M,p,X)$ and $\mathsf{S} = (N,q,Y)$ are B-monoids and N bi-acts on M, one may define the *two-sided semidirect product* of R and S to be the B-monoid $(M{*}{*}N, p\times q, X\times Y)$ where $M{*}{*}N$ is the usual semidirect product of monoids with respect to the given biaction. Note that, even if R and S are both BiM's, the resulting semidirect product need not be a BiM. Let $\Sigma_{A\times 2}(W) = (A\times 2^{\{x\}},\nu,N,q,Y)$ and $\Sigma_Y(V) = (Y,\mu,M,p,X)$. In order to understand the structure of the BiM component of $\Sigma_A(V\Box W)$, we start by showing that N naturally bi-acts on X. Intuitively, that is a consequence of $\mathcal{V}(C)$ being closed under quotients for every finite alphabet C.

▶ **Proposition 22.** *There is a biaction of* N *on* X *whose left and right components at each* $n\in N$*,* $\psi_{\ell,n}$ *and* $\psi_{r,n}$*, respectively, are continuous maps that satisfy*

$$\psi_{\ell,n}\circ p\circ\mu(y_1,\dots,y_k) = p\circ\mu(\lambda_n(y_1),\dots,\lambda_n(y_k)), \tag{1}$$

$$\psi_{r,n}\circ p\circ\mu(y_1,\dots,y_k) = p\circ\mu(\rho_n(y_1),\dots,\rho_n(y_k)). \tag{2}$$

In particular, N bi-acts on M and hence, there is a well-defined two-sided semidirect product $(M, p, X){*}{*}(N, q, Y)$. Moreover, each homomorphism $h : C^* \to M{*}{*}N$ defines a B-stamp $\Sigma_Y(\mathit{V}){*}{*}\Sigma_{A\times 2}(\mathit{W})$ over the alphabet C, whose B-monoid component is a B-submonoid of $(M, p, X){*}{*}(N, q, Y)$. We are now able to state the main result of this section.

▶ **Theorem 23.** *Let* V *and* W *be lp-pseudovarieties of typed stamps. Then, the* BiM *presentation* $\Sigma_A(\mathrm{V} \,\Box\, \mathrm{W})$ *is isomorphic to a two-sided semidirect product* $\Sigma_Y(\mathrm{V}){*}{*}\Sigma_{A\times 2}(\mathrm{W})$*.*

Proof Sketch. Let $\Sigma_A(\mathit{V}\Box\, \mathit{W}) = (A, \eta, P, r, Z)$, and let $\tau : A^* \to Y^*$ be the map given by

$$\tau(w) = (q \circ \nu(w_{x=1}), \dots, q \circ \nu(w_{x=|w|})),$$

for $w \in A^*$. In other words, the co-restriction of τ to its image is the limit of the projective system described in Proposition 16. We claim that the homomorphism $h : A^* \to M{*}{*}N$ sending the word w to the pair $(\mu\circ\tau(w), \nu(w))$ defines a semidirect product $\Sigma_Y(\mathit{V}){*}{*}\Sigma_{A\times 2}(\mathit{W})$, which is isomorphic to $\Sigma_A(\mathit{V}\Box\, \mathit{W})$.

Using the Block Product Principle, we may prove the existence of an onto morphism of Boolean algebras $\mathsf{Clopen}(X) \oplus \mathsf{Clopen}(Y) \twoheadrightarrow (\mathcal{V} \,\Box\, \mathcal{W})(A)$, which dually defines an embedding of Boolean spaces $\theta : Z \hookrightarrow X \times Y$. On the other hand, one may also prove the inclusion $\ker(\eta) \subseteq \ker(h)$, and thus, there is a homomorphism $g : P \to M{*}{*}N$ satisfying $g \circ \eta = h$ and $\theta \circ r = (p \times q) \circ g$. In particular, g is necessarily injective and this proves the claim. ◀

7 Applications to logic

In this section, we show how to apply the results of the paper to decompose a logic class $\mathcal{Q}[\mathcal{N}]$ under the assumption that it admits a prenex normal form. That is, each formula is equivalent to one in which a string of quantifiers is applied to a quantifier free formula. Requiring a logic signature to admit prenex normal forms is a mild condition which is satisfied as soon as the logical language is sufficiently expressive. Indeed, this is the case for many standard classes considered in Boolean circuit complexity, such as those referred to in the introduction.

Given a set of quantifiers $\mathcal{Q}$, we let $\Gamma_{\mathcal{Q}}$ be the logic class of sentences assigning to the finite alphabet A the Boolean algebra generated by formulas of the form $\mathsf{Q}x \bigvee_{a\in B} \mathsf{P}_a(x)$, with $\mathsf{Q} \in \mathcal{Q}$ and $B \subseteq A$. In turn, if $\mathcal{N}$ is a set of numerical predicates, then $\Lambda_{\mathcal{N}}$ is the logic class in which $\Lambda_{\mathcal{N},A}(\Upsilon)$ consists of all Boolean combinations of numerical predicates from $\mathcal{N}$, letter predicates for letters in A, and having free variables in Υ.

Suppose we are given a finite Boolean subalgebra $\mathcal{B} \subseteq \Lambda_{\mathcal{N},A}(\{x_1, \dots, x_{k-1}, x_k\})$. The substitution map defined by $\mathcal{B}$ in Section 4.1 has image $\Gamma_{\mathcal{Q}} \circ \mathcal{B}$, which is a finite Boolean subalgebra of $(\Gamma_{\mathcal{Q}} \circ \Lambda_{\mathcal{N}})_A(\{x_1, \dots, x_{k-1}\})$ (when we take $\Upsilon = \{x_1, \dots, x_{k-1}\}$). We may now consider the substitution map defined by $\Gamma_{\mathcal{Q}} \circ \mathcal{B}$, thereby obtaining a finite Boolean subalgebra of $(\Gamma_{\mathcal{Q}} \circ (\Gamma_{\mathcal{Q}} \circ \Lambda_{\mathcal{N}}))_A(\{x_1, \dots, x_{k-2}\})$. By successively iterating the operation $\Lambda \mapsto (\Gamma_{\mathcal{Q}} \circ \Lambda)$, one is able to produce all the prenex normal form sentences of $\mathcal{Q}_A[\mathcal{N}](\emptyset)$ of a given quantifier depth. Thus we obtain:

▶ **Proposition 24.** *Let* $\mathcal{Q}$ *be a set of quantifiers and* $\mathcal{N}$ *a set of numerical predicates such that every formula of* $\mathcal{Q}_A[\mathcal{N}]$ *admits a prenex normal form. Then,*

$$\mathcal{Q}_A[\mathcal{N}](\emptyset) = \bigcup_{n\in\mathbb{N}} \underbrace{\Gamma_{\mathcal{Q}} \circ (\cdots \circ (\Gamma_{\mathcal{Q}}}_{n\ times} \circ \Lambda_{\mathcal{N}}) \dots)_A(\emptyset).$$

In order to be able to apply the results of previous section and, through Proposition 24, obtain recognisers for the languages definable in $\mathcal{Q}[\mathcal{N}]$, one should identify the *lp*-pseudovarieties of typed stamps that recognise the languages definable in the logic classes $\Gamma_{\mathcal{Q}}$ and $\Lambda_{\mathcal{N}}$.

▶ **Proposition 25.** *Given a quantifier* $\mathsf{Q} : \{0,1\}^* \to \{0,1\}$*, we let* L_{Q} *be the language* $\mathsf{Q}^{-1}(1)$ *and* R_{Q} *its syntactic typed stamp. Then, a language is defined by a sentence of* $\Gamma_{\mathcal{Q},A}(\emptyset)$ *if and only if it is recognised by a restricted product of typed lp-substamps of* R_{Q} *over the alphabet* A*.*

As a consequence, we have that the lp-variety of languages $\mathcal{V}_{\mathcal{Q}}$ mapping the finite alphabet A to the closure under quotients of the Boolean algebra of languages definable in $\Gamma_{\mathcal{Q},A}(\emptyset)$ corresponds to the lp-pseudovariety of typed stamps that is generated by the elements of the form R_{Q}, with $\mathsf{Q} \in \mathcal{Q}$ (recall Theorem 9).

We illustrate Proposition 25 by instantiating Q with some of the most relevant examples of quantifiers appearing in the literature.

▶ **Example 26.** Let $\mathbb{B} = \{0,1\}$ be the two-element join semilattice. Then, $\mathsf{R}_{\exists}$ is given by the typed stamp $(\{0,1\}, \mu_{\exists}, \mathbb{B}, p_{\exists}, \{0,1\})$, where both $\mu_{\exists}$ restricted to $\{0,1\}$ and $p_{\exists}$ are the identity map between the respective underlying sets. On the other hand, for the quantifier $\exists_q^r$, one has $\mathsf{R}_{\exists_q^r} = (\{0,1\}, \mu_{\exists_q^r}, \mathbb{Z}/q\mathbb{Z}, p_{\exists_q^r}, \{0,1\})$, where $\mu_{\exists_q^r}(0) = [0]$, $\mu_{\exists_q^r}(1) = [1]$, and $p_{\exists_q^r}([k])$ is mapped to 1 if and only if $k - r$ is divisible by q. Finally, one may check that $\mathsf{R}_{\mathsf{Maj}}$ is the typed stamp $(\{0,1\}, \mu_{\mathsf{Maj}}, \mathbb{Z}, p_{\mathsf{Maj}}, \{0,1\})$ with $\mu_{\mathsf{Maj}}(0) = -1$, $\mu_{\mathsf{Maj}}(1) = 1$, and $p_{\mathsf{Maj}}(k) = 1$ if and only if $k > 0$.

Next, we illustrate how to define a typed stamp recognising a given numerical predicate. The intuitive idea is to use a product of monoids of the form $M_1 \times M_2 \times M_3$ where M_1 encodes the length of a word, M_2 encodes the first position marked by each variable, while M_3 encodes the last one. Let $(R : n \mapsto R_n)$ be a k-ary numerical predicate, and $\Upsilon = \{x_1, \ldots, x_k\}$ a set of k variables. For each $\ell \in \mathbb{N}$ we define the map $\alpha_\ell : \mathbb{B} \to \mathbb{N}$ by setting $\alpha_\ell(0) = \ell$ and $\alpha_\ell(1) = 0$. We let $\mathbb{N} \circ \mathbb{B}$ and $\mathbb{N} \circ_r \mathbb{B}$ be, respectively the usual wreath and reversed wreath product of monoids, and N the submonoid of $\mathbb{N} \times (\mathbb{N} \circ \mathbb{B})^k \times (\mathbb{N} \circ_r \mathbb{B})^k$ generated by $\{1\} \times (\{\alpha_1\} \times \mathbb{B})^k \times (\{\alpha_1\} \times \mathbb{B})^k$. An easy computation shows that every element of N belongs to $\mathbb{N} \times (\{\alpha_\ell\}_{\ell \in \mathbb{N}} \times \mathbb{B})^k \times (\{\alpha_\ell\}_{\ell \in \mathbb{N}} \times \mathbb{B})^k$. For each subset $S \subseteq \Upsilon$ and each variable $x \in \Upsilon$ we set $n_{S,x} = (\alpha_1, 1)$ if $x \in S$ and $n_{S,x} = (\alpha_1, 0)$ otherwise. Finally, we let $\mathsf{S}_R = (A \times 2^\Upsilon, \nu_R, N, q_R, \{0,1\})$ be the typed stamp defined by

$$\begin{aligned} \nu_R(a,S) &= (1, (n_{S,x_i})_{i=1}^k, (n_{S,x_i})_{i=1}^k) \\ q_R(n, (\alpha_{\ell_i}, \varepsilon_i)_{i=1}^k, (\alpha_{k_i}, \delta_i)_{i=1}^k) = 1 &\iff \forall i \quad (\varepsilon_i = \delta_i = 1 \text{ and } k_i = n - \ell_i + 1) \\ &\qquad \text{and } (\ell_1, \ldots, \ell_k) \in R_n. \end{aligned}$$

▶ **Proposition 27.** *The typed stamp* S_R *recognises* L_R*, the language of* $(A \times 2^\Upsilon)^*$ *defined by* R*, via the subset* $\{1\}$*.*

We let $V_{\mathcal{N}}$ be the lp-pseudovariety generated by the typed stamps S_R, for $R \in \mathcal{N}$ an $|\Upsilon|$-ary numerical predicate.

Combining these constructions with Proposition 24, we have the following:

▶ **Proposition 28.** *Let* $\mathcal{Q}$ *be a set of quantifiers and* $\mathcal{N}$ *a set of numerical predicates such that every formula of* $\mathcal{Q}_A[\mathcal{N}]$ *admits a prenex normal form. Every language definable in* $\mathcal{Q}[\mathcal{N}]$ *is recognised by some typed stamp in*

$$\bigcup_{n \in \mathbb{N}} \underbrace{\mathrm{V}_{\mathcal{Q}} \mathbin{\square} (\ldots \mathbin{\square} (\mathrm{V}_{\mathcal{Q}}}_{n\ times} \mathbin{\square} \mathrm{V}_{\mathcal{N}}) \ldots). \tag{3}$$

On the other hand, as a consequence of the next result, we have that the elements of (3) do not recognise much more languages than the ones definable in $\mathcal{Q}[\mathcal{N}]$.

▶ **Proposition 29.** *Let* S *be the syntactic typed stamp of a formula* $\varphi \in \mathcal{Q}_A[\mathcal{N}](\Upsilon)$. *Then, every language recognised by a typed lp-substamp of* S *is definable in* $(\mathrm{FO} + \mathcal{Q})[\{=\} \cup \mathcal{N}]$.

Proof Sketch. We consider the case where $\Upsilon = \{x\}$ and $\varphi(x)$ has one free variable x. The general one is handled similarly. Let $\mathsf{S} = (A \times 2^{\{x\}}, \nu, N, q, Y)$ be the syntactic typed stamp of $\varphi(x)$, let B be a finite alphabet, and $h : B^* \to (A \times 2^{\{x\}})^*$ an *lp*-morphism. It suffices to show that the language $L = (q \circ \nu \circ h)^{-1}(1)$, is definable in $(\mathcal{Q} + FO)[\mathcal{N} \cup \{=\}]$. Take

$$\phi = \exists ! z \ \varphi(z) \wedge (\vee_{h(b) \in A \times \{\{x\}\}} \mathsf{P}_b(z)).$$

Then, one may check that $L = L_\phi$. ◀

We finish this section with an example of application of Theorem 23.

▶ **Example 30.** Let $\mathcal{N}$ be a set of unary numerical predicates and $\mathcal{Q} = \{\exists\}$.

We write $\Sigma_{A \times 2^{\{x\}}}(V_{\mathcal{N}}) = (A \times 2^{\{x\}}, \nu, N, q, Y)$. By Example 26 we have that $V_{\mathcal{Q}}$ is the *lp*-pseudovariety generated by $\mathsf{R}_\exists$. Applying Theorem 23 and using a well-known fact about the structure of the space component of $\Sigma_Y(V_{\mathcal{Q}})$, we may derive that the BiM component of $\Sigma_A(V_{\mathcal{Q}} \Box V_{\mathcal{N}})$ is the *Schützenberger product of* (N, q, Y) introduced in [10]. In the mentioned paper, the goal was to define a recogniser for the language $L_{\exists x \ \varphi(x)}$ given a recogniser for $L_{\varphi(x)}$, and the Schützenberger product of a BiM recognising $L_{\varphi(x)}$ played that role. We thus provide a different explanation for their result.

8 Discussion

In Proposition 14, we have shown that the transductions are bona fide duals of substitution in terms of Stone duality. This is crucial to the link between logic on words and topo-algebraic methods. Our main result, Theorem 23, allows one to compute the syntactic space of the Boolean algebra obtained by applying a very general form of quantifier to a variety of not necessarily regular languages. This result was first proved by Almeida and Weil in [1] for varieties of regular languages and quantifiers given by such. Several surprises have to be overcome for the generalisation: the closure properties of pseudovarieties of typed stamps are delicate, in the block-product of typed monoids, the 'right' finitely generated monoid must be identified. Finally, the B-monoid based on the semidirect product in Theorem 23 is *not* a BiM – even though the B-submonoid we seek is a BiM. More recently, the result of Theorem 23 in the special cases of the classical existential quantifier [10] and of quantifiers arising from finite semirings [11] has been obtained using codensity monads and transducers.

Boolean spaces are promising objects for separation results. As laid out in Section 7, the block product and substitution principle and their connection to semidirect products of B-monoids allow us, by induction, to compute the relevant spaces for important Boolean circuit complexity classes. The equations satisfied by these spaces may in turn help in finding a way of showing separations [9, 4].

Acknowledgements. The authors would like to thank Howard Straubing for fruitful discussions on the interaction of logic and algebraic language theory as well as the anonymous referees whose comments helped improve the final presentation of the paper.

References

1 Jorge Almeida and Pascal Weil. Free profinite semigroups over semidirect products. *Russian Mathematics*, 39(1):1–27, 1995.

2 David A. Mix Barrington, Kevin J. Compton, Howard Straubing, and Denis Thérien. Regular Languages in NC^1. *J. Comput. Syst. Sci.*, 44(3):478–499, 1992. doi:10.1016/0022-0000(92)90014-A.

3 Christoph Behle, Andreas Krebs, and Stephanie Reifferscheid. Typed Monoids – An Eilenberg-like Theorem for non regular Languages. *Electronic Colloq. on Comp. Complexity (ECCC)*, 18:35, 2011. URL: https://eccc.weizmann.ac.il/report/2011/035/.

4 Silke Czarnetzki and Andreas Krebs. Using duality in circuit complexity. In *Language and Automata Theory and Applications – 10th International Conference, LATA 2016, Prague, Czech Republic, March 14-18, 2016, Proceedings*, pages 283–294, 2016. doi:10.1007/978-3-319-30000-9_22.

5 Merrick L. Furst, James B. Saxe, and Michael Sipser. Parity, circuits, and the polynomial-time hierarchy. In *22nd Annual Symposium on Foundations of Computer Science, Nashville, Tennessee, USA, 28-30 October 1981*, pages 260–270, 1981. doi:10.1109/SFCS.1981.35.

6 Mai Gehrke. Duality in computer science. In *Proceedings of the 31st Annual ACM/IEEE Symposium on Logic in Computer Science, LICS'16, New York, NY, USA, July 5-8, 2016*, pages 12–26, 2016. doi:10.1145/2933575.2934575.

7 Mai Gehrke, Serge Grigorieff, and Jean-Eric Pin. A topological approach to recognition. In *Automata, Languages and Programming, 37th International Colloquium, ICALP 2010, Bordeaux, France, July 6-10, 2010, Proceedings, Part II*, pages 151–162, 2010. doi:10.1007/978-3-642-14162-1_13.

8 Mai Gehrke and Andreas Krebs. Stone duality for languages and complexity. *SIGLOG News*, 4(2):29–53, 2017. doi:10.1145/3090064.3090068.

9 Mai Gehrke, Andreas Krebs, and Jean-Éric Pin. Ultrafilters on words for a fragment of logic. *Theor. Comput. Sci.*, 610:37–58, 2016. doi:10.1016/j.tcs.2015.08.007.

10 Mai Gehrke, Daniela Petrisan, and Luca Reggio. The Schützenberger Product for Syntactic Spaces. In *43rd International Colloquium on Automata, Languages, and Programming, ICALP 2016, July 11-15, 2016, Rome, Italy*, pages 112:1–112:14, 2016. doi:10.4230/LIPIcs.ICALP.2016.112.

11 Mai Gehrke, Daniela Petrişan, and Luca Reggio. Quantifiers on languages and codensity monads. To appear in LICS, 2017. URL: https://arxiv.org/abs/1702.08841.

12 Neil Immerman. *Descriptive complexity*. Graduate texts in computer science. Springer, 1999. doi:10.1007/978-1-4612-0539-5.

13 Peter T. Johnstone. *Stone spaces*, volume 3 of *Cambridge Studies in Advanced Mathematics*. Cambridge University Press, Cambridge, 1986. Reprint of the 1982 edition.

14 Andreas Krebs, Klaus-Jörn Lange, and Stephanie Reifferscheid. Characterizing TC^0 in Terms of Infinite Groups. In *STACS 2005, 22nd Annual Symposium on Theoretical Aspects of Computer Science, Stuttgart, Germany, February 24-26, 2005, Proceedings*, pages 496–507, 2005. doi:10.1007/978-3-540-31856-9_41.

15 Howard Straubing. *Finite Automata, Formal Logic, and Circuit Complexity*. Birkhäuser, Boston, Basel, Switzerland, 1994.

16 Howard Straubing. On logical descriptions of regular languages. In *LATIN 2002: Theoretical Informatics, 5th Latin American Symposium, Cancun, Mexico, April 3-6, 2002, Proceedings*, pages 528–538, 2002. doi:10.1007/3-540-45995-2_46.

17 Pascal Tesson and Denis Thérien. Logic meets algebra: the case of regular languages. *Logical Methods in Computer Science*, 3(1), 2007. doi:10.2168/LMCS-3(1:4)2007.

18 Ryan Williams. Nonuniform ACC circuit lower bounds. *J. ACM*, 61(1):2:1–2:32, 2014. doi:10.1145/2559903.

A Projective and direct limits

A *projective system* (also known as an inverse limit system, or a cofiltered diagram) $\mathcal{F}$ of sets assigns to each element i of a *directed* partially ordered set I, a set S_i, and to each ordered pair $i \geq j$ in I, a map $f_{i,j}\colon S_i \to S_j$ so that for all $i, j, k \in I$ with $i \geq j \geq k$ we have $f_{i,i} = id_{S_i}$ and $f_{j,k} \circ f_{i,j} = f_{i,k}$. The *projective limit* (or inverse limit or cofiltered limit) of $\mathcal{F}$, denoted $\varprojlim \mathcal{F}$, comes equipped with projection maps $\pi_i : \varprojlim \mathcal{F} \to S_i$ compatible with the system. That is, for $i, j \in I$ with $i \geq j$, $f_{i,j} \circ \pi_i = \pi_j$. Further, it satisfies the following *universal property*: whenever $\{\pi'_i : S' \to S_i\}_{i\in I}$ is a family of maps satisfying $f_{i,j} \circ \pi'_i = \pi'_j$ for all $i \geq j$, there exists a map $g : S' \to \varprojlim \mathcal{F}$ satisfying $\pi'_i = \pi_i \circ g$, for all $i \in I$.

There are several things worth noting about this notion. First, the projective limit of $\mathcal{F}$ may be constructed as follows:

$$\varprojlim \mathcal{F} = \left\{ (s_i)_{i\in I} \in \prod_{i\in I} S_i \mid f_{i,j}(s_i) = s_j \text{ whenever } i \geq j \right\}.$$

Second, projective limits of finite sets, called *profinite sets*, are equivalent to Boolean spaces. If each S_i is finite, then it is a Boolean space in the discrete topology, and the projective limit is a closed subspace of the product and thus again a Boolean space. Conversely, a Boolean space is the projective limit of the projective system of its finite continuous quotients.

Third, one also has projective systems and projective limits of richer structures than sets, such as algebras, topological spaces, maps between sets, etc. In these enriched settings the connecting maps are then required to be morphisms of the appropriate kind. A very useful fact, which is used throughout this work, is that in all these settings, the projective limits are given as for sets with the obvious enriched structure.

The notion dual to projective limit, obtained by reversing the directions of the maps, is that of *direct limit* (also known as an injective limit or inductive limit or filtered colimit), and it is denoted $\varinjlim$. It corresponds to the construction invoked in Definition 12.

B Appendix to Section 3

B.1 Proof of Proposition 7

We write $\mathsf{R} = (A, \mu, M, p, X)$ and we let $\mathsf{R}_\mathcal{B} = (A, \mu_\mathcal{B}, M_\mathcal{B}, p_\mathcal{B}, X_\mathcal{B})$ be the syntactic B-stamp of $\mathcal{B}$. It is clear that if $\mathsf{R}_\mathcal{B}$ factors through R, then every language recognised by $\mathsf{R}_\mathcal{B}$ is also recognised by R and, in particular, $\mathcal{B}$ is recognised by R. Conversely, suppose that $\mathcal{B}$ is recognised by R. Then, the kernel of μ is contained in the syntactic congruence $\sim_\mathcal{B}$, and therefore, the syntactic morphism $\mu_\mathcal{B}$ factors through μ, say $g \circ \mu = \mu_\mathcal{B}$. On the other hand, the dual of the embedding $\mathcal{B} \hookrightarrow \mathsf{Clopen}(X)$ yields a continuous quotient map φ form X to $X_\mathcal{B}$. Is is easy to check that the triple (id, g, φ) indeed defined a morphism from R to $\mathsf{R}_\mathcal{B}$. ◀

C Appendix to Section 4

C.1 Proof of Proposition 13

For a given map $\zeta : A \to B$ between finite alphabets, and a set $\Upsilon = \{x_1, \dots, x_k\}$ of variables, we define the map $\zeta \otimes \Upsilon : A^* \otimes \Upsilon \to B^* \otimes \Upsilon$ by $(\zeta \otimes \Upsilon)(w_{x_1=i_1,\dots,x_k=i_k}) = \zeta^*(w)_{x_1=i_1,\dots,x_k=i_k}$. Recall that, by Property (LC.3) of a logic class, ζ induces a map $\zeta_{\Gamma,\Upsilon} : \Gamma_B(\Upsilon) \to \Gamma_A(\Upsilon)$. We first observe the following:

▶ **Lemma 31.** *Let Γ be a logic class, Υ a finite set of variables, and $\zeta : A \to B$ a map between finite alphabets. Then, for every formula $\varphi \in \Gamma_B(\Upsilon)$ and Υ-structure $u \in A^* \otimes \Upsilon$, we have*

$$u \models \zeta_{\Gamma,\Upsilon}(\varphi) \iff (\zeta \otimes \Upsilon)(u) \models \varphi.$$

The proof is omitted as it follows straightforward from induction on the construction of a formula.

To prove Proposition 13 we shall also argue by induction on the construction of a formula. This requires a slight extension of the substitution map defined in Section 4.1.

▶ **Definition 32.** Let Γ and Λ be logic classes, Υ a finite set of variables, and x a variable that does not belong to Υ. Given a finite Boolean subalgebra $\mathcal{B} \subseteq \Lambda_A(\Upsilon \cup \{x\})$ and a finite set of variables Ψ disjoint from $\Upsilon \cup \{x\}$, the map $\sigma_{\mathcal{B},\Psi} : \Gamma_{C_\mathcal{B}}(\Psi) \to \mathcal{Q}_A[\mathcal{N}](\Psi \cup \Upsilon \cup \{x\})$ is the natural extension of the substitution map $\sigma_\mathcal{B}$.

Proof of Proposition 13. Properties (LC.1) and (LC.2) follow immediately from the fact that Γ and Λ are logic classes and from the definition of $\Gamma \circ \Lambda$. To prove (LC.3), we fix a map $\zeta : A \to B$ and, in order to simplify, we take $\Upsilon = \emptyset$. No additional difficulty arises from the general case. By definition of $\Gamma \circ \Lambda$, it suffices to show that, for every finite Boolean subalgebra $\mathcal{B} \subseteq \Lambda_B(x)$, the image of $\zeta_{\Gamma\circ\Lambda,\emptyset} \circ \sigma_\mathcal{B}$ is contained in $(\Gamma \circ \Lambda)_A(\emptyset)$. Since Λ is a logic class, the function ζ defines a morphism $\zeta_{\Lambda,\{x\}} : \Lambda_B(x) \to \Lambda_A(x)$. Let $\mathcal{B}'$ be the Boolean algebra generated by $\zeta_{\Lambda,\{x\}}(\mathcal{B})$. We claim that the set of atoms of $\mathcal{B}'$ is given by $\zeta_{\Lambda,\{x\}}(\mathsf{At}(\mathcal{B}))$. Indeed, this follows from the equivalence

$$w_{x=i} \models \zeta_{\Lambda,\{x\}}(\varphi) \iff \zeta(w)_{x=i} \models \varphi, \text{ for every } \varphi \in \mathcal{B} \text{ and } w \in A^*,$$

which is given by Lemma 31. Hence, the bijection $\zeta' : C_\mathcal{B} \to C_{\mathcal{B}'}$ induced by $\zeta_{\Lambda,\{x\}}$ defines a map $\zeta'_{\Gamma,\emptyset} : \Gamma_{C_\mathcal{B}}(\emptyset) \to \Gamma_{C_{\mathcal{B}'}}(\emptyset)$, and one may check the equality $\sigma_{\mathcal{B}'} \circ \zeta'_{\Gamma,\emptyset} = \zeta_{\Gamma\circ\Lambda,} \circ \sigma_\mathcal{B}$. This completes the proof. ◀

C.2 Proof of Proposition 14

Let $w \in A^*$, $u \in C_\mathcal{B}^*$, and $c \in C_\mathcal{B}$, and denote by $\varphi_c(x)$ the atom of $\mathcal{B}$ that the letter c stands for. Then the fact that ξ is dual to the inclusion means

$$w_{x=i} \vDash \varphi_c(x) \iff c = \xi(w_{x=i})$$

and by the definition of letter predicates we have

$$u_{x=i} \vDash \mathsf{P}_c(x) \iff u_i = c.$$

So, by definition of $\tau_\mathcal{B}$, for each $i \in \{1, \dots, |w|\}$ and each $c \in C_\mathcal{B}$, we have

$$w_{x=i} \vDash \varphi_c(x) \iff (\tau_\mathcal{B}(w))_{x=i} \vDash \mathsf{P}_c(x).$$

Since the validity of quantifiers is determined by the truth values of the formulas in their scope at all points of the model, and since $\psi \in \Gamma_{C_\mathcal{B}}(\emptyset)$ and $\tau_\mathcal{B}(\psi)$ are built up identically once the substitutions of $\mathsf{P}_c(x)$ by $\varphi_c(x)$ have been made, it follows that, for each $\psi \in \Gamma_{C_\mathcal{B}}(\emptyset)$, we have

$$\tau_\mathcal{B}(w) \in L_\psi \iff w \in L_{\sigma_\mathcal{B}(\psi)}. \tag{4}$$

However

$$w \in L_{\sigma_{\mathcal{B}}(\psi)} \quad \iff \quad L_{\sigma_{\mathcal{B}}(\psi)} \in p(w) \quad \iff \quad L_\psi \in \Sigma_{\mathcal{B}}(p(w))$$

so that

$$\tau_{\mathcal{B}}(w) \in L_\psi \quad \iff \quad L_\psi \in \Sigma_{\mathcal{B}}(p(w))$$

and thus $\Sigma_{\mathcal{B}}(p(w)) = q(\tau_{\mathcal{B}}(w))$ as required. ◀

D Appendix to Section 5

D.1 Proof of Proposition 20

It is enough to prove the claim for languages of the form

$$L = ((p \,\square\, q) \circ (\mu \,\square_q\, \nu))^{-1}(x, y),$$

for some $(x, y) \in X \times Y$. Note that every other language recognised by $\mathsf{R} \,\square\, \mathsf{S}$ is a Boolean combination of languages of this form. Let $w \in A^*$ be a word. By the definitions, w belongs to L if and only if

$$(p \circ \mu \circ \tau_{\mathsf{S}}(w), q \circ \nu(w)) = (x, y).$$

But this means that $L = \tau_{\mathsf{S}}^{-1}(K_1) \cap K_2$ where $K_1 = (p \circ \mu)^{-1}(x)$ and $K_2 = (q \circ \nu)^{-1}(y)$ are languages recognised, respectively, by R and by the *lp*-substamp of S determined by the injective map $A^* \hookrightarrow (A \times 2^{\{x\}})^*$ sending a to $(a, \emptyset)$. ◀

E Appendix to Section 6

E.1 Proof of Proposition 22

In Section 3.4 we defined the B-stamp $\Sigma_Y(\mathcal{V})$ to be the projective limit of a certain family of BiM presentations. We use the notation of that section in the rest of the proof, which we briefly recall here. If I is the set indexing the finite continuous quotients of Y, which are denoted $\pi_i : Y \twoheadrightarrow Y_i$, we say that $i \geq j$ provided there exists a quotient map $h_{i,j} : Y_i \twoheadrightarrow Y_j$ such that $h_{i,j} \circ \pi_i = \pi_j$. In particular, writing $\Sigma_{Y_i}(\mathcal{V}) = (Y_i, \mu_i, M_i, p_i, X_i)$, we have a connecting morphism $\Phi_{i,j} = (h_{i,j}, g_{i,j}, \varphi_{i,j})$ whenever $i \geq j$.

We start by defining the left action of N on X. Given $n \in N$, we let $\lambda_n : Y \to Y$ be the left action of N on Y (recall Definition 2). For each $i \in I$, the co-restriction to the image of the map $\pi_i \circ \lambda_n$ is a continuous quotient of Y. Thus, there exists an index $\varepsilon(i) \in I$ such that $\pi_{\varepsilon(i)}$ is the co-restriction of $\pi_i \circ \lambda_n$ to $Y_{\varepsilon(i)}$. In order to make more explicit at each step the set to which a certain element belongs, we denote by $h_i : Y_{\varepsilon(i)} \hookrightarrow Y_i$ the inclusion map. In particular, the equality $h_i \circ \pi_{\varepsilon(i)} = \pi_i \circ \lambda_n$ holds. Since $\mathcal{V}$ is a $\mathcal{C}$-pseudovariety, the map h_i yields a representation of $\Sigma_{Y_{\varepsilon(i)}}(\mathcal{V})$ as a B-substamp of $\Sigma_{Y_i}(\mathcal{V})$. We denote by $g_i : M_{\varepsilon(i)} \hookrightarrow M_i$ and $\varphi_i : X_{\varepsilon(i)} \hookrightarrow X_i$ the corresponding inclusion maps. In particular, we have the equality

$$\varphi_i \circ p_{\varepsilon(i)} \circ \mu_{\varepsilon(i)} = p_i \circ \mu_i \circ h_i^*. \tag{5}$$

On the other hand, whenever $i \geq j$, it is an easy observation that the codomain of the map $h_{i,j} \circ h_i$ is precisely $Y_{\varepsilon(j)}$. Therefore, we also have $\varepsilon(i) \geq \varepsilon(j)$. Note that, for $\kappa \in \{h, g, \varphi\}$, we have

$$\kappa_j \circ \kappa_{\varepsilon(i),\varepsilon(j)} = \kappa_{i,j} \circ \kappa_i. \tag{6}$$

To define the map $\psi_{\ell,n}$, we remark that X is the subspace of $\prod_{i\in I} X_i$ that consists of the tuples $(x_i)_{i\in I}$ satisfying $\varphi_{i,j}(x_i) = x_j$ for every $i \geq j$. Then, we set

$$\psi_{\ell,n}((x_i)_{i\in I}) = (\varphi_i(x_{\varepsilon(i)}))_{i\in I}. \tag{7}$$

This is a well-defined continuous map provided $\varphi_j(x_{\varepsilon(j)}) = \varphi_{i,j} \circ \varphi_i(x_{\varepsilon(i)})$, whenever $i \geq j$. But this follows immediately from the definition of X, together with (6) for $\kappa = \varphi$. More generally, we define $\psi_{r,n}$.

Finally, it is easy to see that setting $n \cdot x = \psi_{\ell,n}(x)$ and $x \cdot n = \psi_{r,n}(x)$, for $n \in N$ and $x \in X$, defines a biaction.

Now, given $(y_1, \ldots, y_k) \in Y^*$, we may compute:

$$\begin{aligned}
\psi_{\ell,n} \circ p \circ \mu(y_1, \ldots, y_k) &= \psi_{\ell,n}[(p_i \circ \mu_i \circ \pi_i^*(y_1, \ldots, y_k))_{i\in I}] && \text{by definition of } X \\
&= (\varphi_i \circ p_{\varepsilon(i)} \circ \mu_{\varepsilon(i)} \circ \pi_{\varepsilon(i)}^*(y_1, \ldots, y_k))_{i\in I} && \text{by (7)} \\
&= (p_i \circ \mu_i \circ h_i^* \circ \pi_{\varepsilon(i)}^*(y_1, \ldots, y_k))_{i\in I} && \text{by (5)} \\
&= (p_i \circ \mu_i \circ \pi_i^* \circ \lambda_n^*(y_1, \ldots, y_k))_{i\in I} && \text{because } \pi_i \circ \lambda_n = h_i \circ \pi_{\varepsilon(i)}.
\end{aligned}$$

This computation proves (1). Equality (2) is derived similarly. ◀

F Appendix to Section 7

F.1 Proof of Proposition 25

It suffices to consider the case where we are given a sentence of the form

$$\varphi = (\mathsf{Q}x \bigvee_{a\in B} \mathsf{P}_a(x)),$$

for a certain subset $B \subseteq A$. Let $h : A^* \to \{0,1\}^*$ be the unique homomorphism that sends the letter a to 1 if and only if a belongs to B. Then, by the way quantifiers are interpreted, we have that the models of φ are precisely the elements of $(\mathsf{Q} \circ h)^{-1}(1)$ and thus, L_φ is recognised by the typed lp-substamp of R_Q defined by h. ◀

F.2 Proof of Proposition 27

We illustrate the proof in the case R is a unary predicate. We first observe that, for every $\ell, \ell_1, \ell_2 \in \mathbb{N}$, the equalities

$$\alpha_\ell \cdot 0 = \alpha_\ell = 0 \cdot \alpha_\ell; \quad \alpha_\ell \cdot 1 = \alpha_0 = 1 \cdot \alpha_\ell; \quad \textit{and} \quad \alpha_{\ell_1} + \alpha_{\ell_2} = \alpha_{\ell_1+\ell_2};$$

hold. Given a word $u = (a_1, S_1) \cdots (a_m, S_m) \in (A \times 2^{\{x\}})^*$ with at least one marked position, we set

$$i_{\min} = \min\{j \mid S_j \neq \emptyset\} \quad \text{and} \quad i_{\max} = \max\{j \mid S_j \neq \emptyset\}.$$

Then, one may compute

$$\nu_R(u) = (m, (\alpha_{i_{\min}}, 1), (\alpha_{m-i_{\max}+1}, 1))$$

Thus, u belongs to $A^* \otimes \{x\}$ if and only if $i_{\min} = m - i_{\max} + 1$, and in that case the unique marked position of w is $i_{\min}$. This means that u is a model of R if and only if $p_R \circ \nu_R(u) = 1$. On the other hand, if w is a word over A, then we have

$$\nu_R(w) = (\alpha_\ell, 0),$$

for some $\ell \in \mathbb{N}$. Thus, $p_R \circ \nu_R(w) = 0$. This proves that $L_R = (p_R \circ \nu_R)^{-1}(1)$ as intended. ◀