Characterizing NC^1 with Typed Semigroups

Anonymous author
Anonymous affiliation

Anonymous author
Anonymous affiliation

Abstract

[TODO]:

2012 ACM Subject Classification Replace ccsdesc macro with valid one

Keywords and phrases Dummy keyword

Digital Object Identifier 10.4230/LIPIcs.CVIT.2016.23

Acknowledgements Anonymous acknowledgements

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

1 Introduction

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

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

When it comes to studying computational complexity, we are mainly interested in classes of languages richer than just the regular languages. Thus the syntactic semigroups of the languages are not necessarily finite any longer and the extensive tools of Krohn-Rhodes theory are not available to study them. Nonetheless, some attempts have been made to extend the methods of algebraic automata theory to classes beyond the regular languages. Most significant is the work of Krebs and collaborators [2, 3, 10, 9, 4], which introduces the notion of *typed semigroups*. The idea is to allow for languages with infinite syntactic semigroups, but limit the languages they recognize by associating with them a finite collection of types. This allows for the formulation of a version of Eilenberg's Correspondence theorem associating closure properties on classes of typed semigroups with corresponding closure properties of classes of languages. In particular, this implies that most complexity classes of interest can be uniquely characterized in terms of an associated class of typed semigroups [3]. An explicit description of the class characterizing DLogTime-uniform TC^0 is given in [10, 9]. This is obtained through a general method which allows us to construct typed semigroups corresponding to *unary quantifiers* defined from specific languages [9] (see also Theorem 25 below).

In this paper, we extend this work to obtain a characterization of DLogTime-uniform NC^1 as the class of languages recognized by the collection of typed semigroups obtained as the closure under *ordered strong block products* of three typed semigroups: the group of integers with types for positive and negative integers; the group of natural numbers with types for the square numbers and non-square numbers; and a finite non-solvable group such as S_5 with a type for each subset of the group. Full definitions of these terms follow below. Our result is obtained by first characterizing 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 ($\mathrm{FO}(+,\times)$) [1]. From this, we know we can describe NC^1 as the class of languages definable in an extension of $\mathrm{FO}(+,\times)$ 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 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 $\{1,\ldots,n\}$ 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 $(S,\cdot_S)$, we say that a set $G \subseteq S$ *generates* S if S is equal to the closure of G under $\cdot_S$; we denote this by $S = \langle G\rangle_{\cdot_S}$, 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 $h : S \to T$ 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 $\eta : S \to S/\sim$ defined by $\eta(a) = [a]$ is then a homomorphism, known as the *canonical homomorphism* of S onto $S/\sim$.

For a finite alphabet Σ and a language $L \subseteq \Sigma^*$, define the equivalence relation $\sim_L$ on Σ^* by $x \sim_L y$ if for all $u, v \in \Sigma^*$ we have $uxv \in L$ if, and only if, $uyv \in L$. It is easily seen that this relation is a congruence on the free monoid Σ^*. The quotient semigroup $\Sigma^*/\sim_L$ 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 $h : \Sigma^+ \to S$ 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 $R_1, \ldots, R_k$. 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 1.** *For τ a relational vocabulary consisting of a binary relation symbol $<$ and unary relation symbols $R_1, \ldots, R_k$, 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 whose ith element is the k-tuple whose jth position is 1 if, and only if, R_j holds at the ith element of $\mathfrak{A}$.*

As the elements of a string $\mathfrak{A}$ are linearly ordered, we can identify them with an initial segment $\{1, \ldots, n\}$ of the positive integers. In other words, we treat a string with universe $\{1, \ldots, n\}$ 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 $\tau = \langle R_1, \ldots, R_l\rangle$, 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 $I = (\phi_1(\overline{x}_1), \ldots, \phi_l(\overline{x}_l))$ of vocabulary σ where ϕ_i is associated with a tuple $\overline{x}_i$ 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 $\{\overline{a} \in |\mathfrak{A}|^{dr_i} \mid \mathfrak{A} \models \phi_i[\alpha[\overline{x}_i/\overline{a}]]\}$. 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

$$Q\overline{x}_1 \cdots \overline{x}_l(\phi_1, \ldots, \phi_l)$$

in which the variables among $\overline{x_i}$ bind free variables in the subformula ϕ_i. The semantics of this quantifier are given by the rule that $Q\overline{x}_1 \cdots \overline{x}_l(\phi_1, \ldots, \phi_l)$ 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.

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 tranducers* [**?**]. 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 $(\phi_1(\overline{x}_1), \ldots, \phi_l(\overline{x}_l))$, where all tuples of variables have length d. We can then assume, without loss of generality, that they are all the same tuple $\overline{x}$ and we thus write a formula with a string quantifier Q as

$$Q\overline{x}(\phi_1, \ldots, \phi_l).$$

Observe that a quantifier 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 2.** *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 $(\mathfrak{Q})[\mathfrak{N}]$.

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 $\mathcal{L}((\mathfrak{Q})[\mathfrak{N}]) = \mathcal{L}((\mathfrak{Q} \cup \{Q\})[\mathfrak{N}])$.

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 NC^1 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 $\gamma : \Sigma_k \to S$. 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, $w_{\mathfrak{A}}$ denotes the string associated with a structure $\mathfrak{A}$ in the sense of Definition 1.

▶ **Definition 3.** *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 $\gamma : \{0,1\}^k \to S$, the quantifier $\Gamma_\gamma^{S,B}$ is the Lindström quantifier associated with the class of structures*

$$\{\mathfrak{A} \mid w_{\mathfrak{A}} \in L_\gamma^{S,B}\}.$$

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 $\gamma : \{0,1\} \to U_1$ such that $\gamma(0) = 1$ and $\gamma(1) = 0$, is the standard existential quantifier. The universal quantifier can be defined similarly.

▶ **Definition 4.** *For a semigroup S, we define the following sets of quantifiers:*

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

Finally, let Γ^{fin} be the set of all multiplication quantifiers over finite semigroups and Γ_1^{fin} be the set of all unary multiplication quantifiers over finite semigroups.

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

▶ **Theorem 5 ([1]).** DLogTime-uniform $NC^1 = \mathcal{L}((\text{FO} \cup \Gamma^{\text{fin}})[+,\times])$.

▶ Remark 6. In fact, simply the set of multiplication quantifiers for some finite, non-solvable monoid will suffice. 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 DLogTime-uniform $NC^1 = \mathcal{L}((\text{FO} \cup \Gamma^{S_5})[+,\times])$.

We also have a similar characterization for the regular languages:

▶ **Theorem 7 ([1]).** $\text{Reg} = \mathcal{L}((\text{FO} \cup \Gamma_1^{\text{fin}})[<])$.

Later, [12, Theorem 5.1] showed that introducing non-unary quantifiers doesn't increase the expressive power in the case of order predicates:

▶ **Theorem 8.** $\text{Reg} = \mathcal{L}((\text{FO} \cup \Gamma^{\text{fin}})[<])$.

2.4 Typed Semigroups

▶ **Definition 9** (Boolean Algebra). *A* Boolean algebra *over a set S is a set $B \subseteq \wp(S)$ such that $\varnothing, S \in B$ and B is closed under union, intersection, and complementation. If B is finite, we call it a* finite *Boolean algebra.*

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

▶ **Definition 10.** *Let B_1 and B_2 be Boolean algebras over sets S and T, respectively. We call $h : B_1 \to B_2$ a* homomorphism of Boolean algebras *if $h(\varnothing) = \varnothing$, $h(S) = T$, 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 $h(s^C) = (h(s))^C$.*

▶ **Definition 11** (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 $G = \{\varnothing, A, S - A, S\}$, 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.

▶ **Definition 12.** *A* typed homomorphism *$h : (S, G, E) \to (T, H, F)$ of typed semigroups is a triple (h_1, h_2, h_3) where $h_1 : S \to T$ is a semigroup homomorphism, $h_2 : G \to H$ is a homomorphism of Boolean algebras, and $h_3 : E \to F$ 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 $e \in E$, $h_1(e) = h_3(e)$.*

▶ **Definition 13.** *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.*

We then have the following definitions and facts about typed semigroups:

▶ **Proposition 14.** *If the base monoid of a typed semigroup T is finite, then $\mathcal{L}(T) \subseteq$* REG*.*

▶ **Definition 15.** *Let (S, G, E) and (T, H, F) be typed semigroups.*

- *A typed homomorphism $h = (h_1, h_2, h_3) : (S, G, E) \to (T, H, F)$ is* injective *(*surjective*, or* bijective*) if 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 $(S, G, E) \le (T, H, F)$, if S is a subsemigroup of T and there exists an injective typed homomorphism $h : (S, G, E) \to (T, H, F)$.*
- *(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).*

▶ **Proposition 16** ([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 $\mathcal{L}(T_1) \subseteq \mathcal{L}(T_2)$.*

▶ **Definition 17.** *Let L be a language. We define the* syntactic congruence *of L as the relation $\sim_L$ on Σ^+ such that for all $x, y \in \Sigma^+$, $x \sim_L y$ if and only if for all $w, v \in \Sigma^+$, $wxv \in L$ iff $wyv \in L$.*

▶ **Definition 18.** *The* syntactic semigroup *of a language $L \subseteq \Sigma^+$ is the quotient semigroup $\Sigma^+/{\sim_L}$. We call the canonical homomorphism $\eta_d : \Sigma^+ \to \Sigma^+/{\sim_L}$ the* syntactic homomorphism *of L.*

▶ Remark 19. Observe that η_d is surjective.

▶ **Definition 20.** *Let $T = (S, G, E)$ be a typed semigroup. A congruence $\sim$ over S is a* typed 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 $s_2 \in A$.*

For a typed congruence $\sim$ over T, let

$$S'/{\sim} = \{[x]_\sim \mid x \in S'\} \text{ where } S' \subseteq S$$
$$G/{\sim} = \{A/{\sim} \mid A \in G\}$$
$$E/{\sim} = \{[x]_\sim \mid x \in E\}.$$

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

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 $x, y \in S$ and $A \in G$, $xs_1y \in A$ iff $xs_2y \in A$. We then refer to the quotient semigroup $T/{\sim_T}$ as the minimal reduced semigroup of T.

▶ **Definition 21.** *For a language $L \subseteq \Sigma^+$, we define the* syntactic typed semigroup of L, *denoted* $\mathrm{syn}(L)$, *to be the typed semigroup $(\Sigma^+, L, \Sigma)/{\sim_L}$. Recall that $\sim_L$ is the syntactic congruence of L, defined in Definition 17.*

We also get the canonical typed homomorphism, $\eta_d : (\Sigma^+, L, \Sigma) \to \mathrm{syn}(L)$ *induced by the syntactic homomorphism of L.*

▶ **Definition 22.** *For a* unary *multiplication quantifier $Q = \Gamma^{S,A}_{1,\gamma}$ where $\gamma : \{0,1\}^k \to S$, we define the* typed quantifier semigroup *of Q, denoted $\mathcal{S}(Q)$, to be the syntactic typed semigroup of the language $L_Q \subseteq (\{0,1\}^k)^+$ where $w \in L_Q$ iff*

$$w \models Qx\langle B_1(x), \ldots, B_k(x)\rangle$$

where $w_{x=i} \models B_j x$ iff the j^{th} bit of w_i equals 1. Thus, $\mathcal{S}(Q) = ((\{0,1\}^k)^+, L_Q, \{0,1\}^k)/{\sim_{L_Q}}$.

▶ **Proposition 23** ([9])**.** *A typed semigroup is the syntactic semigroup of a language iff it is reduced, generated by its unites, and has four or two types. (In the case of two types, then it only recognizes the empty language or the language of all strings.)*

▶ **Definition 24.** *For a set of typed semigroups T, we denote by $\mathrm{sbpc}_<(T)$ the* ordered strong block product closure of T. *Because the definition of this closure is quite technical but not needed to understand the proofs in this paper, we include it in Appendix A.*

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

▶ **Theorem 25.** *Let $\mathfrak{Q}$ be a set of unary quantifiers and $\boldsymbol{Q}$ its set of typed quantifier semigroups for $\mathfrak{Q}$. Then, $\mathcal{L}((\mathfrak{Q})[<]) = \mathcal{L}(\mathrm{sbpc}_<(\boldsymbol{Q}))$.*

3 Simplifying Multiplication Quantifiers

We aim to construct an algebraic characterization of DLogTime-uniform NC[1] by taking advantage of Theorem 25. To do so, however, we need a logic which characterizes DLogTime-uniform NC[1] using only unary first-order quantifiers.

[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.

Now, from Remark 6, we know of a logic containing non-unary first-order quantifiers:

$$\text{DLogTime-uniform NC}^1 = \mathcal{L}((\text{FO} \cup \Gamma^{S_5})[+, \times])$$

To take us a step closer to applying Theorem 25, we will prove that we can substitute multiplication quantifiers of higher dimension with unary ones, therefore proving that having unary quantifiers alone suffices to express the same languages:

$$\mathcal{L}((\text{FO} \cup \Gamma^{S_5})[+, \times]) = \mathcal{L}((\text{FO} \cup \Gamma_1^{S_5})[+, \times]),$$

Answering a question left open in [12].

While we only need to show an equivalence of $(FO + \Gamma^{S_5})[+, \times]$ and $(FO + \Gamma_1^{S_5})[+, \times]$ at the language level—i.e., that they express the same languages—we will actually prove the stronger claim that for every finite semigroup S, all quantifiers in Γ^S are definable in $(\Gamma_1^S)[\varnothing]$. In other words, we will prove that any use of Γ^S quantifiers may be substituted by a $(\Gamma_1^S)[\varnothing]$ formulae without loss or gain in expressive power. Moreover, we will prove that we don't need an infinite number of quantifiers to express DLogTime-uniform NC^1. Simply a finite set of multiplication quantifiers binding one variable and extending over k-tuples (for some fixed k) will suffice.

We first prove that we can fix the size of the tuple over which the quantifier acts:

▶ **Lemma 26.** *For every finite semigroup S, there exists a function $\delta : \{0,1\}^c \to S$ such that for every $s \in S$, $d \in \mathbb{N}$, and $\gamma : \{0,1\}^k \to S$, the quantifier $\Gamma^{S,s}_{d,\gamma}$ is definable in $(\Gamma^{S,s}_{d,\delta})[\varnothing]$.*

Proof. Let S be an arbitrary finite semigroup.

To fix the tuple size, we will increase the tuple size so that each element $s \in S$ may be associated with a unique element of $v \in \{0,1\}^c$ and set $\delta(v) = s$. We will then construct the formulas $\psi_1(\overline{x}), \dots, \psi_c(\overline{x})$ defining the interpretation I_δ of $\Gamma^{S,s}_{d,\delta}\overline{x}(\psi_1(\overline{x}), \dots, \psi_c(\overline{x}))$ in such a way that for the interpretation I_γ of $\Gamma^{S,s}_{d,\gamma}\overline{x}(P_1, \dots, P_k)$, structure $\mathfrak{A}$, and assignment α, $\gamma(w_{I_\gamma(\mathfrak{A},\alpha)}) = s = \delta(w_{I_\delta(\mathfrak{A},\alpha)})$ and the letters of $w_{I_\delta(\mathfrak{A},\alpha)}$ are only from the subset of $\{0,1\}^c$ consisting of the unique elements used to construct δ.

To construct δ, we will let $c = |S|$ be the size of the tuples over which δ acts and we'll use one-hot encodings to associate each element of S with an element in the domain of δ. Let $z \in S$ be fixed and arbitrary. Say that $S = \{s_1, \dots, s_c\}$. Let $\delta : \{0,1\}^c \to S$ such that if $w \in \{0,1\}^c$ is a one-hot encoding of i where $1 \le i \le c$, then $\delta(w) = s_i$; else, $\delta(w) = z$. For example, if $|S| = 3$, then $\delta(100) = s_1$, $\delta(010) = s_2$, $\delta(001) = s_3$, $\delta(110) = \delta(000) = z$, etc. Note that z is simply used to ensure that δ is total; the construction of our formulas $\psi_1, \dots, \psi_c$ bound by the multiplication quantifier using δ will ensure that no non-one-hot encoding in $\{0,1\}^c$ is ever passed into δ during evaluation of the quantifier.

Now that we've defined δ, let $s \in S$, $d \in \mathbb{N}$, and $\gamma : \{0,1\}^k \to S$ be arbitrary. Let $\tau = \{P_1^{(d)}, \dots, P_k^{(d)}\}$ be a relational vocabulary. We will now show that $\Gamma^{S,s}_{d,\gamma}$ is definable in $(\Gamma^{S,s}_{d,\delta})[\varnothing]$.

Specifically, we will now show that for

$$\Phi_1 := \Gamma^{S,s}_{d,\gamma}\overline{x}(P_1\overline{x}, \dots, P_k\overline{x_d})$$

there exists a τ-sentence

$$\Phi_2 := \Gamma^{S,s}_{d,\delta}\overline{x}(\psi_1(\overline{x}), \dots, \psi_c(\overline{x_d})),$$

where each ψ_i is a boolean combination of $P_1, \dots, P_k$, such that $\text{Mod}(\Phi_1) = \text{Mod}(\Phi_2)$.

We now construct $\psi_1, \ldots, \psi_c$ so that for each structure $\mathfrak{A}$, assignment α, and all $\overline{a} \in |\mathfrak{A}|^d$, if $\gamma(P_1^{\mathfrak{A},\alpha}[\overline{a}] \circ \cdots \circ P_k^{\mathfrak{A},\alpha}[\overline{a}]) = s_j$, then $\psi_j[\overline{a}] = 1$ and $\psi_i[\overline{a}] = 0$ for all $i \neq j$. Thus, $\psi_1^{\mathfrak{A},\alpha}[\overline{a}] \circ \cdots \circ \psi_c^{\mathfrak{A},\alpha}[\overline{a}]$ will be the one-hot encoding of s_j, causing the application of δ to output s_j.

Let γ^P be a map from S to sets of boolean combinations of $P_1, \ldots, P_k$ such that if $w_1 \ldots w_k \in \{0,1\}^k$ maps to s under γ, then $P_1' \wedge \cdots \wedge P_k' \in \gamma^P(s)$ where $P_i' = P_i\overline{x}$ if $w_i = 1$ and $P_i' = \neg P_i\overline{x}$ if $w_i = 0$. For example, if $S = \{s_1, s_2, s_3\}$, $k = 2$, $\gamma(00) = \gamma(10) = \gamma(01) = s_1$, and $\gamma(11) = s_3$, then $\gamma^P(s_1) = \{\neg P_1\overline{x} \wedge \neg P_2\overline{x}, P_1\overline{x} \wedge \neg P_2\overline{x}, \neg P_1\overline{x} \wedge P_2\overline{x}\}$, $\gamma^P(s_2) = \varnothing$, and $\gamma^P(s_3) = \{P_1\overline{x} \wedge P_2\overline{x}\}$. We then set

$$\psi_i := \bigvee_{\phi \in \gamma^P(s_i)} \phi.$$

By construction since γ is a total function, observe that for every structure, there will be *exactly* one i such that ψ_i evaluates to true. We have now defined $\psi_1, \ldots, \psi_c$ and, thus, Φ_2.

We now show that $\mathrm{Mod}(\Phi_1) = \mathrm{Mod}(\Phi_2)$.

Let $\mathfrak{A}$ be an arbitrary τ-structure and α a variable assignment. Because we are operating over dimension d, the length of $w_{I_\gamma(\mathfrak{A},\alpha)}$ is $||\mathfrak{A}||^d$.

We first aim to prove that $\gamma((w_{I_\gamma(\mathfrak{A},\alpha)})_i) = \delta((w_{I_\delta(\mathfrak{A},\alpha)})_i)$ for all $i \in [||\mathfrak{A}||^d]$. Call this proposition $(\star)$. Let $i \in [||\mathfrak{A}||^d]$ and $s_j \in S = \{s_1, \ldots, s_c\}$ be arbitrary. Let $\overline{a}$ be a tuple of length d and denote the base-$||\mathfrak{A}||$ encoding of i. Then,

$$\begin{aligned}
&\gamma((w_{I_\gamma(\mathfrak{A},\alpha)})_i) = s_j && \\
\text{iff } &\gamma(P_1^{\mathfrak{A},\alpha}[\overline{a}] \circ \cdots \circ P_k^{\mathfrak{A},\alpha}[\overline{a}]) = s_j && \text{by definition of } w_{I_\gamma(\mathfrak{A},\alpha)} \\
\text{iff } &\psi_j^{\mathfrak{A},\alpha}[\overline{a}] = 1 && \text{by construction of } \psi_j \\
\text{iff } &\delta(\psi_1^{\mathfrak{A},\alpha}[\overline{a}] \circ \cdots \circ \psi_c^{\mathfrak{A},\alpha}[\overline{a}]) = s_j && \text{by construction of } \delta \\
\text{iff } &\delta((w_{I_\delta(\mathfrak{A},\alpha)})_i) = s_j && \text{by definition of } \delta((w_{I_\delta(\mathfrak{A},\alpha)})_i)
\end{aligned}$$

Because s_i was arbitrary, it follows that $\gamma((w_{I_\gamma(\mathfrak{A},\alpha)})_i) = \delta((w_{I_\delta(\mathfrak{A},\alpha)})_i)$. We will use this fact to help prove that Φ_1 and Φ_2 have the same models:

$$\begin{aligned}
&\mathfrak{A} \models \Phi_1\ [\alpha] && \\
\text{iff } &w_{I_\gamma(\mathfrak{A},\alpha)} \in L_\gamma^{S,s} && \text{by definition of } \Gamma_{d,\gamma}^{S,s} \\
\text{iff } &\gamma(w_{I_\gamma(\mathfrak{A},\alpha)}) = s && \text{by definition of } L_\gamma^{S,s} \\
\text{iff } &\prod_{1 \leq i \leq ||\mathfrak{A}||^d} \gamma((w_{I_\gamma(\mathfrak{A},\alpha)})_i) = s && \text{because } \gamma \text{ is a homomorphism} \\
\text{iff } &\prod_{1 \leq i \leq ||\mathfrak{A}||^d} \delta((w_{I_\delta(\mathfrak{A},\alpha)})_i) = s && \text{by } (\star) \\
\text{iff } &\delta(w_{I_\delta(\mathfrak{A},\alpha)}) = s && \text{because } \delta \text{ is a homomorphism} \\
\text{iff } &w_{I_\delta(\mathfrak{A},\alpha)} \in L_\delta^{S,s} && \text{by definition of } L_\delta^{S,s} \\
\text{iff } &\mathfrak{A} \models \Phi_2\ [\alpha] && \text{by definition of } \Gamma_{l,\delta}^{S,s}
\end{aligned}$$

We have now proved that $\mathrm{Mod}(\Phi_1) = \mathrm{Mod}(\Phi_2)$. ◀

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

▶ **Theorem 27.** *For every finite semigroup S, there exists a function $\delta : \{0,1\}^c \to S$ such that for every $s \in S$, $d \in \mathbb{N}$, and $\gamma : \{0,1\}^k \to S$, the quantifier $\Gamma_{d,\gamma}^{S,s}$ is definable in $(\Gamma_{1,\delta}^S)[\varnothing]$.*

Proof. Let $S = \{s_1, \ldots, s_c\}$ be an arbitrary finite semigroup and let $\delta : \{0,1\}^c \to S$ be constructed as done in Lemma 26. Let $d \in \mathbb{N}$ and $\gamma : \{0,1\}^k \to S$ be arbitrary and let $\tau = \{P_1^{(d)}, \ldots, P_k^{(d)}\}$ be a relational vocabulary. Finally, for each $s \in S$, let

$$\Phi_1^s := \Gamma_{d,\gamma}^{S,s} \overline{x}(P_1\overline{x}, \ldots, P_k\overline{x})$$

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 $(\Gamma_{1,\delta}^S)[\varnothing]$ such that $\mathrm{Mod}(\Phi_1^s) = \mathrm{Mod}(\Phi_2^s)$.

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 26, 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 26. Thus, assume that for each $s \in S$,

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

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

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

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

where

$$\theta_i(x_1) := \Gamma_{d-1,\gamma}^{S,s_i} x_2 \ldots x_d(P_1x_1x_2 \ldots x_d, \ldots, P_kx_1x_2 \ldots x_d)$$

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 $\mathrm{Mod}(\Phi_1^s) = \mathrm{Mod}(\Phi^s)$, 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 $\theta_1, \ldots, \theta_c$, we get that

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

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 $s_i \in S$,

$$\delta((w_{I_\delta(\mathfrak{A},\alpha)})_a) = s_i$$

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

Because s_i was arbitrary, we get that

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

and, therefore,

$$
\begin{aligned}
&\quad \mathfrak{A} \models \Phi^s\ [\alpha]\\
&\text{iff } w_{I_\delta(\mathfrak{A},\alpha)} \in L_\delta^{S,s} && \text{by definition of } \Gamma_{d,\delta}^{S,s}\\
&\text{iff } \delta(w_{I_\delta(\mathfrak{A},\alpha)}) = s && \text{by definition of } L_\delta^{S,s}\\
&\text{iff } \prod_{1\le a\le ||\mathfrak{A}||} \delta((w_{I_\delta(\mathfrak{A},\alpha)})_a) = s && \text{because } \delta \text{ is a homomorphism}\\
&\text{iff } \prod_{1\le a\le ||\mathfrak{A}||} \gamma(w_{I_\gamma^i(\mathfrak{A},\alpha[a/x_1])}) = s && \text{by above, where } i \text{ is s.t. } s_i = s\\
&\text{iff } \prod_{1\le a\le ||\mathfrak{A}||} \gamma(w_{I_\gamma(\mathfrak{A},\alpha[a/x_1])}) = s && \text{by definition of } I_\gamma \text{ and } I_\gamma^i, \text{ and } s_i = s\\
&\text{iff } \gamma(w_{I_\gamma(\mathfrak{A},\alpha)}) = s && \text{because } \gamma \text{ is a homomorphism}\\
&\text{iff } w_{I_\gamma(\mathfrak{A},\alpha)} \in L_\gamma^{S,s} && \text{by definition of } L_\gamma^{S,s}\\
&\text{iff } \mathfrak{A} \models \Phi_1^s\ [\alpha] && \text{by definition of } \Gamma_{d,\gamma}^{S,s}
\end{aligned}
$$

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

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

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

◀

▶ **Corollary 28.** *For every finite semigroup S, there exists a function $\delta : \{0,1\}^c \to S$ such that for any set of quantifiers $\mathfrak{Q}$ and set of numerical predicates $\mathfrak{N}$,*

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

▶ Remark 29. Because we are considering finite semigroups, we can always take disjunctions of the multiplication quantifiers which check if the product is equal to a single element of a semigroup in order to define multiplication quantifiers which check if the product is equal to any element of a specified subset of a semigroup.

▶ Remark 30. Note that for a finite semigroup S, while Γ^S and Γ_1^S are infinite sets, $\Gamma_{1,\delta}^S$ is a finite set.

Therefore, this gives us a logic characterizing DLogTime-uniform NC^1 which not only uses unary quantifiers but also only has a finite number of quantifiers:

▶ **Corollary 31.** *There exists a* $\delta : \{0,1\}^k \to S_5$ *such that*

$$\text{DLogTime-uniform NC}^1 = \mathcal{L}((\text{FO} \cup \Gamma^{S_5}_{1,\delta})[+,\times])$$

This will simplify our construction of an algebra capturing DLogTime-uniform NC^1.

Moreover, this theorem serves as an alternative proof of Theorem 8 ([12, Theorem 5.1]) which, unlike the original proof, does not rely on the use of automata:

▶ **Corollary 32.** $\text{Reg} = \mathcal{L}((\text{FO} \cup \Gamma^{\text{fin}})[<]) = \mathcal{L}((\text{FO} \cup \Gamma^{\text{fin}}_1)[<])$.

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

▶ **Corollary 33.** $\mathcal{L}((\text{FO} \cup \Gamma^{\text{fin}})[+,\times]) = \mathcal{L}((\text{FO} \cup \Gamma^{\text{fin}}_1)[+,\times])$

4 The Algebraic Characterization

Now that we have a first-order logic with only unary quantifiers capturing DLogTime-uniform NC^1, we are closer to applying Theorem 25 to construct an algebra for it. We now just need to convert the logic to a form whose only numerical predicate is $<$ without introducing any new non-unary quantifiers.

To do this, we first define two unary quantifiers. The quantifier Maj which is true if the majority of the assignments to the bound variable satisfy the formula. The quantifier Sq which is true if the number of assignments to the bound variable satisfying the formula is a positive square number.

We note the following from the literature:

▶ **Lemma 34.**
- **(i)** DLogTime-uniform NC^1 *can compute majority.*
- **(ii)** *The quantifiers in* FO *are definable in* $(\text{Maj})[<]$. *([11, Theorem 3.2])*
- **(iii)** *The numerical predicate* $+$ *is definable in* $(\text{Maj})[<]$. *([11, Theorem 4.1])*
- **(iv)** *The numerical predicate* $\times$ *is definable in* $(\{\text{Maj}, \text{Sq}\})[<]$ *and* Sq *is definable in* $(\text{Maj})[<,+,\times]$. *(cf. [14, Theorem 2.3.f] and [10, Section 2.3])*

▶ **Theorem 35.**

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

Proof. Let $\delta : \{0,1\}^c \to S_5$ be as it was defined in Lemma 26. Note that the typed quantifier semigroup for Maj is $(\mathbb{Z},\mathbb{Z}^+,\pm 1)$, for Sq is $(\mathbb{N},\mathbb{S},\{0,1\})$, and for $\Gamma^{S_5,s}_{1,\delta}$ is $(S_5,\{s\},S_5)$. Then,

$$\begin{aligned}
\text{DLogTime-uniform NC}^1 &= \mathcal{L}((\text{FO} \cup \Gamma^{S_5})[+,\times]) \text{ via [1]}\\
&= \mathcal{L}((\text{FO} \cup \Gamma^{S_5}_{1,\delta})[+,\times]) \text{ via Corollary 31}\\
&= \mathcal{L}((\Gamma^{S_5}_{1,\delta} \cup \{\text{Maj},\text{Sq}\})[<]) \text{ via Lemma 34}\\
&= \mathcal{L}(\text{sbpc}_<(\{(\mathbb{Z},\mathbb{Z}^+,\pm 1),(\mathbb{N},\mathbb{S},\{0,1\})\}\\
&\qquad\qquad \cup \{(S_5,s,S_5) \mid s \in S_5\}))\\
&\qquad \text{via Theorem 25}\\
&= \mathcal{L}(\text{sbpc}_<(\{(\mathbb{Z},\mathbb{Z}^+,\pm 1),(\mathbb{N},\mathbb{S},\{0,1\}),(S_5,\wp(S_5),S_5)\}))\\
&\qquad \text{since } \forall s \in S_5, (S_5,s,S_5) \preceq (S_5,\wp(S_5),S_5)\\
&\qquad \text{and } \mathcal{L}((S_5,\wp(S_5),S_5)) \subseteq \text{Reg} \subseteq \text{ALogTime}
\end{aligned}$$

◀

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 36** (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 37** (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 38** (Direct Product of Typed Semigroups).
The direct product *$(S, G, E) \times (T, H, F)$ is the typed semigroup $(S \times T, G \times H, E \times F)$.*

▶ **Definition 39** (Trivial Extension). *If there exists a surjective typed homomorphism from (S, G, E) to (T, H, F), then we say that (S, G, E) is a* trivial extension *of (T, H, F).*

▶ **Definition 40** (Weakly Closed Class). *We call a set of typed semigroups T a* weakly closed class *if it is closed under*

- *Division: If $(S, G, E) \in T$ and $(S, G, E) \preceq (T, H, F)$, then $(T, H, F) \in T$.*
- *Direct Product: If $(S, G, E), (T, H, F) \in T$, then $(S, G, E) \times (T, H, F) \in T$.*
- *Trivial Extension: If (S, G, E) is a trivial extension of (T, H, F) and $(T, H, F) \in T$, then $(S, G, E) \in T$.*

We write wc(T) *to denote the smallest weakly closed set of typed semigroups containing T.*

A.2 The Block Product

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

▶ **Definition 41** (Left and Right Actions). *A* left action $\star_l$ *of a semigroup $(N, \cdot)$ on a semigroup $(M, +)$ is a function from $N \times M$ to M such that for $n_1, n_2 \in N$ and $m_1, m_2 \in M$,*

$$n \star_l (m_1 + m_2) = n \star_l m_1 + n \star_l m_2$$
$$(n_1 \cdot n_2) \star_l m = n_1 \star_l (n_2 \star_l m)$$

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

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

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

▶ **Definition 42** (Two-sided Semidirect Product). *For a pair of compatible left and right actions, $\star_l$ and $\star_r$ of $(N, \cdot)$ on $(M, +)$, the* two-sided (or bilateral) semidirect product *of $(M, +)$ and $(N, \cdot)$ with respect to $\star_l$ and $\star_r$ is the semigroup $(M \times N, \circ)$ where for $(m_1, n_1), (m_2, n_2) \in M \times N$,*

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

▶ **Definition 43** (Block Product). *The* block product *of $(M, \cdot_M)$ with $(N, \cdot_N)$, denoted $M \Box N$, is the two-sided semidirect product of $(M^{N^1 \times N^1}, +)$ and $(N, \cdot)$ with respect to the left and right actions $\star_l$ and $\star_r$ where for $f, g \in M^{N^1 \times N^1}$ and $n, n_1, n_2 \in N^1$,*

- *$(M^{N^1 \times N^1}, +)$ is the monoid of all functions from $N^1 \times N^1$ to M under componentwise product $+$:*
$$(f + g)(n_1, n_2) = f(n_1, n_2) \cdot_M g(n_1, n_2).$$
- *The left action $\star_l$ of $(N, \cdot)$ on $(M^{N^1 \times N^1}, +)$ is defined by*
$$(n \star_l f)(n_1, n_2) = f(n_1 \cdot_N n, n_2).$$
- *The right action $\star_r$ of $(N, \cdot)$ on $(M^{N^1 \times N^1}, +)$ is defined by*
$$(f \star_r n)(n_1, n_2) = f(n_1, n \cdot_N n_2).$$

[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.

A.3 The Typed Block Product

▶ **Definition 44** (Typed Block Product). *Let* (S, G, E) *and* (S', G', E') *be typed semigroups and* $C \subseteq S'$ *be a finite set. Then, the* typed block product with C *of* (S, G, E) *and* (S', G', E')*, denoted* $(S, G, E) \boxdot_C (S', G', E')$*, is the typed semigroup* (T, H, F) *where*

(1) $T \leq S \square S'$ *such that* T *is generated by the elements* (f, s') *such that*
  (a) $s' \in E' \cup C$ *and*
  (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* $A' \in G'$*,* $b_1 c b_2 \in A'$ *iff* $b_3 c b_4 \in A'$*, then* $f(b_1, b_2) = f(b_3, b_4)$*,*

(2) $H = \{\{(f, s) \mid f(1, 1) \in A\} \mid A \in G\}$ *where* 1 *is the identity of* S'^1*,*

(3) *and* $F = \{(f, s') \mid (f, s)$ *is a generator of* T *and* $s' \in E'\}$*.*

▶ **Definition 45.** *Because the typed semigroup corresponding to the order predicate will be a very common, it is convenient to define an* ordered typed block product*,* $(S, G, E) \boxtimes_C (S', G', E')$ *which will help simplify our algebraic representations whose numerical predicates only include order; this is defined the same as the typed block product above but with a change to condition (1)(b):*

(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* $A' \in G'$*,*
  (i) $b_1 c b_2 \in A'$ *iff* $b_3 c b_4 \in A'$*,*
  (ii) $b_1 c \in A'$ *iff* $b_3 c \in A'$*,*
  (iii) *and* $c b_2 \in A'$ *iff* $c b_4 \in A'$*,*
  then $f(b_1, b_2) = f(b_3, b_4)$*.*

▶ **Definition 46.** *For a set of typed semigroups* W*, we let*

$$W_0 = \mathrm{wc}(W)$$

and for each $k \geq 1$*,*

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

We define the (ordered) strong block product closure of W*, denoted sbpc*(W) *(sbpc*$_<(W)$*), as*

- *sbpc*$(W) = \bigcup_{k \in \mathbb{N}} W_k$
- *sbpc*$_<(W) = \bigcup_{k \in \mathbb{N}} W_k^<$.