

1 Characterizing NC^1 with Typed Semigroups

2 **Anonymous author**

3 Anonymous affiliation

4 **Anonymous author**

5 Anonymous affiliation

6 — **Abstract** —

7 *[TODO]:*

8 **2012 ACM Subject Classification** Replace ccsdesc macro with valid one

9 **Keywords and phrases** Dummy keyword

10 **Digital Object Identifier** 10.4230/LIPIcs.CVIT.2016.23

11 **Acknowledgements** Anonymous acknowledgements



© Anonymous author(s);

licensed under Creative Commons License CC-BY 4.0

42nd Conference on Very Important Topics (CVIT 2016).

Editors: John Q. Open and Joan R. Access; Article No. 23; pp. 23:1–23:16

Leibniz International Proceedings in Informatics



LIPICs Schloss Dagstuhl – Leibniz-Zentrum für Informatik, Dagstuhl Publishing, Germany

12 **1** Introduction

13 Much work in theoretical computer science is concerned with studying classes of formal
 14 languages, whether these are classes defined in terms of grammars and expressions, such as
 15 the class of regular or context-free languages, or whether they are *complexity classes* such as
 16 P and NP, defined by resource bounds on machine models. Indeed, the distinction between
 17 these are largely historical as most classes of interest admit different characterizations based
 18 on machine models, grammars, logical definability, or algebraic expressions. The class of
 19 regular languages can be characterized as the languages accepted by linear-time-bounded
 20 single-tape Turing machines [8] while P can be characterized without reference to resources
 21 as the languages recognized by multi-head two-way pushdown automata [5]. The advantage
 22 of the variety of characterizations is, of course, the fact that these bring with them different
 23 mathematical toolkits that can be brought to the study of the classes.

24 The class of regular languages has arguably the richest theory in this sense of diversity
 25 of characterizations. Virtually all students of computer science learn of the equivalence of
 26 deterministic and nondeterministic finite automata, regular languages and linear grammars
 27 and many also know that the regular languages are exactly those definable in monadic
 28 second-order logic with an order predicate. Perhaps the most productive approach to the
 29 study of regular languages is via their connection to finite semigroups. Every language L
 30 has a syntactic semigroup, which is finite if, and only if, L is regular. Moreover, closure
 31 properties of classes of regular languages relate to natural closure properties of classes of
 32 semigroups, via Eilenberg's Correspondence Theorem [7]. This, together with the tools of
 33 *Krohn-Rhodes theory*, gives rise to *algebraic automata theory*—which leads to the definition
 34 of natural subclasses of the class of regular languages, to effective decision procedures for
 35 automata recognizing such classes, and to separation results.

36 When it comes to studying computational complexity, we are mainly interested in classes
 37 of languages richer than just the regular languages. Thus the syntactic semigroups of the
 38 languages are not necessarily finite any longer and the extensive tools of Krohn-Rhodes
 39 theory are not available to study them. Nonetheless, some attempts have been made to
 40 extend the methods of algebraic automata theory to classes beyond the regular languages.
 41 Most significant is the work of Krebs and collaborators [2, 3, 10, 9, 4], which introduces
 42 the notion of *typed semigroups*. The idea is to allow for languages with infinite syntactic
 43 semigroups, but limit the languages they recognize by associating with them a finite collection
 44 of types. This allows for the formulation of a version of Eilenberg's Correspondence theorem
 45 associating closure properties on classes of typed semigroups with corresponding closure
 46 properties of classes of languages. In particular, this implies that most complexity classes of
 47 interest can be uniquely characterized in terms of an associated class of typed semigroups [3].
 48 An explicit description of the class characterizing $\text{DLOGTIME-uniform TC}^0$ is given in [10, 9].
 49 This is obtained through a general method which allows us to construct typed semigroups
 50 corresponding to *unary quantifiers* defined from specific languages [9] (see also Theorem 22
 51 below).

52 In this paper, we extend this work to obtain a characterization of DLOGTIME-uniform
 53 NC^1 as the class of languages recognized by the collection of typed semigroups obtained as the
 54 closure under *ordered strong block products* of three typed semigroups: the group of integers
 55 with types for positive and negative integers; the group of natural numbers with types for
 56 the square numbers and non-square numbers; and a finite non-solvable group such as S_5 with
 57 a type for each subset of the group. Full definitions of these terms follow below. Our result
 58 is obtained by first characterizing $\text{DLOGTIME-uniform NC}^1$ in terms of logical definability

in an extension of first-order logic with only unary quantifiers. It is known that any regular language whose syntactic semigroup is a non-solvable groups is complete for NC^1 under reductions definable in first-order logic with arithmetic predicates ($\text{FO}(+, \times)$) [1]. From this, we know we can describe NC^1 as the class of languages definable in an extension of $\text{FO}(+, \times)$ with quantifiers (of arbitrary arity) associated with the regular language corresponding to the word problem for S_5 . Our main technical contribution is to show that the family of such quantifiers associated with any regular language L can be replaced with just the unary quantifiers. This also answers a question left open in [12].

In Section 2, we cover the relevant background material on semigroup theory, typed semigroups, and multiplication quantifiers. In Section 3, we establish the main technical result showing that quantifiers of higher arity over a regular language L can be defined using just unary quantifiers over the syntactic semigroup of L . Finally, in Section 4, we apply this to obtain the algebraic characterization of $\text{DLOGTIME-uniform NC}^1$.

2 Preliminaries

We assume the reader is familiar with basic concepts of formal language theory, automata theory, complexity theory, and logic. We quickly review definitions we need to fix notation and establish conventions.

We write \mathbb{Z} for the set of integers, \mathbb{N} for the set of natural numbers (including 0), and \mathbb{Z}^+ for the set of positive integers. We write $[n]$ for the set of integers $\{1, \dots, n\}$ and \mathbb{S} for the set of *square* integers. That is, $\mathbb{S} = \{x \in \mathbb{Z}^+ \mid x = y^2 \text{ for some } y \in \mathbb{Z}\}$.

For a fixed $n \in \mathbb{Z}^+$ and an integer $i \in [n]$, we define the *n-bit one-hot encoding* of i to be the binary string $b \in \{0, 1\}^n$ such that $b_j = 1$ if, and only if, $j = i$.

2.1 Semigroups, Monoids and Groups

A *semigroup* (S, \cdot) is a set S equipped with an *associative* binary operation. We call a semigroup *finite* if S is finite. Context permitting, we may refer to a semigroup (S, \cdot) simply by its underlying set S . A *monoid* (M, \cdot) is a semigroup with a distinguished element $1 \in M$ such that for all $m \in M$, $1 \cdot m = m \cdot 1 = m$. We call 1 the *identity* or *neutral* element of M . A *group* (G, \cdot) is a monoid such that for every $g \in G$, there exists an element $g^{-1} \in G$ such that $g \cdot g^{-1} = g^{-1} \cdot g = 1$. We call g^{-1} the *inverse* of g .

Note that $(\mathbb{Z}, +)$ is a group, $(\mathbb{N}, +)$ is a monoid but not a group and $(\mathbb{Z}^+, +)$ is a semigroup but not a monoid. In the first two cases, the identity element is 0. When we refer to the semigroups \mathbb{Z} or \mathbb{N} we assume that the operation referred to is standard addition.

For a semigroup (S, \cdot) , we say that a set $G \subseteq S$ *generates* S if S is equal to the closure of G under \cdot ; we denote this by $S = \langle G \rangle$, or, simply, $\langle G \rangle$ if the operation is clear from context, and call G a *generating set* of S . We say that S is *finitely generated* if there exists a finite generating set of S . All semigroups we consider are finitely generated. Note that \mathbb{Z}^+ is generated by $\{1\}$, \mathbb{N} by $\{0, 1\}$ and \mathbb{Z} by $\{1, -1\}$.

We write U_1 for the monoid $(\{0, 1\}, \cdot)$ where the binary operation is the standard multiplication. Note that 1 is the identity element here. For any set S , we denote by S^+ the set of non-empty finite strings over S and by S^* the set of all finite strings over S . Equipped with the concatenation operation on strings, which we denote by either \circ or simply juxtaposition, S^* is a monoid and S^+ is a semigroup but not a monoid. We refer to these as the *free monoid* and *free semigroup* over S , respectively. Note that S is a set of generators for S^+ and $S \cup \{\epsilon\}$ is a set of generators for S^* .

A homomorphism from a semigroup (S, \cdot_S) to a semigroup (T, \cdot_T) is a function $h : S \rightarrow T$ such that for all $s_1, s_2 \in S$, $h(s_1 \cdot_S s_2) = h(s_1) \cdot_T h(s_2)$. A *congruence* on a semigroup (S, \cdot) is an equivalence relation \sim on S such that for all $a, b, c, d \in S$, if $a \sim b$ and $c \sim d$, then $a \cdot c \sim b \cdot d$. We denote by S/\sim the set of equivalence classes of \sim on S . We denote by $[a]_\sim$, or simply $[a]$, the equivalence class of $a \in S$ under \sim . Any congruence \sim gives rise to the *quotient semigroup* of S by \sim , namely the semigroup $(S/\sim, \star)$ where for $[a], [b] \in S/\sim$, $[a] \star [b] = [a \cdot b]$. The map $\eta : S \rightarrow S/\sim$ defined by $\eta(a) = [a]$ is then a homomorphism, known as the *canonical homomorphism* of S onto S/\sim .

For future reference, we formally define the syntactic congruence associated with a language L .

► **Definition 1.** Let L be a language. We define the syntactic congruence of L as the equivalence relation \sim_L on Σ^+ such that for all $x, y \in \Sigma^+$, $x \sim_L y$ if and only if for all $w, v \in \Sigma^+$, $wxy \in L$ iff $wyv \in L$.

It is easily seen that this relation is a congruence on the free monoid Σ^* . The quotient semigroup Σ^*/\sim_L is known as the *syntactic semigroup* of L . More generally, we say that a semigroup S *recognizes* the language L if there is a homomorphism $h : \Sigma^+ \rightarrow S$ and a set $A \subseteq S$ such that $L = h^{-1}(A)$. It is easily seen that the syntactic semigroup of L recognizes L . A language is regular if, and only if, its syntactic semigroup is finite.

2.2 Logics and Quantifiers

We assume familiarity with the basic syntax and semantics of first-order logic. In this paper, the logic is always interpreted in finite relational structures. We generally denote structures by Fraktur letters, \mathfrak{A} , \mathfrak{B} , etc., and the corresponding universe of the structure is denoted $|\mathfrak{A}|$, $|\mathfrak{B}|$, etc. We are almost exclusively interested in *strings* over a finite alphabet. Thus, fix an alphabet Σ . A Σ -string is then a structure \mathfrak{A} whose universe A is linearly ordered by a binary relation $<$ and which interprets a set of unary relation symbols $(R_\sigma)_{\sigma \in \Sigma}$. For each element $a \in |\mathfrak{A}|$ there is a unique $\sigma \in \Sigma$ such that a is in the interpretation of R_σ .

More generally, let τ be any relational vocabulary consisting of a binary relation symbol $<$ and unary relation symbols R_1, \dots, R_k . We can associate with any τ -structure in which $<$ is a linear order a string over an alphabet of size 2^k as formalized in the following definition.

► **Definition 2.** For τ a relational vocabulary consisting of a binary relation symbol $<$ and unary relation symbols R_1, \dots, R_k , and \mathfrak{A} a τ -structure with n elements that interprets the symbol $<$ as a linear order of its universe, we define the string $w_{\mathfrak{A}}$ associated with \mathfrak{A} as the string of length n over the alphabet $\Sigma_k = \{0, 1\}^k$ of size 2^k so that if a is the i th element of $w_{\mathfrak{A}}$, then a is the k -tuple where $a_j = 1$ if, and only if, R_j holds at the i th element of \mathfrak{A} .

This way, we can associate a language with any isomorphism-closed class of structures over the vocabulary τ . We formalise this definition for future use.

► **Definition 3.** For τ a relational vocabulary consisting of a binary relation symbol $<$ and unary relation symbols R_1, \dots, R_k , and \mathcal{A} a class of τ -structures, we define the language $L_{\mathcal{A}}$ over the alphabet $\Sigma_k = \{0, 1\}^k$ to be

$$L_{\mathcal{A}} = \{w_{\mathfrak{A}} \mid \mathfrak{A} \in \mathcal{A}\}.$$

Conversely, for any language L over the alphabet Σ_k , we define the class of τ -structures \mathcal{S}_L to be

$$\mathcal{S}_L = \{\mathfrak{A} \mid w_{\mathfrak{A}} \in L\}.$$

As the elements of a string \mathfrak{A} are linearly ordered, we can identify them with an initial segment $\{1, \dots, n\}$ of the positive integers. In other words, we treat a string with universe $\{1, \dots, n\}$ and the standard order on these elements as a canonical representative of its isomorphism class. In addition to the order predicate, we may allow other *numerical predicates* to appear in formulas of our logics. These are predicates whose meaning is completely determined by the size n of the structure and the ordering of its elements. In particular, we have ternary predicates $+$ and \times for the partial addition and multiplication functions.

An insight due to Lindström allows us to define a *quantifier* from any isomorphism-closed class of structures (see [6]). Specifically, let Q be any isomorphism-closed class of structures in a relational vocabulary $\tau = \langle R_1, \dots, R_l \rangle$, where for each i , R_i is a relation symbol of arity r_i . For any vocabulary σ and positive integer d , an *interpretation* of τ in σ of dimension d is a tuple of formulas $I = (\phi_1(\bar{x}_1), \dots, \phi_l(\bar{x}_l))$ of vocabulary σ where ϕ_i is associated with a tuple \bar{x}_i of variables of length dr_i . Suppose we are given a σ -structure \mathfrak{A} and an assignment α that takes variables to elements of \mathfrak{A} . Then let $\phi_i^{\mathfrak{A}, \alpha}$ denote the relation of arity dr_i consisting of the set of tuples $\{\bar{a} \in |\mathfrak{A}|^{dr_i} \mid \mathfrak{A} \models \phi_i[\alpha[\bar{x}_i/\bar{a}]]\}$. Then, the interpretation I defines a map that takes a σ -structure \mathfrak{A} , along with an assignment α to the τ -structure $I(\mathfrak{A}, \alpha)$ with universe $|\mathfrak{A}|^d$ where the interpretation of R_i is the set $\phi_i^{\mathfrak{A}, \alpha}$, seen as a relation of arity r_i on $|\mathfrak{A}|^d$.

Then, in a logic with quantifier Q , we can form formulas of the form

$$Q\bar{x}_1 \cdots \bar{x}_l(\phi_1, \dots, \phi_l)$$

in which occurrences in the the subformula ϕ_i of variables among x_i are bound. The semantics of this quantifier are given by the rule that $Q\bar{x}_1 \cdots \bar{x}_l(\phi_1, \dots, \phi_l)$ is true in a structure \mathfrak{A} under some interpretation α of values to the free variables if the τ -structure $I(\mathfrak{A}, \alpha)$ is in Q . Note, we have defined what are usually called *vectorized quantifiers*, in that they can take interpretations of any dimension. Another way of formulating this is to have a separate quantifier Q_d for each dimension d . We switch between these notations when it causes no confusion and we call Q_d the *vectorization* of Q of dimension d .

We are particularly interested in interpretations I where both σ and τ are vocabularies of strings. These are also known in the literature as *string-to-string transducers* [?]. We further restrict ourselves to interpretations in which the definition of the linear order in $I(\mathfrak{A}, \alpha)$ is always the lexicographic order on d -tuples of \mathfrak{A} induced by the order in \mathfrak{A} . This order is easily defined by a (quantifier-free) first-order formula, and we simply omit it from the description of I . Hence, we only need to specify the interpretation giving the unary relations in τ and an interpretation of dimension d has the simple form $(\phi_1(\bar{x}_1), \dots, \phi_l(\bar{x}_l))$, where all tuples of variables have length d . We can then assume, without loss of generality, that they are all the same tuple \bar{x} and we thus write a formula with a string quantifier Q as

$$Q\bar{x}(\phi_1, \dots, \phi_l).$$

Observe that a quantifier applied to an interpretation of dimension d will then bind d variables.

We say that an interpretation is *unary* if it has dimension 1. We now introduce some notation we use in the rest of the paper for various logics formed by combining particular choices of quantifiers and numerical predicates.

► **Definition 4.** For a set of quantifiers \mathfrak{Q} and numerical predicates \mathfrak{N} , we denote by $(\mathfrak{Q})[\mathfrak{N}]$ the logic constructed by extending quantifier-free first-order logic with the quantifiers in \mathfrak{Q} and allowing the numerical predicates in \mathfrak{N} .

We denote by FO the set of standard first-order quantifiers: $\{\exists, \forall\}$.

For a singleton set of quantifiers $\mathfrak{Q} = \{Q\}$, we sometimes denote $(\mathfrak{Q})[\mathfrak{N}]$ as $(Q)[\mathfrak{N}]$. We use similar notation for the sets of numerical predicates. We use $\mathcal{L}((\mathfrak{Q})[\mathfrak{N}])$ to denote the languages expressible by the logic $(\mathfrak{Q})[\mathfrak{N}]$. We also use $(\mathfrak{Q}_1)[\mathfrak{N}]$ to denote the logic obtained as a restriction of $(\mathfrak{Q})[\mathfrak{N}]$ to formulas in which quantifiers in \mathfrak{Q} are only applied to interpretations of dimension 1.

All the logics we consider are *substitution closed* in the sense of [6]. This means in particular that if a quantifier Q is definable in a logic $(\mathfrak{Q})[\mathfrak{N}]$, then extending the logic with the quantifier Q does not add to its expressive power. This is because we can replace occurrences of the quantifier Q by its definition, with a suitable substitution of the interpretation for the relation symbols. Hence, if Q is definable in $(\mathfrak{Q})[\mathfrak{N}]$, then $\mathcal{L}((\mathfrak{Q})[\mathfrak{N}]) = \mathcal{L}((\mathfrak{Q} \cup \{Q\})[\mathfrak{N}])$.

A remark is due on our notation for numerical predicates. All structures we consider are ordered, including those defining the quantifiers. Thus the order predicate is implicitly present in the collection of numerical predicates \mathfrak{N} and is used (implicitly) to define the interpretations to a quantifier. We sometimes write $(\mathfrak{Q})[\emptyset]$ to indicate a logic in which this is the only use of the order that is allowed. By our choice of notation, the order symbol then does not appear explicitly in the syntax of the formulas.

2.3 Multiplication Quantifiers

The definition of multiplication quantifier has its origin in Barrington, Immerman, and Straubing [1, Section 5] where they were referred to as monoid quantifiers; the authors proved that the languages in DLOGTIME-uniform NC¹ are exactly those expressible by first-order logic with quantifiers whose truth-value is determined via multiplication in a finite semigroup. The notion was extended by Lautemann et al. [12] to include quantifiers for the word problem over more general algebras with a binary operation. Multiplication quantifiers over a finite semigroup S can be understood as generalized quantifiers corresponding to languages recognized by S , and here we define them as such.

Fix a semigroup S , a set $B \subseteq S$ and a positive integer k . Let Σ_k denote the set $\{0, 1\}^k$ which we think of as an alphabet of size 2^k , and fix a function $\gamma : \Sigma_k \rightarrow S$. We extend γ to strings in Σ_k^+ inductively in the standard way: $\gamma(wa) = \gamma(w)\gamma(a)$. Together these define a language

$$L_{\gamma}^{S,B} = \{x \in \Sigma_k^* \mid \gamma(x) \in B\}.$$

We can now define a *multiplication quantifier*. In the following, \mathcal{S}_L denotes the class of structures associated with a language L in the sense of Definition 3.

► **Definition 5.** Let τ be a vocabulary including an order symbol $<$ and k unary relations. For a semigroup S , a set $B \subseteq S$, a positive integer k and a function $\gamma : \{0, 1\}^k \rightarrow S$, the multiplication quantifier $\Gamma_{\gamma}^{S,B}$ is the Lindström quantifier associated with the class of structures $\mathcal{S}_{L_{\gamma}^{S,B}}$.

We also write $\Gamma_{d,\gamma}^{S,B}$ for the vectorization of this quantifier of dimension d . If B is a singleton $\{s\}$, then we often write $\Gamma_{d,\gamma}^{S,s}$ for short.

Recall that U_1 denotes the two-element semigroup $\{0, 1\}$ with standard multiplication. Then, it is easily seen that $\Gamma_{1,\gamma}^{U_1,0}$, where $\gamma : \{0, 1\} \rightarrow U_1$ such that $\gamma(0) = 1$ and $\gamma(1) = 0$, is the standard existential quantifier. The universal quantifier can be defined similarly.

► **Definition 6.** For a semigroup S , we define the following collections of quantifiers:

$$\Gamma^S = \left\{ \Gamma_{l,\gamma}^{S,B} \mid B \subseteq S, \gamma : \{0, 1\}^k \rightarrow S, \text{ and } l, k \geq 1 \right\}$$

$$\Gamma_l^S = \left\{ \Gamma_{l,\gamma}^{S,B} \mid B \subseteq S \text{ and } \gamma : \{0,1\}^k \rightarrow S \right\}$$

$$\Gamma_{l,\gamma}^S = \left\{ \Gamma_{l,\gamma}^{S,B} \mid B \subseteq S \right\}$$

Finally, let Γ^{fin} be the collection of all multiplication quantifiers over finite semigroups.

From [1, Corollary 9.1], we know that DLOGTIME-uniform NC^1 is characterized by $(\text{FO})[+, \times]$ equipped with finite multiplication quantifiers:

► **Theorem 7 ([1]).** $\text{DLOGTIME-uniform NC}^1 = \mathcal{L}((\Gamma^{\text{fin}})[+, \times])$.

► **Remark 8.** In fact, simply allowing multiplication quantifiers for some fixed finite, non-solvable monoid suffices. The definition of “non-solvable monoid” is not needed for our proofs here but, for example, the *symmetric group of degree five*, denoted S_5 , is a non-solvable monoid. Therefore, we know that $\text{DLOGTIME-uniform NC}^1 = \mathcal{L}((\text{FO} \cup \Gamma^{S_5})[+, \times])$.

In the absence of the arithmetic predicates for addition and multiplication, the logic of multiplication quantifiers over finite semigroups only allows us to define regular languages. Specifically, Barrington et al. [1] established that the regular languages are characterized by the logic using such quantifiers with only unary interpretations. We denote this logic $(\Gamma_1^{\text{fin}})[<]$.

► **Theorem 9 ([1]).** $\text{REG} = \mathcal{L}((\Gamma_1^{\text{fin}})[<])$.

Later, Lautemann et al. [12, Theorem 5.1] showed that allowing interpretations of higher dimension to the quantifiers does not increase the expressive power when order is the only numerical predicate.

► **Theorem 10.** $\text{REG} = \mathcal{L}((\Gamma^{\text{fin}})[<])$.

Our main technical result shows that this is true even in the presence of any numerical predicates and therefore Γ^{fin} can be replaced by Γ_1^{fin} even in Theorem 7.

2.4 Typed Semigroups

In this subsection, we review the definitions and results from [?, ?] on typed semigroups, their relationship to languages and corresponding characterizations of complexity classes.

A typed semigroup is a semigroup equipped with a collection of *types*, which form a Boolean algebra, and a set of *units*. We only deal with concrete Boolean algebras, given as collections of subsets of a fixed universe.

► **Definition 11 (Boolean Algebra).** A Boolean algebra over a set S is a set $B \subseteq \wp(S)$ such that $\emptyset, S \in B$ and B is closed under union, intersection, and complementation. If B is finite, we call it a finite Boolean algebra.

We call \emptyset and S the trivial elements (or in some contexts, the trivial types) of B .

A homomorphism between Boolean algebras is defined as standard. That is, if B_1 and B_2 are Boolean algebras over sets S and T , respectively, then we call $h : B_1 \rightarrow B_2$ a homomorphism if $h(\emptyset) = \emptyset$, $h(S) = T$, and for all $s_1, s_2 \in B_1$, $h(s_1 \cap s_2) = h(s_1) \cap h(s_2)$, $h(s_1 \cup s_2) = h(s_1) \cup h(s_2)$, and $h(s^C) = (h(s))^C$. Now we are ready to define typed semigroups.

► **Definition 12 (Typed Semigroup).** Let S be a semigroup, G a Boolean algebra over S , and E a finite subset of S . We call the tuple $T = (S, G, E)$ a typed semigroup over S and the

elements of G types and the elements of E units. We call S the base semigroup of T . If S is a monoid or group, then we may also call T a typed monoid or typed group, respectively.

If $G = \{\emptyset, A, S - A, S\}$, then we often abbreviate T as (S, A, E) , i.e., the Boolean algebra is signified by an element, or elements, which generates it—in this case, A .

We say that a typed semigroup (S, G, E) is finite if S is.

We also need a notion of morphism between typed semigroups.

► **Definition 13.** A typed homomorphism $h : (S, G, E) \rightarrow (T, H, F)$ of typed semigroups is a triple (h_1, h_2, h_3) where $h_1 : S \rightarrow T$ is a semigroup homomorphism, $h_2 : G \rightarrow H$ is a homomorphism of Boolean algebras, and $h_3 : E \rightarrow F$ is a mapping of sets such that the following conditions hold:

(i) For all $A \in G$, $h_1(A) = h_2(A) \cap h_1(S)$.

(ii) For all $e \in E$, $h_1(e) = h_3(e)$.

Note that h_3 is redundant in the definition as it is completely determined by h_1 . We retain it as part of the definition for consistency with [?].

To motivate the definitions, recall that a language $L \subseteq \Sigma^*$ is recognized by a semigroup S if there is a homomorphism $h : \Sigma^* \rightarrow S$ and a set $B \subseteq S$ such that $L = h^{-1}(B)$. When the semigroup S is infinite, the languages recognized form a rather rich collection and we aim to restrict this in two ways. First, B cannot be an arbitrary set but must be an element of the algebra of types. Secondly, the homomorphism h must map the letters in Σ to units of the typed semigroup. Formally, we have the following definition.

► **Definition 14.** A typed semigroup $T = (S, G, E)$ recognizes a language $L \subseteq \Sigma^+$ if there exists a typed homomorphism from (Σ^+, L, Σ) to T . We let $\mathcal{L}(T)$ denote the set of languages recognized by T .

When the base semigroup of a typed semigroup is finite, we recover the classical definition of a recognition. Hence, the languages recognized by finite typed semigroup are necessarily regular.

► **Proposition 15.** If T is a finite typed semigroup, then $\mathcal{L}(T) \subseteq \text{REG}$.

We can now state the definitions of the key relationships between typed semigroups.

► **Definition 16.** Let (S, G, E) and (T, H, F) be typed semigroups.

■ A typed homomorphism $h = (h_1, h_2, h_3) : (S, G, E) \rightarrow (T, H, F)$ is injective (surjective, or bijective) if all of h_1 , h_2 , and h_3 are.

■ (S, G, E) is a typed subsemigroup (or, simply, “subsemigroup” when context is obvious) of (T, H, F) , denoted $(S, G, E) \leq (T, H, F)$, if there exists an injective typed homomorphism $h : (S, G, E) \rightarrow (T, H, F)$.

■ (S, G, E) divides (T, H, F) , denoted $(S, G, E) \preceq (T, H, F)$, if there exists a surjective typed homomorphism from a typed subsemigroup of (T, H, F) to (S, G, E) .

These have the expected properties.

► **Proposition 17 ([3]).** Let T_1 , T_2 , and T_3 be typed semigroup.

■ Typed homomorphisms are closed under composition.

■ Division is transitive: if $T_1 \preceq T_2$ and $T_2 \preceq T_3$, then $T_1 \preceq T_3$.

■ If $T_1 \preceq T_2$, then $\mathcal{L}(T_1) \subseteq \mathcal{L}(T_2)$.

We can formulate the notion of the *syntactic typed semigroup* of a language L as an extension of the syntactic semigroup of L with a minimal collection of types and units necessary.

293 ► **Definition 18.** Let $T = (S, G, E)$ be a typed semigroup. A congruence \sim over S is a typed
 294 congruence over T if for every $A \in G$ and $s_1, s_2 \in S$, if $s_1 \sim s_2$ and $s_1 \in A$, then $s_2 \in A$.

295 For a typed congruence \sim over T , let

$$296 \quad S'/\sim = \{[x]_\sim \mid x \in S'\} \text{ where } S' \subseteq S$$

$$297 \quad G/\sim = \{A/\sim \mid A \in G\}$$

$$298 \quad E/\sim = \{[x]_\sim \mid x \in E\}.$$

299 Then, $T/\sim := (S/\sim, G/\sim, E/\sim)$ is the typed quotient semigroup of T by \sim .

300 Let \sim_T denote the typed congruence on T such that for $s_1, s_2 \in S$, $s_1 \sim_T s_2$ iff for all
 301 $x, y \in S$ and $A \in G$, $xs_1y \in A$ iff $xs_2y \in A$. We then refer to the quotient semigroup T/\sim_T
 302 as the minimal reduced semigroup of T .

303 Recall that \sim_L is the syntactic congruence of L , defined in Definition 1.

304 ► **Definition 19.** For a language $L \subseteq \Sigma^+$, the syntactic typed semigroup of L , denoted
 305 $\text{syn}(L)$, is the typed semigroup $(\Sigma^+, L, \Sigma)/\sim_L$.

306 We also get the canonical typed homomorphism, $\eta_d : (\Sigma^+, L, \Sigma) \rightarrow \text{syn}(L)$ induced by the
 307 syntactic homomorphism of L .

308 We now turn to the relationship between the expressive power of logics with multiplication
 309 quantifiers and typed semigroups. A formal association is defined through the definition
 310 below.

311 ► **Definition 20.** For a multiplication quantifier $Q = \Gamma_\gamma^{S,B}$ where $\gamma : \{0,1\}^k \rightarrow S$, we define
 312 the typed quantifier semigroup of Q , denoted $\mathcal{S}(Q)$, to be the syntactic typed semigroup of
 313 the language $L_\gamma^{S,B}$.

314 It turns out that we can give a purely structural characterization of those typed semigroups
 315 that are syntactic semigroups.

316 ► **Proposition 21 ([9]).** A typed semigroup is the syntactic semigroup of a language if, and
 317 only if, it is reduced, generated by its units, and has four or two types.

318 In case it has just two types, then it only recognizes the empty language or the language
 319 of all strings.

320 We now want to state the formal connection between the languages expressible in a
 321 logic with a collection of quantifiers and the corresponding class of typed semigroups. For this
 322 we need the notion of the *ordered strong block product closure* of a set of typed semigroups
 323 T , which we denote $\text{sbpc}_<(T)$. The definition is technical and can be found in [9] and is also
 324 reproduced in the appendix for ease of reference.

325 From [9, Theorem 4.14], we then get the following relationship between logics and
 326 algebras:¹

327 ► **Theorem 22.** Let \mathcal{Q} be a collection of quantifiers and Q the corresponding set of typed
 328 quantifier semigroups for \mathcal{Q} . Then, $\mathcal{L}((\mathcal{Q}_1)[<]) = \mathcal{L}(\text{sbpc}_<(Q))$.

¹ The theorem in [9] is actually more general as it accounts for more predicates than just order; however, for our purposes, order alone suffices.

3 Simplifying Multiplication Quantifiers

To use Theorem 22 to obtain an algebraic characterization of NC^1 , we need to characterize this class in a logic with only unary quantifiers, i.e. where quantifiers are only applied to unary interpretations. Remark 8 gives us a characterization using first-order quantifiers and Γ^{S^5} . Our aim in this section is to show that we can eliminate the use of interpretations of dimension higher than 1 in this logic. As a first step, we show that we can restrict ourselves to quantifiers $\Gamma_\delta^{S^5, s}$ for a *fixed* function δ .

► **Lemma 23.** *For every finite semigroup S , there exists a function $\delta : \{0, 1\}^{|S|} \rightarrow S$ such that for every $s \in S$ and $\gamma : \{0, 1\}^k \rightarrow S$, the quantifier $\Gamma_\gamma^{S, s}$ is definable in $(\Gamma_\delta^{S, s})[\emptyset]$.*

Proof. Recall that $\Gamma_\gamma^{S, s}$ is the class of structures \mathfrak{A} in a vocabulary τ with one binary relation $<$ and k unary relations R_1, \dots, R_k such that $\gamma(w_\mathfrak{A}) = s$ where $w_\mathfrak{A}$ is the string associated with \mathfrak{A} as in Def. 2.

Let $c = |S|$, fix an enumeration $\{s_1, \dots, s_c\}$ of S , and let z be an arbitrary element of S . Let $\delta : \{0, 1\}^c \rightarrow S$ be the function where $\delta(w) = s_i$ if w is the one-hot encoding of i and $\delta(w) = z$ otherwise (that is, if the number of occurrences of the symbol 1 in the string w is not exactly one).

For each $t \in S$, define the formula $\psi_t(x)$ as follows:

$$\psi_t(x) := \bigvee_{w \in \{0, 1\}^k : \gamma(w) = t} \left(\bigwedge_{i \in [k] : w_i = 1} R_i(x) \wedge \bigwedge_{i \in [k] : w_i = 0} \neg R_i(x) \right).$$

It is easy to see that in a τ -structure \mathfrak{A} , we have $\mathfrak{A} \models \psi_t[a]$ if, and only if, the a th element of $w_\mathfrak{A}$ is mapped by γ to t . Thus, in particular, the formulas $\psi_{s_1}, \dots, \psi_{s_c}$ define disjoint sets that partition the universe of \mathfrak{A} . We now claim that the quantifier $\Gamma_\gamma^{S, s}$ is defined by the formula:

$$\Gamma_\delta^{S, s}(\psi_{s_1}, \dots, \psi_{s_c}).$$

To see this, let I denote the unary interpretation $(\psi_{s_1}, \dots, \psi_{s_c})$ so that $w_{I(\mathfrak{A})}$ is a string over $\{0, 1\}^c$. Moreover, by the fact that the sets defined by the formulas $\psi_{s_1}, \dots, \psi_{s_c}$ partition $|\mathfrak{A}|$ it follows that each letter of $w_{I(\mathfrak{A})}$ is a vector in $\{0, 1\}^c$ with exactly one 1. Indeed, the a element of $w_{I(\mathfrak{A})}$ is the one-hot encoding of i precisely if $\mathfrak{A} \models \psi_{s_i}[a]$. Since δ takes the one-hot encoding of i to s_i , we have for any $a \in |\mathfrak{A}|$

$$\begin{aligned} \delta((w_{I(\mathfrak{A})})_a) &= s_i \\ \text{iff } \mathfrak{A} \models \psi_i[a] \\ \text{iff } \gamma((w_\mathfrak{A})_a) &= s_i. \end{aligned}$$

Hence, $\delta(w_{I(\mathfrak{A})}) = \gamma(w_\mathfrak{A})$ and therefore $I(\mathfrak{A}) \in \Gamma_\delta^{S, s}$ if, and only if $\mathfrak{A} \in \Gamma_\gamma^{S, s}$ as required. ◀

We now prove that having quantifiers binding only one variable is sufficient:

► **Theorem 24.** *For every finite semigroup S , there exists a function $\delta : \{0, 1\}^c \rightarrow S$ such that for every $s \in S$, $d \in \mathbb{N}$, and $\gamma : \{0, 1\}^k \rightarrow S$, the quantifier $\Gamma_{d, \gamma}^{S, s}$ is definable in $(\Gamma_{1, \delta}^S)[\emptyset]$.*

Proof. Let $S = \{s_1, \dots, s_c\}$ be an arbitrary finite semigroup and let $\delta : \{0, 1\}^c \rightarrow S$ be constructed as done in Lemma 23. Let $d \in \mathbb{N}$ and $\gamma : \{0, 1\}^k \rightarrow S$ be arbitrary and let $\tau = \{P_1^{(d)}, \dots, P_k^{(d)}\}$ be a relational vocabulary. Finally, for each $s \in S$, let

$$\Phi_1^s := \Gamma_{d, \gamma}^{S, s}(\bar{P}_1 \bar{x}, \dots, \bar{P}_k \bar{x})$$

and I_γ the interpretation of Φ_1^s . We want to show that for each $s \in S$, there exists a τ -sentence Φ_2^s in $(\Gamma_{1,\delta}^S)[\emptyset]$ such that $\text{Mod}(\Phi_1^s) = \text{Mod}(\Phi_2^s)$.

We will approach this by taking our multiplication quantifier of dimension d and “unpacking” it into a nesting of quantifiers of dimension one, with quantifier depth d . The evaluation of a d -dimensional quantifier may be viewed as being factored through the evaluation of each successive level of nesting. For example, a multiplication quantifier $\Gamma_{2,\gamma}^{S,s}$ with interpretation I is evaluated in a structure \mathfrak{A} and assignment α by checking whether $\gamma(w_{I(\mathfrak{A},\alpha)}) = s$. Because the quantifier has dimension two, the length of $w_{I(\mathfrak{A},\alpha)}$ is $||\mathfrak{A}||^2$. Instead of applying γ to the entire tuple, we may first apply γ to each consecutive $|\mathfrak{A}|$ -length subword to obtain $|\mathfrak{A}|$ elements of S which may then be multiplied together to obtain our result. Our outermost quantifier performs the multiplication of the $|\mathfrak{A}|$ elements, i.e., the intermediate results, while the innermost quantifier performs the multiplication of the elements of each subword. We will pass the intermediate result from the innermost quantifier to the outermost by encoding the result in the evaluation of the outermost quantifier’s tuple of formulas. Because we don’t know which element of S the application of γ to the subword will be, we need to ensure our tuple of large enough to encode any possible element of S . Thus, by fixing the tuple size using the same encoding as Lemma 23, we may then pass the intermediate result of the innermost quantifier’s multiplication to the outermost quantifier. We now go into the details of this construction.

We proceed by induction on the dimension d . If $d = 1$, then the result follows from Lemma 23. Thus, assume that for each $s \in S$,

$$\Gamma_{d-1,\gamma}^{S,s} \text{ is definable in } (\Gamma_{1,\delta}^S)[\emptyset]. \quad (\text{I.H.})$$

We now show that for each $s \in S$, $\Gamma_{d,\gamma}^{S,s}$ is definable in $(\Gamma_{1,\delta}^S)[\emptyset]$.

Let $s \in S$ be arbitrary. We now construct a sentence Φ^s and prove that $\text{Mod}(\Phi_1^s) = \text{Mod}(\Phi^s)$; we will then use the inductive hypothesis to convert Φ^s into a sentence Φ_2^s in $(\Gamma_{1,\delta}^S)[\emptyset]$ such that $\text{Mod}(\Phi^s) = \text{Mod}(\Phi_2^s)$. Let

$$\Phi^s := \Gamma_{1,\delta}^{S,s} x_1(\theta_1(x_1), \dots, \theta_c(x_1))$$

where

$$\theta_i(x_1) := \Gamma_{d-1,\gamma}^{S,s_i} x_2 \dots x_d (P_1 x_1 x_2 \dots x_d, \dots, P_k x_1 x_2 \dots x_d)$$

Let \mathfrak{A} be an arbitrary τ -structure and α a variable assignment. Let I_δ be the interpretation of Φ^s and I_γ^i denote the interpretation of θ_i . To show that $\text{Mod}(\Phi_1^s) = \text{Mod}(\Phi^s)$, we will show that $\gamma(w_{I_\gamma(\mathfrak{A},\alpha)}) = \delta(w_{I_\delta(\mathfrak{A},\alpha)})$.

First, note that $w_{I_\gamma(\mathfrak{A},\alpha)}$ is of length $||\mathfrak{A}||^d$ while $w_{I_\delta(\mathfrak{A},\alpha)}$ is of length $||\mathfrak{A}||$. Also, by construction of $\theta_1, \dots, \theta_c$, we get that

$$\text{for every } a \in |\mathfrak{A}|, \text{ if } \theta_i^{\mathfrak{A},\alpha}[a] = \theta_j^{\mathfrak{A},\alpha}[a] = 1, \text{ then } i = j \quad (\star)$$

since each θ_i will perform the same multiplication within S during evaluation but each θ_i will check if the product is equal to a different s_i . Then, for every $a \in [||\mathfrak{A}||]$ and $s_i \in S$,

$$\begin{aligned} \delta((w_{I_\delta(\mathfrak{A},\alpha)})_a) &= s_i \\ \text{iff } \delta(\theta_1^{\mathfrak{A},\alpha}[a] \circ \dots \circ \theta_c^{\mathfrak{A},\alpha}[a]) &= s_i && \text{by definition of } w_{I_\delta(\mathfrak{A},\alpha)} \\ \text{iff } \theta_i^{\mathfrak{A},\alpha}[a] &= 1 && \text{by construction of } \delta \text{ and } (\star) \\ \text{iff } \gamma(w_{I_\gamma^i(\mathfrak{A},\alpha[a/x_1])}) &= s_i && \text{by definition of } \theta_i \end{aligned}$$

23:12 Characterizing NC^1 with Typed Semigroups

403 Because s_i was arbitrary, we get that

$$404 \quad \delta((w_{I_\delta(\mathfrak{A}, \alpha)})_a) = \gamma(w_{I_\gamma^i(\mathfrak{A}, \alpha[a/x_1])})$$

405 and, therefore,

$$406 \quad \mathfrak{A} \models \Phi^s [\alpha]$$

$$407 \quad \text{iff } w_{I_\delta(\mathfrak{A}, \alpha)} \in L_\delta^{S, s} \quad \text{by definition of } \Gamma_{d, \delta}^{S, s}$$

$$408 \quad \text{iff } \delta(w_{I_\delta(\mathfrak{A}, \alpha)}) = s \quad \text{by definition of } L_\delta^{S, s}$$

$$409 \quad \text{iff } \prod_{1 \leq a \leq |\mathfrak{A}|} \delta((w_{I_\delta(\mathfrak{A}, \alpha)})_a) = s \quad \text{because } \delta \text{ is a homomorphism}$$

$$410 \quad \text{iff } \prod_{1 \leq a \leq |\mathfrak{A}|} \gamma(w_{I_\gamma^i(\mathfrak{A}, \alpha[a/x_1])}) = s \quad \text{by above, where } i \text{ is s.t. } s_i = s$$

$$411 \quad \text{iff } \prod_{1 \leq a \leq |\mathfrak{A}|} \gamma(w_{I_\gamma(\mathfrak{A}, \alpha[a/x_1])}) = s \quad \text{by definition of } I_\gamma \text{ and } I_\gamma^i, \text{ and } s_i = s$$

$$412 \quad \text{iff } \gamma(w_{I_\gamma(\mathfrak{A}, \alpha)}) = s \quad \text{because } \gamma \text{ is a homomorphism}$$

$$413 \quad \text{iff } w_{I_\gamma(\mathfrak{A}, \alpha)} \in L_\gamma^{S, s} \quad \text{by definition of } L_\gamma^{S, s}$$

$$414 \quad \text{iff } \mathfrak{A} \models \Phi_1^s [\alpha] \quad \text{by definition of } \Gamma_{d, \gamma}^{S, s}$$

415 so $\text{Mod}(\Phi_1^s) = \text{Mod}(\Phi^s)$.

416 By the I.H., we know that each quantifier $\Gamma_{d-1, \gamma}^{S, s_i}$ is definable in $(\Gamma_{1, \delta}^S)[\emptyset]$. Therefore, we
 417 know that for each θ_i , there exists a formula θ'_i in $(\Gamma_{1, \delta}^S)[\emptyset]$ such that $\text{Mod}(\theta_i) = \text{Mod}(\theta'_i)$.
 418 Thus, we can construct a sentence Φ_2^s by replacing each θ_i in Φ^s with θ'_i ; we immediately get
 419 that $\text{Mod}(\Phi^s) = \text{Mod}(\Phi_2^s)$. Therefore, we have constructed a sentence Φ_2^s in $(\Gamma_{1, \delta}^S)[\emptyset]$ such
 420 that $\text{Mod}(\Phi_1^s) = \text{Mod}(\Phi_2^s)$. Since $s \in S$ was arbitrary, this completes the inductive step.

421 All together, we get that for every $d \in \mathbb{N}$, $\gamma : \{0, 1\}^k \rightarrow S$, and $s \in S$, the quantifier $\Gamma_{d, \gamma}^{S, s}$
 422 is definable in $(\Gamma_{1, \delta}^S)[\emptyset]$. ◀

424 ► **Corollary 25.** *For every finite semigroup S , there exists a function $\delta : \{0, 1\}^c \rightarrow S$ such*
 425 *that for any set of quantifiers \mathfrak{Q} and set of numerical predicates \mathfrak{N} ,*

$$426 \quad \mathcal{L}((\mathfrak{Q} \cup \Gamma^S)[\mathfrak{N}]) = \mathcal{L}((\mathfrak{Q} \cup \Gamma_{1, \delta}^S)[\mathfrak{N}])$$

427 ► **Remark 26.** Because we are considering finite semigroups, we can always take disjunctions
 428 of the multiplication quantifiers which check if the product is equal to a single element of a
 429 semigroup in order to define multiplication quantifiers which check if the product is equal to
 430 any element of a specified subset of a semigroup.

431 ► **Remark 27.** Note that for a finite semigroup S , while Γ^S and Γ_1^S are infinite sets, $\Gamma_{1, \delta}^S$ is a
 432 finite set.

433 Therefore, this gives us a logic characterizing $\text{DLOGTIME-uniform NC}^1$ which not only
 434 uses unary quantifiers but also only has a finite number of quantifiers:

435 ► **Corollary 28.** *There exists a $\delta : \{0, 1\}^k \rightarrow S_5$ such that*

$$436 \quad \text{DLOGTIME-uniform NC}^1 = \mathcal{L}((\text{FO} \cup \Gamma_{1, \delta}^{S_5})[+, \times])$$

437 This will simplify our construction of an algebra capturing $\text{DLOGTIME-uniform NC}^1$.

438 Moreover, this theorem serves as an alternative proof of Theorem 10 ([12, Theorem 5.1])
 439 which, unlike the original proof, does not rely on the use of automata:

440 ► **Corollary 29.** $\text{REG} = \mathcal{L}((\Gamma^{\text{fin}})[<]) = \mathcal{L}((\Gamma_1^{\text{fin}})[<])$.

441 and, furthermore, resolves an open question from [12]:

442 ► **Corollary 30.** $\mathcal{L}((\Gamma^{\text{fin}})[+, \times]) = \mathcal{L}((\Gamma_1^{\text{fin}})[+, \times])$

443 4 The Algebraic Characterization

444 Now that we have a first-order logic with only quantifiers containing interpretations of unary
 445 dimension capturing DLOGTIME-uniform NC^1 , we are closer to applying Theorem 22 to
 446 construct an algebra for it. We now just need to convert the logic to a form whose only
 447 numerical predicate is $<$ without introducing quantifiers of a higher dimension.

448 To do this, we follow what was done for the construction of an algebra for TC^0 . First, we
 449 note that the quantifier Maj is true if the majority of the assignments to the bound variable
 450 satisfy the formula and the quantifier Sq which is true if the number of assignments to the
 451 bound variable satisfying the formula is a positive square number. In the below, we always
 452 assume the use of unary interpretations with these quantifiers.

453 The following lemma displays some known results about the expressiveness of these
 454 quantifiers:

455 ► **Lemma 31.**

- 456 (i) Maj is definable in $(\Gamma^{\text{fin}})[+, \times]$. (cf. [1])
- 457 (ii) The quantifiers in FO are definable in $(\text{Maj})[<]$. ([11, Theorem 3.2])
- 458 (iii) The numerical predicate $+$ is definable in $(\text{Maj})[<]$. ([11, Theorem 4.1])
- 459 (iv) The numerical predicate \times is definable in $(\{\text{Maj}, \text{Sq}\})[<]$ and Sq is definable in
 460 $(\text{Maj})[<, +, \times]$. (cf. [14, Theorem 2.3.f] and [10, Section 2.3])

461 Bringing everything together, we get the following algebraic characterization of DLOG-
 462 TIME-uniform NC^1 :

► **Theorem 32.**

$$463 \quad \text{DLOGTIME-uniform NC}^1 = \mathcal{L}(\text{sbpc}_{<}(\{(\mathbb{Z}, \mathbb{Z}^+, \pm 1), (\mathbb{N}, \mathbb{S}, \{0, 1\}), (S_5, \wp(S_5), S_5)\})).$$

464 **Proof.** Let $\delta : \{0, 1\}^c \rightarrow S_5$ be as it was defined in Lemma 23. It is easy to see that the
 465 typed quantifier semigroup for Maj is $(\mathbb{Z}, \mathbb{Z}^+, \pm 1)$, for Sq is $(\mathbb{N}, \mathbb{S}, \{0, 1\})$, and for $\Gamma_{1,\delta}^{S_5, s}$ is
 466 $(S_5, \{s\}, S_5)$. Then,

$$\begin{aligned}
 467 \quad \text{DLOGTIME-uniform NC}^1 &= \mathcal{L}((\text{FO} \cup \Gamma^{S_5})[+, \times]) \text{ via [1]} \\
 468 &= \mathcal{L}((\text{FO} \cup \Gamma_{1,\delta}^{S_5})[+, \times]) \text{ via Corollary 28} \\
 469 &= \mathcal{L}((\Gamma_{1,\delta}^{S_5} \cup \{\text{Maj}, \text{Sq}\})[<]) \text{ via Lemma 31} \\
 470 &= \mathcal{L}(\text{sbpc}_{<}(\{(\mathbb{Z}, \mathbb{Z}^+, \pm 1), (\mathbb{N}, \mathbb{S}, \{0, 1\})\} \\
 471 &\quad \cup \{(S_5, s, S_5) \mid s \in S_5\})) \\
 472 &\quad \text{via Theorem 22} \\
 473 &= \mathcal{L}(\text{sbpc}_{<}(\{(\mathbb{Z}, \mathbb{Z}^+, \pm 1), (\mathbb{N}, \mathbb{S}, \{0, 1\}), (S_5, \wp(S_5), S_5)\})) \\
 474 &\quad \text{since } \forall s \in S_5, (S_5, s, S_5) \preceq (S_5, \wp(S_5), S_5) \\
 475 &\quad \text{and } \mathcal{L}((S_5, \wp(S_5), S_5)) \subseteq \text{REG} \subseteq \text{DLT-uniform NC}^1
 \end{aligned}$$

476

5 Conclusion

[TODO]:

References

- 1 David A Mix Barrington, Neil Immerman, and Howard Straubing. On uniformity within NC¹. *Journal of Computer and System Sciences*, 41(3):274–306, 1990.
- 2 Christoph Behle, Andreas Krebs, and Mark Mercer. Linear circuits, two-variable logic and weakly blocked monoids. In *International Symposium on Mathematical Foundations of Computer Science*, pages 147–158. Springer, 2007.
- 3 Christoph Behle, Andreas Krebs, and Stephanie Reifferscheid. Typed monoids—An Eilenberg-like theorem for non regular languages. In *Algebraic Informatics: 4th International Conference, CAI 2011, Linz, Austria, June 21-24, 2011. Proceedings 4*, pages 97–114. Springer, 2011.
- 4 A Cano, J Cantero, and Ana Martínez-Pastor. A positive extension of Eilenberg’s variety theorem for non-regular languages. *Applicable Algebra in Engineering, Communication and Computing*, 32(5):553–573, 2021.
- 5 Stephen A Cook. Characterizations of Pushdown Machines in Terms of Time-Bounded Computers. *Journal of the ACM (JACM)*, 18(1):4–18, 1971.
- 6 H.-D. Ebbinghaus. Extended logics: The general framework. In J. Barwise and S. Feferman, editors, *Model-Theoretic Logics*, pages 25–76. Springer-Verlag, New York, 1985.
- 7 Samuel Eilenberg. *Automata, Languages, and Machines (Vol. B)*. Academic Press, 1976.
- 8 Fred C Hennie. One-tape, off-line turing machine computations. *Information and Control*, 8(6):553–578, 1965.
- 9 Andreas Krebs. *Typed semigroups, majority logic, and threshold circuits*. PhD thesis, Tübingen, Univ., Diss., 2008, 2008.
- 10 Andreas Krebs, Klaus-Jörn Lange, and Stephanie Reifferscheid. Characterizing TC₀ in terms of infinite groups. *Theory of Computing Systems*, 40(4):303–325, 2007.
- 11 K-J Lange. Some results on majority quantifiers over words. In *Proceedings. 19th IEEE Annual Conference on Computational Complexity, 2004.*, pages 123–129. IEEE, 2004.
- 12 Clemens Lautemann, Pierre McKenzie, Thomas Schwentick, and Heribert Vollmer. The descriptive complexity approach to LOGCFL. *Journal of Computer and System Sciences*, 62(4):629–652, 2001.
- 13 John Rhodes and Bret Tilson. The kernel of monoid morphisms. *J. Pure Appl. Algebra*, 62(3):227–268, 1989.
- 14 Nicole Schweikardt. *On the Expressive Power of First-order Logic with Built in Predicates*. Logos-Verlag, 2002.

A Strong Block Product Closure

A.1 Weakly Closed Classes

► **Definition 33** (Direct Product of Semigroups). *The direct product of two semigroups (S, \cdot_S) and (T, \cdot_T) is the semigroup $(S \times T, \cdot)$ where $(s_1, t_1) \cdot (s_2, t_2) = (s_1 \cdot_S s_2, t_1 \cdot_T t_2)$.*

► **Definition 34** (Direct Product of Boolean Algebras). *We define the direct product of Boolean algebras B_1 and B_2 , denoted $B_1 \times B_2$, to be the Boolean algebra generated by the set $\{A_1 \times A_2 \mid A_1 \in B_1 \text{ and } A_2 \in B_2\}$.*

► **Definition 35** (Direct Product of Typed Semigroups).

The direct product $(S, G, E) \times (T, H, F)$ is the typed semigroup $(S \times T, G \times H, E \times F)$.

520 ► **Definition 36** (Trivial Extension). *If there exists a surjective typed homomorphism from*
 521 *(S, G, E) to (T, H, F) , then we say that (S, G, E) is a trivial extension of (T, H, F) .*

522 ► **Definition 37** (Weakly Closed Class). *We call a set of typed semigroups T a weakly closed*
 523 *class if it is closed under*

524 ■ *Division: If $(S, G, E) \in T$ and $(S, G, E) \preceq (T, H, F)$, then $(T, H, F) \in T$.*

525 ■ *Direct Product: If $(S, G, E), (T, H, F) \in T$, then $(S, G, E) \times (T, H, F) \in T$.*

526 ■ *Trivial Extension: If (S, G, E) is a trivial extension of (T, H, F) and $(T, H, F) \in T$, then*
 527 *$(S, G, E) \in T$.*

528 *We write $\text{wc}(T)$ to denote the smallest weakly closed set of typed semigroups containing T .*

529 A.2 The Block Product

530 The block product will be our main tool for the construction of algebraic characterizations
 531 of language classes via logic.² We now build up to its definition:

532 ► **Definition 38** (Left and Right Actions). *A left action \star_l of a semigroup (N, \cdot) on a semigroup*
 533 *$(M, +)$ is a function from $N \times M$ to M such that for $n_1, n_2 \in N$ and $m_1, m_2 \in M$,*

$$534 \quad n \star_l (m_1 + m_2) = n \star_l m_1 + n \star_l m_2$$

$$535 \quad (n_1 \cdot n_2) \star_l m = n_1 \star_l (n_2 \star_l m)$$

536

537 *The right action \star_r of (N, \cdot) on $(M, +)$ is defined dually. We say that left and right actions*
 538 *of (N, \cdot) on $(M, +)$ are compatible if for all $n_1, n_2 \in N$ and $m \in M$,*

$$539 \quad (n_1 \star_l m) \star_r n_2 = n_1 \star_l (m \star_r n_2).$$

540 *When clear from context, we may simply write nm for $n \star_l m$ and mn for $m \star_r n$.*

541 ► **Definition 39** (Two-sided Semidirect Product). *For a pair of compatible left and right*
 542 *actions, \star_l and \star_r of (N, \cdot) on $(M, +)$, the two-sided (or bilateral) semidirect product*
 543 *of $(M, +)$ and (N, \cdot) with respect to \star_l and \star_r is the semigroup $(M \times N, \circ)$ where for*
 544 *$(m_1, n_1), (m_2, n_2) \in M \times N$,*

$$545 \quad (m_1, n_1) \circ (m_2, n_2) = (m_1 n_2 + n_1 m_2, n_1 \cdot n_2).$$

546 ► **Definition 40** (Block Product). *The block product of (M, \cdot_M) with (N, \cdot_N) , denoted $M \square N$,*
 547 *is the two-sided semidirect product of $(M^{N^1 \times N^1}, +)$ and (N, \cdot) with respect to the left and*
 548 *right actions \star_l and \star_r where for $f, g \in M^{N^1 \times N^1}$ and $n, n_1, n_2 \in N^1$,*

549 ■ *$(M^{N^1 \times N^1}, +)$ is the monoid of all functions from $N^1 \times N^1$ to M under componentwise*
 550 *product $+$:*

$$551 \quad (f + g)(n_1, n_2) = f(n_1, n_2) \cdot_M g(n_1, n_2).$$

552 ■ *The left action \star_l of (N, \cdot) on $(M^{N^1 \times N^1}, +)$ is defined by*

$$553 \quad (n \star_l f)(n_1, n_2) = f(n_1 \cdot_N n, n_2).$$

554 ■ *The right action \star_r of (N, \cdot) on $(M^{N^1 \times N^1}, +)$ is defined by*

$$555 \quad (f \star_r n)(n_1, n_2) = f(n_1, n \cdot_N n_2).$$

² Historically, the “wreath product” was first used for this purpose. Since [13], however, the block product has been the preferred and easier-to-work-with tool of choice.

556 A.3 The Typed Block Product

557 ► **Definition 41** (Typed Block Product). *Let (S, G, E) and (S', G', E') be typed semigroups*
 558 *and $C \subseteq S'$ be a finite set. Then, the typed block product with C of (S, G, E) and (S', G', E') ,*
 559 *denoted $(S, G, E) \boxtimes_C (S', G', E')$, is the typed semigroup (T, H, F) where*

- 560 (1) $T \leq S \boxtimes S'$ such that T is generated by the elements (f, s') such that
 561 (a) $s' \in E' \cup C$ and
 562 (b) $f \in E^{S'^1 \times S'^1}$ such that for $b_1, b_2, b_3, b_4 \in S'$, if for all $c \in C$ and all $A' \in G'$,
 563 $b_1 c b_2 \in A'$ iff $b_3 c b_4 \in A'$, then $f(b_1, b_2) = f(b_3, b_4)$,
 564 (2) $H = \{ \{(f, s) \mid f(1, 1) \in A\} \mid A \in G \}$ where 1 is the identity of S'^1 ,
 565 (3) and $F = \{(f, s') \mid (f, s) \text{ is a generator of } T \text{ and } s' \in E'\}$.

566 ► **Definition 42.** *Because the typed semigroup corresponding to the order predicate will be*
 567 *a very common, it is convenient to define an ordered typed block product, $(S, G, E) \boxtimes_C$*
 568 *(S', G', E') which will help simplify our algebraic representations whose numerical predicates*
 569 *only include order; this is defined the same as the typed block product above but with a change*
 570 *to condition (1)(b):*

- 571 (1)(b_<) $f \in E^{S'^1 \times S'^1}$ such that for $b_1, b_2, b_3, b_4 \in S'$, if for all $c \in C$ and all $A' \in G'$,
 572 (i) $b_1 c b_2 \in A'$ iff $b_3 c b_4 \in A'$,
 573 (ii) $b_1 c \in A'$ iff $b_3 c \in A'$,
 574 (iii) and $c b_2 \in A'$ iff $c b_4 \in A'$,
 575 then $f(b_1, b_2) = f(b_3, b_4)$.

576 ► **Definition 43.** *For a set of typed semigroups W , we let*

577
$$W_0 = \text{wc}(W)$$

578 and for each $k \geq 1$,

579
$$\blacksquare W_k = \{S_1 \boxtimes_C S_2 \mid S_1 \in W_0, S_2 \in W_{k-1}, \text{ and finite } C \subseteq S_2\}$$

580
$$\blacksquare W_k^< = \{S_1 \boxtimes_C S_2 \mid S_1 \in W_0, S_2 \in W_{k-1}^<, \text{ and finite } C \subseteq S_2\}$$

581 We define the (ordered) strong block product closure of W , denoted $\text{sbpc}(W)$ ($\text{sbpc}_<(W)$), as

582
$$\blacksquare \text{sbpc}(W) = \bigcup_{k \in \mathbb{N}} W_k$$

583
$$\blacksquare \text{sbpc}_<(W) = \bigcup_{k \in \mathbb{N}} W_k^<.$$