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. 23:1–23:17
Leibniz International Proceedings in Informatics
LIPICS Schloss Dagstuhl – Leibniz-Zentrum für Informatik, Dagstuhl Publishing, Germany

1 Introduction

We call classes of languages like P and NP *complexity classes* because the historically predominant approach to defining them is via the medium of a machine together with a complexity measure. Likewise, we favor the nomenclature of *formal languages* for the classes of regular and context-free languages because of the historical context in which they first gained wide-spread attention, i.e., the study of formal grammars. Ultimately, however, all these classes are simply classes of languages and there exist possible worlds where, for example, the regular languages would be predominantly called a complexity class, P an 'automaton class', and NP a 'logic class'.[1] These different media of recognition, i.e., the mathematical objects we use to carve out classes of languages, simply provide us different ways to characterize the same abstract objects. Therefore, in being different characterizations, they, provide different insights into the nature of their respective classes and tools with which to study them.

Historically, the three main alternative[2] characterizations of classes of regular languages used by researchers during the early study of formal language theory were those via automata, logics, and algebras. The automata/machine-based approaches to language recognition had the benefits of being intuitive to understand from the algorithm construction, programming, and systems building perspectives. The logic-based approaches helped us break down the structure of languages into their component parts because of the use of quantifiers, numerical predicates, and logical connectives. Finally, the algebra-based approaches helped us understand the deeper structure of the languages.

The algebraic approach, known as *algebraic automata theory* or *Krohn-Rhodes theory*, characterized classes of regular languages with the languages recognized by classes of finite semigroups via homomorphisms. This approach to the study of languages proved to be a powerful tool for with it, we found success in proving results where other approaches, such as those via machines or logics, had either failed completely or, at the very least, failed to easily generalize.[3] However, with finite semigroups, we limit ourselves to the study of classes of regular languages.

Research continued in the 1970s and after for providing logics and machines which characterized classes of languages well beyond the regular languages—e.g., P, NP, PH, etc.— yet research into the construction of algebras characterizing these classes did not. In fact, it would not be until the 2000s that the algebraic approach to representing classes of languages would finally be properly generalized by the use of *typed semigroups* to robustly handle non-regular language classes [3, 4, 11, 10, 5]. Still, however, we have yet to construct algebraic characterizations[4] of classes any larger than DLogTime-uniform TC^0.

---

[1] In [9], Hennie proved that linear-time-bounded single-tape Turing machines recognize exactly the regular languages. In [6], Cook provides an automata theoretic definition of P, proving that the languages recognized by multi-head two-way pushdown automata are exactly those in P; this provided a machine-based definition of P without any reference to complexity measures. In [8], Fagin proved that the languages expressible by existential second-order logic are exactly those in NP.

[2] Alternative to the use of regular expressions and grammars.

[3] A long-standing open question was whether there existed an effective procedure for determining whether a give regular expression recognized a star-free language or not. An answer eluded researchers completely until the star-free regular languages were characterized via algebra; once they were, an algorithm followed naturally from the work. Also, when it came to separating subclasses of regular languages, often game-like techniques were used, i.e., Ehrenfeucht-Fraïssé games, but these methods were often difficult to transfer from one subclass to another; with algebra, however, separations typically just fell-out immediately as a result of Eilenberg's Correspondence Theorem [7] and the study of the syntactic semigroup.

[4] To be clear, by "algebraic approach" and "algebraic characterizations" here, we do not mean algebraic

In this work, we take the first steps towards constructing algebraic characterizations of more non-regular language classes by constructing one for DLogTime-uniform NC^1. The core of the proofs reduces to answering an open question from the work of [13] regarding the expressive power of first-order logic over strings extended with arithmetic predicates and what Barrington, Immerman, and Straubing refer to as generalized quantifiers [2]—or as we more generally refer to them here, multiplication quantifiers. Specifically, we prove that the expressive power over strings of a logic (regardless of the numerical predicates or quantifiers in use) extended with multiplication quantifiers binding one variable is the same as if it was extended with multiplication quantifiers binding multiple variables. Combined with results from [3, 10], we obtain our main result.

In Section 2, we cover the relevant background material on semigroup theory, typed semigroups, and multiplication quantifiers. In Section 3, we prove our intermediate results regarding the expressive power of multiplication quantifiers. In Section 4, we prove our main result providing an algebraic characterization of DLogTime-uniform NC^1. Finally, in Section 5, we conclude and outline future work.

2 Preliminaries

We assume familiarity with the basic concepts of formal language theory, automata theory, complexity theory, and finite model theory. We do not assume familiarity with algebraic automata theory or algebra in general.

In Section 2.1, we clarify the standard notations and conventions used in this report. In Section 2.2, we cover the necessary background material on semigroups and groups. In Section 2.3.2, we cover logic and multiplication quantifiers. In Section 2.4, we cover the algebraic approach to the recognition of languages via typed semigroups needed for Section 4.

2.1 Notations and Conventions

▶ **Definition 1.** *We let* $[n] = \{1, \ldots, n\}$.

▶ **Definition 2.**
- $\mathbb{Z}$ *denotes the set of integers.*
- $\mathbb{N}$ *denotes the set of natural numbers, including 0.*
- $\mathbb{Z}^+$ *denotes the set of positive integers.*
- $\mathbb{S}$ *denotes the set of positive square integers.*

▶ **Definition 3.** *For a tuple* t*, we denote by* $\pi_i(t)$ *the* i^{th} *element of* t*.*

▶ **Definition 4.** *A* one-hot encoding *of an integer* $i \in [n]$ *is a length* n *binary string* b *such that* $b_j = 1$ *if* $j = i$ *and* $b_j = 0$ *otherwise. For example, the one-hot encoding of* $3 \in [5]$ *is 00100.*

geometry, arithmetic circuits, VP vs VNP, or anything that is typically referred to today as algebraic complexity theory (cf. [16, Ch. 12]). Neither should one confuse "algebra" here with the portmanteau "algebraization" (cf. [1]). Unless otherwise stated, we are only referring to algebra as it is understood in the older field of algebraic automata theory.

2.2 Semigroup and Group Theory

▶ **Definition 5** (Semigroups, Monoids, and Groups)**.** *A* semigroup $(S, \cdot)$ *is a set* S *closed under an* associative *binary operation* $\cdot : S \times S \to S$. *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 an 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. *For a semigroup* S, *we denote by* S^1, *the monoid generated by* S; *i.e.,* $S = S^1$ *if* S *is a monoid or, otherwise, we introduce a new element* 1 *to* S *and define it to be the identity.*

A group $(G, \cdot)$ *is a monoid with the additional property 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.

▶ Remark 6. Observe that all groups are monoids and all monoids are semigroups.

▶ Remark 7. If $\mathbb{Z}$ or $\mathbb{N}$ is referred to as a semigroup, then we assume the operation to be the usual addition unless stated otherwise.

▶ **Definition 8.** *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.

▶ Remark 9. Unless otherwise stated, we assume that all semigroups are finitely generated.

▶ **Definition 10.** *A semigroup homomorphism* $h : S \to T$ *is a function from a semigroup* $(S, \cdot_S)$ *to a semigroup* $(T, \cdot_T)$ *such that for all* $s_1, s_2 \in S$, $h(s_1 \cdot_S s_2) = h(s_1) \cdot_T h(s_2)$.

▶ **Definition 11.** U_1 *is the monoid over* $\{0, 1\}$ *with multiplication defined as usual.*

▶ **Definition 12.** *For a set* S, *we denote by* S^+ *the set of non-empty strings over* S *and by* S^* *the set of all strings over* S. *We call* S^+ (S^*) *the* free semigroup (monoid) *over* S *and will denote string concatenation by either* $\circ$ *or simply juxtaposition.*

▶ **Definition 13.** *We say that a semigroup* $(S, \cdot)$ *is* cancellative *if for* $a, b, c \in S$, *(1) if* $a \cdot b = a \cdot c$, *then* $b = c$ *and (2) if* $b \cdot a = c \cdot a$, *then* $b = c$.

▶ **Proposition 14.** *Every group is a cancellative semigroup.*

▶ **Definition 15** (Congruence, Quotient Semigroup, Canonical Homomorphism)**.** *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* $\sim$.

We may then define a semigroup $(S/\sim, \star)$ *where for* $[a], [b] \in S/\sim$, $[a] \star [b] = [a \cdot b]$. *We call this semigroup the* quotient semigroup *of* S *by* $\sim$.

We then define the canonical homomorphism $\eta : S \to S/\sim$ *by* $\eta(a) = [a]$.

2.3 Logics and Multiplication Quantifiers

In Section 2.3.1, we clarify our notations and definitions regarding logics. In Section 2.3.2, we define multiplication quantifiers and provide the necessary background material on them. In Section 2.3.3, we make some brief remarks regarding *definability* in logics.

2.3.1 Logics

▶ Remark 16. In this report, we will only consider finite structures over relational vocabularies; specifically, we assume that all structures are over initial segments of the natural numbers, excluding 0. Thus, for a structure $\mathfrak{A}$, $|\mathfrak{A}| = [n] = \{1, \ldots, n\}$ for some $n \in \mathbb{N}$.

▶ Remark 17. We often consider logics containing numerical predicates. These are predicates included in the vocabulary of the logic and are interpreted in the natural way. For example, for the order predicate $<$, we say that $\mathfrak{A} \models a < b$ iff $a^{\mathfrak{A}}$ is less than $b^{\mathfrak{A}}$, where $a^{\mathfrak{A}} \in |\mathfrak{A}| = [n]$, for some $n \in \mathbb{N}$, is the interpretation of variable a in $\mathfrak{A}$. We will also refer to the commonly used numerical predicates $=$, $+$, and $\times$.

▶ Remark 18. We let $\mathrm{Mod}(\varphi)$ denote all models of a formula φ: $\mathrm{Mod}(\varphi) = \{\mathfrak{A} \mid \mathfrak{A} \models \varphi\}$.

▶ Remark 19. Let $\mathfrak{A}$ be a structure. For a formula $\varphi(x_1, \ldots, x_k)$ with free variables $x_1, \ldots, x_k$, we denote by $\varphi^{\mathfrak{A}}[a_1, \ldots, a_k]$ the function which maps the tuple $(a_1, \ldots, a_k)$ to the truth value of φ in $\mathfrak{A}$ when the free variables are interpreted as $a_1, \ldots, a_k \in |\mathfrak{A}|$. For example, if $\varphi(x) := x < 3$, then $\varphi^{\mathfrak{A}}[a] = 1$ if $a^{\mathfrak{A}} < 3$ and 0 otherwise.

▶ **Definition 20.** *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 the numerical predicates in $\mathfrak{N}$.*

For a singleton set of quantifiers $\mathfrak{Q} = \{Q\}$, we will sometimes denote $(\mathfrak{Q})[\mathfrak{N}]$ as $(Q)[\mathfrak{N}]$. We use similar notation for the sets of numerical predicates.

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

▶ **Definition 21.** *We say that a quantifier is* unary *if it binds only one variable.*

Besides the standard first-order quantifiers, we will also use the following two unary first-order quantifiers in this paper:

▶ **Definition 22.**
- Maj *is the unary* majority quantifier *such that for a structure $\mathfrak{A}$, $\mathfrak{A} \models \mathrm{Maj}x\varphi(x)$ iff $|\{a \in \mathfrak{A} \mid \varphi^{\mathfrak{A}}[a] = 1\}| > ||\mathfrak{A}||/2$.*
- Sq *is the unary* square quantifier *such that for a structure $\mathfrak{A}$, $\mathfrak{A} \models \mathrm{Sq}x\varphi(x)$ iff $|\{a \in \mathfrak{A} \mid \varphi^{\mathfrak{A}}[a] = 1\}|$ is a positive square number.*

▶ **Example 23.**
- $(\mathrm{FO})[=]$ is our usual first-order logic with equality.
- $(\mathrm{Maj})[<]$ is majority logic, i.e., it contains the majority quantifier and the order predicate.
- $(\varnothing)[\varnothing]$ is quantifier-free first-order logic with no numerical predicates.

▶ **Definition 24.** *We say that a logical formula is a* depth-k formula *if its quantifier depth is at most k.*

▶ **Definition 25.** *We say that a structure $\mathfrak{A}$ is a* string structure *over $\Sigma = \{\sigma_1, \ldots, \sigma_c\}$ if it is ordered and over a relational vocabulary $\tau = \{R_{\sigma_1}, \ldots, R_{\sigma_c}\}$ where each R_{σ_i} is unary and for every $a \in |\mathfrak{A}|$, there exists exactly one σ_i such that $a \in R_{\sigma_i}^{\mathfrak{A}}$. Thus, we may interpret a string $w \in \Sigma^+$ as a string structure over Σ, and vice versa.*

We say that a language $L \subseteq \Sigma^+$ is expressible *by a logic $\mathfrak{L}$ if there exists a $\mathfrak{L}$-sentence φ over a unary relational vocabulary $\tau = \{R_{\sigma_1}, \ldots, R_{\sigma_c}\}$ such that for all string structures $\mathfrak{A}$ over Σ, $\mathfrak{A} \models \varphi$ iff $\mathfrak{A} \in L$—or more precisely, iff $\mathfrak{A}$ encodes a string in L.*

For a logic $\mathfrak{L}$, we denote by $\mathcal{L}(\mathfrak{L})$, the languages expressible by $\mathfrak{L}$.

2.3.2 Multiplication Quantifiers

The definition of multiplication quantifier has its origin in Barrington, Immerman, and Straubing [2, Section 5] where they were refered to as monoid quantifiers or generalized 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. We now define 'multiplication quantifiers':

▶ **Definition 26** (Multiplication Quantifiers). *Let S be a semigroup, $B \subseteq S$, and $\gamma : \{0,1\}^k \to S$ a total function. We call $\Gamma^{S,B}_{l,\gamma}$ the* multiplication quantifier *for S, B, and γ which binds l variables and extends over a k-tuple of formulae. If $B = \{s\}$ is a singleton, then we simply write $\Gamma^{S,s}_{l,\gamma}$.*

For an ordered structure $\mathfrak{A}$ and formulae $\varphi_1, \dots, \varphi_k$, we evaluate the formula

$$\Phi := \Gamma^{S,B}_{l,\gamma} x_1 \dots x_l \langle \varphi_1(x_1, \dots, x_l), \dots, \varphi_k(x_1, \dots, x_l)\rangle$$

as follows. Let $\gamma^{\mathfrak{A}} : |\mathfrak{A}|^l \to S$ be a function such that

$$\gamma^{\mathfrak{A}}(a_1, \dots, a_l) = \gamma(\varphi_1^{\mathfrak{A}}[a_1, \dots, a_l], \dots, \varphi_k^{\mathfrak{A}}[a_1, \dots, a_l]).$$

We call $\gamma^{\mathfrak{A}}$ the evaluator function *for our formula. We say that $\mathfrak{A} \models \Phi$ iff*

$$\prod_{a_1 \in |\mathfrak{A}|} \cdots \prod_{a_l \in |\mathfrak{A}|} \gamma^{\mathfrak{A}}(a_1, \dots, a_l) \in B$$

where the products iterate over the elements of $|\mathfrak{A}|$ based on the order of the structure.

▶ **Example 27.** Our normal first-order existential quantifier may be represented by $\Gamma^{U_1,0}_{1,\gamma}$ where $\gamma : \{0,1\} \to U_1$ such that $\gamma(0) = 1$ and $\gamma(1) = 0$. Similar goes for the universal quantifier.

▶ **Definition 28.** *For a semigroup S, we define the following sets of quantifiers:*

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

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

$$\Gamma^S_{l,\gamma} = \left\{\Gamma^{S,B}_{l,\gamma} \mid B \subseteq S\right\}$$

Finally, let Γ^{fin} be the set of all multiplication quantifiers over finite semigroups and Γ^{fin}_1 be the set of all unary multiplication quantifiers over finite semigroups.

From [2, Corollary 9.1], we know that DLogTime-uniform NC^1 is characterized by (FO)$[+,\times]$ equipped with finite multiplication quantifiers:

▶ **Theorem 29** ([2]). DLogTime-uniform $NC^1 = \mathcal{L}((\mathrm{FO} \cup \Gamma^{\text{fin}})[+,\times])$.

▶ Remark 30. 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 $NC^1 = \mathcal{L}((\mathrm{FO} \cup \Gamma^{S_5})[+,\times])$.

We also have a similar characterization for the regular languages:

▶ **Theorem 31** ([2]). $\mathrm{Reg} = \mathcal{L}((\mathrm{FO} \cup \Gamma^{\text{fin}}_1)[<])$.

Later, [13, Theorem 5.1] showed that introducing non-unary quantifiers doesn't increase the expressive power in the case of order predicates:

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

2.3.3 Definability

▶ **Definition 33.** *Say that a first-order quantifier Q binds l variables and extends over a k-tuple of formulae. We say that Q is* definable *in a logic $\mathfrak{L}$ if there exists a sentence φ in $\mathfrak{L}$ over a vocabulary $\tau = \{R_1^{(l)}, \ldots, R_k^{(l)}\}$, i.e., each relation is l-ary, such that for all structures $\mathfrak{A}$, $\mathfrak{A} \models \varphi$ iff $\mathfrak{A} \models Qx_1 \ldots x_l \langle R_1(x_1, \ldots, x_l), \ldots, R_k(x_1, \ldots, x_l)\rangle$.*

▶ Remark 34. Observe that if a quantifier Q is definable in a logic $\mathfrak{L}$, then any use of Q in a formula can be substituted with a formula from $\mathfrak{L}$.

▶ **Definition 35.** *Similar to the above, we say that a k-ary numerical predicate R is* definable *in a logic $\mathfrak{L}$ if there exists a formula φ in $\mathfrak{L}$ with free variables $x_1, \ldots, x_k$ such that for all structures $\mathfrak{A}$, $\mathfrak{A} \models \varphi(x_1, \ldots, x_k)$ iff $\mathfrak{A} \models Rx_1 \ldots x_k$.*

2.4 Typed Semigroups

▶ **Definition 36** (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 37.** *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 38** (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 39.** *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 40.** *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 41.** *If the base monoid of a typed semigroup T is finite, then $\mathcal{L}(T) \subseteq$ REG.*

▶ **Definition 42.** *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 $(S, G, E) \leq (T, H, F)$, 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 43** ([4]). *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 44.** *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 45.** *The* syntactic semigroup *of a language* $L \subseteq \Sigma^+$ *is the quotient semigroup* $\Sigma^+/{\sim_L}$. *We call the canonical homomorphism* $\eta_L : \Sigma^+ \to \Sigma^+/{\sim_L}$ *the* syntactic homomorphism *of* L.

▶ Remark 46. Observe that η_L is surjective.

▶ **Definition 47.** *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 48.** *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 44.*

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

▶ **Definition 49.** *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 50** ([10]). *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 51.** *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 [10, Theorem 4.14], we then get the following relationship between logics and algebras:[5]

▶ **Theorem 52.** *Let $\mathfrak{Q}$ be a set of 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 52. To do so, however, we need a logic which characterizes DLogTime-uniform NC^1 using only unary first-order quantifiers.

Now, from Remark 30, we know of a logic containing non-unary first-order quantifiers:

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

Thus, to take us a step closer to applying Theorem 52, we will prove in this section that having unary quantifiers alone suffices to express the same languages:

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

which answers an open question first raised in [13].

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

Proof. Let $\overline{x_l}$ denote the tuple $(x_1, \ldots, x_l)$. Let S be an arbitrary finite semigroup.

To construct δ, we will let $|S|$ be the size of the tuples over which δ acts; thus, let $c = |S|$. Let $z \in S$ be fixed and arbitrary. Say that $S = \{s_1, \ldots, s_c\}$. 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.

Now, let $s \in S$, $l \in \mathbb{N}$, and $\gamma : \{0,1\}^k \to S$ be arbitrary. Let $\tau = \{P_1^{(l)}, \ldots, P_k^{(l)}\}$ be a relational vocabulary. We will now show that $\Gamma_{l,\gamma}^{S,s}$ is definable in $(\Gamma_{l,\delta}^{S,s})[\varnothing]$.

Specifically, we will now show that for

$$\Phi_1 := \Gamma_{l,\gamma}^{S,s} \overline{x_l} \langle P_1 \overline{x_l}, \ldots, P_k \overline{x_l} \rangle$$

there exists a τ-sentence

$$\Phi_2 := \Gamma_{l,\delta}^{S,s} \overline{x_l} \langle \psi_1(\overline{x_l}), \ldots, \psi_c(\overline{x_l}) \rangle,$$

[5] The theorem in [10] is actually more general as it accounts for more predicates than just order; however, for our purposes, order alone suffices.

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

We now construct $\psi_1, \ldots, \psi_c$.

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 $P'_i = P_i\overline{x_l}$ if $w_i = 1$ and $P'_i = \neg P_i\overline{x_l}$ 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 $\gamma^P(s_1) = \{\neg P_1\overline{x_l} \wedge \neg P_2\overline{x_l}, P_1\overline{x_l} \wedge \neg P_2\overline{x_l}, \neg P_1\overline{x_l} \wedge P_2\overline{x_l}\}$, $\gamma^P(s_2) = \varnothing$, and $\gamma^P(s_3) = \{P_1\overline{x_l} \wedge P_2\overline{x_l}\}$. 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. Let $\gamma^{\mathfrak{A}} : |\mathfrak{A}|^l \to S$ and $\delta^{\mathfrak{A}} : |\mathfrak{A}|^l \to S$ be the evaluator functions for Φ_1 and Φ_2, respectively. Thus, for $\overline{a_l} \in |\mathfrak{A}|^l$,

$$\begin{aligned} &\gamma^{\mathfrak{A}}(\overline{a_l}) = \gamma(\langle P_1^{\mathfrak{A}}[\overline{a_l}], \ldots, P_k^{\mathfrak{A}}[\overline{a_l}]\rangle) \\ \text{and } &\delta^{\mathfrak{A}}(\overline{a_l}) = \delta(\langle \psi_1^{\mathfrak{A}}[\overline{a_l}], \ldots, \psi_c^{\mathfrak{A}}[\overline{a_l}]\rangle). \end{aligned}$$

By construction of δ and ψ_i, observe that $\gamma^{\mathfrak{A}}$ and $\delta^{\mathfrak{A}}$ are in fact the same function. Let $s_i \in S = \{s_1, \ldots, s_c\}$ be arbitrary:

$$\begin{aligned} &\gamma^{\mathfrak{A}}(\overline{a_l}) = s_i && \\ \text{iff } &\gamma(\langle P_1^{\mathfrak{A}}[\overline{a_l}], \ldots, P_k^{\mathfrak{A}}[\overline{a_l}]\rangle) = s_i && \text{by definition of } \gamma^{\mathfrak{A}} \\ \text{iff } &\psi_i^{\mathfrak{A}}[\overline{a_l}] = 1 && \text{by construction of } \psi_i \\ \text{iff } &\delta(\langle \psi_1^{\mathfrak{A}}[\overline{a_l}], \ldots, \psi_c^{\mathfrak{A}}[\overline{a_l}]\rangle) = s_i && \text{by construction of } \delta \\ \text{iff } &\delta^{\mathfrak{A}}(\overline{a_l}) = s_i && \text{by definition of } \delta^{\mathfrak{A}} \end{aligned}$$

Therefore,

$$\prod_{a_1 \in |\mathfrak{A}|} \cdots \prod_{a_l \in |\mathfrak{A}|} \gamma^{\mathfrak{A}}(a_1, \ldots, a_l) = \prod_{a_1 \in |\mathfrak{A}|} \cdots \prod_{a_l \in |\mathfrak{A}|} \delta^{\mathfrak{A}}(a_1, \ldots, a_l)$$

so by definition of our multiplication quantifiers,

$$\mathfrak{A} \models \Phi_1 \text{ iff } \mathfrak{A} \models \Phi_2$$

and, thus, $\mathrm{Mod}(\Phi_1) = \mathrm{Mod}(\Phi_2)$. ◀

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

▶ **Theorem 54.** *For every finite semigroup S, there exists a function $\delta : \{0,1\}^c \to S$ such that for every $s \in S$, $l \in \mathbb{N}$, and $\gamma : \{0,1\}^k \to S$, the quantifier $\Gamma^{S,s}_{l,\gamma}$ is definable in $(\Gamma^S_{1,\delta})[\varnothing]$.*

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 53. Let $l \in \mathbb{N}$ and $\gamma : \{0,1\}^k \to S$ be arbitrary and let $\tau = \{P_1^{(l)}, \ldots, P_k^{(l)}\}$ be a relational vocabulary. Finally, for each $s \in S$, let

$$\Phi_1^s := \Gamma^{S,s}_{l,\gamma}\overline{x_l}\langle P_1\overline{x_l}, \ldots, P_k\overline{x_l}\rangle.$$

We want to show that for each $s \in S$, there exists a τ-sentence Φ_2^s in $(\Gamma^S_{1,\delta})[\varnothing]$ such that $\mathrm{Mod}(\Phi_1^s) = \mathrm{Mod}(\Phi_2^s)$.

We proceed by induction on l. If $l = 1$, then the result follows from Lemma 53. Thus, assume that for each $s \in S$,

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

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

Let $s \in S$ be arbitrary. We now construct a sentence Φ^s and prove that $\mathrm{Mod}(\Phi^s_1) = \mathrm{Mod}(\Phi^s)$; we will then use the inductive hypothesis to convert Φ^s into a sentence Φ^s_2 in $(\Gamma^{S}_{1,\delta})[\varnothing]$ such that $\mathrm{Mod}(\Phi^s) = \mathrm{Mod}(\Phi^s_2)$. Let

$$\Phi^s := \Gamma^{S,s}_{1,\delta} x_1 \langle \theta_1(x_1), \ldots, \theta_c(x_1)\rangle$$

where

$$\theta_i(x_1) = \Gamma^{S,s_i}_{l-1,\gamma} x_2 \ldots x_l \langle P_1 x_1 x_2 \ldots x_l, \ldots, P_k x_1 x_2 \ldots x_l\rangle$$

Let $\mathfrak{A}$ be an arbitrary τ-structure. Let $\gamma^{\mathfrak{A}} : |\mathfrak{A}|^l \to S$ and $\delta^{\mathfrak{A}} : |\mathfrak{A}| \to S$ be the evaluator functions for Φ^s_1 and Φ^s, respectively. To show that $\mathrm{Mod}(\Phi^s_1) = \mathrm{Mod}(\Phi^s)$, we will show that

$$\prod_{a_1 \in |\mathfrak{A}|} \cdots \prod_{a_l \in |\mathfrak{A}|} \gamma^{\mathfrak{A}}(a_1, \ldots, a_l) = \prod_{a \in |\mathfrak{A}|} \delta^{\mathfrak{A}}(a).$$

First, note that by construction of $\theta_1, \ldots, \theta_c$, we get that

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

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 $s_i \in S$,

$$\begin{aligned}
&\delta^{\mathfrak{A}}(a) = s_i \\
\text{iff } &\delta(\langle \theta^{\mathfrak{A}}_1[a], \ldots, \theta^{\mathfrak{A}}_c[a]\rangle) = s_i && \text{by definition of } \delta^{\mathfrak{A}} \\
\text{iff } &\theta^{\mathfrak{A}}_i[a] = 1 && \text{by construction of } \delta \text{ and } (\star)
\end{aligned}$$

Then, by construction of θ_i, we get that $\delta^{\mathfrak{A}}(a) = s_i$ iff

$$\prod_{a_2 \in |\mathfrak{A}|} \cdots \prod_{a_l \in |\mathfrak{A}|} \gamma(\langle P^{\mathfrak{A}}_1[a, a_2, \ldots, a_l], \ldots, P^{\mathfrak{A}}_k[a, a_2, \ldots, a_l]\rangle) = s_i$$

and, thus,

$$\delta^{\mathfrak{A}}(a) = \prod_{a_2 \in |\mathfrak{A}|} \cdots \prod_{a_l \in |\mathfrak{A}|} \gamma(\langle P^{\mathfrak{A}}_1[a, a_2, \ldots, a_l], \ldots, P^{\mathfrak{A}}_k[a, a_2, \ldots, a_l]\rangle)$$

Therefore,

$$\begin{aligned}
\prod_{a \in |\mathfrak{A}|} \delta^{\mathfrak{A}}(a) &= \prod_{a \in |\mathfrak{A}|} \prod_{a_2 \in |\mathfrak{A}|} \cdots \prod_{a_l \in |\mathfrak{A}|} \gamma(\langle P^{\mathfrak{A}}_1[a, a_2, \ldots, a_l], \ldots, P^{\mathfrak{A}}_k[a, a_2, \ldots, a_l]\rangle) \\
&= \prod_{a_1 \in |\mathfrak{A}|} \cdots \prod_{a_l \in |\mathfrak{A}|} \gamma^{\mathfrak{A}}(a_1, \ldots, a_l) \text{ by definition of } \gamma^{\mathfrak{A}}
\end{aligned}$$

and, thus, $\mathfrak{A} \models \Phi^s_1$ iff $\mathfrak{A} \models \Phi^s$ so $\mathrm{Mod}(\Phi^s_1) = \mathrm{Mod}(\Phi^s)$.

By the I.H., we know that each quantifier $\Gamma^{S,s_i}_{l-1,\gamma}$ is definable in $(\Gamma^{S}_{1,\delta})[\varnothing]$. Therefore, we know that for each θ_i, there exists a formula θ'_i in $(\Gamma^{S}_{1,\delta})[\varnothing]$ 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 $l \in \mathbb{N}$, $\gamma : \{0,1\}^k \to S$, and $s \in S$, the quantifier $\Gamma_{l,\gamma}^{S,s}$ is definable in $(\Gamma_{1,\delta}^S)[\varnothing]$. ◀

▶ **Corollary 55.** *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 56. 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 57. 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 58.** *There exists a $\delta : \{0,1\}^k \to S_5$ such that*

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

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

Moreover, this theorem serves as an alternative proof of Theorem 32 ([13, Theorem 5.1]) which, unlike the original proof, does not rely on the use of automata:

▶ **Corollary 59.** $\mathrm{Reg} = \mathcal{L}((\mathrm{FO} \cup \Gamma^{\mathrm{fin}})[<]) = \mathcal{L}((\mathrm{FO} \cup \Gamma_1^{\mathrm{fin}})[<])$.

and, furthermore, resolves an open question from [13]:

▶ **Corollary 60.** $\mathcal{L}((\mathrm{FO} \cup \Gamma^{\mathrm{fin}})[+,\times]) = \mathcal{L}((\mathrm{FO} \cup \Gamma_1^{\mathrm{fin}})[+,\times])$

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 52 to construct an algebra for it.

We first need to prove some results concerning the typed quantifier semigroups of multiplication quantifiers:

▶ **Theorem 61.** *Let $s \in S_5$ and $\gamma : \{0,1\}^k \to S_5$, where $\mathrm{Img}(\gamma) = S_5$, be arbitrary. Then, the typed quantifier semigroup of $\Gamma_{1,\gamma}^{S_5,s}$ equals (S_5, s, S_5).*

Proof. Let $s \in S_5$ and $\gamma : \{0,1\}^k \to S_5$ be arbitrary. Let $T = (S_5, s, S_5)$. We will show that T is isomorphic to the typed quantifier semigroup of $\Gamma_{1,\gamma}^{S_5,s}$.

Let $Q = ((\{0,1\}^k)^+, L_\Gamma, \{0,1\}^k)$ such that for $w = w_1 \ldots w_n \in (\{0,1\}^k)^+$, $w \in L_\Gamma$ iff $w \models \Gamma_{1,\gamma}^{S_5,s} x \langle P_1 x, \ldots, P_k x \rangle$ where $P_i^{\mathfrak{w}} = \{a \in [n] \mid (w_a)_i = 1\}$. To be clear, $(w_a)_i$ denotes the i^{th} bit of $w_a \in \{0,1\}^k$. Let $\gamma^* : (\{0,1\}^k)^+ \to S_5$ be the homomorphism induced by γ.

By definition of typed quantifier semigroup, we now want to show that $T \cong \mathrm{syn}(L_\Gamma)$. We know (1) that there exists a syntactic typed homomorphism $\eta = (\eta_1, \eta_2, \eta_3)$ from Q to $\mathrm{syn}(L_\Gamma)$ and (2) that for $w \in (\{0,1\}^k)^+$,

$$
\begin{aligned}
w \in L_\Gamma \text{ iff } & w \models \Gamma_{1,\gamma}^{S_5,s} x\langle P_1 x, \ldots, P_k x\rangle \\
\text{iff } & \prod_{a \in [n]} \gamma(\langle P_1^{\mathfrak{w}}[a], \ldots, P_k^{\mathfrak{w}}[a]\rangle) = s \\
\text{iff } & \gamma^*(w) = s.
\end{aligned}
$$

Say $\mathrm{syn}(L_\Gamma) = (S_\Gamma, B_\Gamma, E_\Gamma)$. We first prove that

▶ **Lemma 62.** *For every $v_1, v_2 \in \{0,1\}^k$, $\gamma(v_1) = \gamma(v_2)$ iff $\eta_3(v_1) = \eta_3(v_2)$.*

Proof. Assume that $\gamma(v_1) = \gamma(v_2)$. Let $x, y \in (\{0,1\}^k)^+$ be arbitrary.

$$
\begin{aligned}
xv_1y \in L_\Gamma \text{ iff } & \gamma^*(xv_1y) = s && \text{by (2)} \\
\text{iff } & \gamma^*(x)\gamma(v_1)\gamma^*(y) = s && \text{by definition} \\
\text{iff } & \gamma^*(x)\gamma(v_2)\gamma^*(y) = s && \text{since } \gamma(v_1) = \gamma(v_2) \\
\text{iff } & \gamma^*(xv_2y) = s && \text{by definition} \\
\text{iff } & xv_2y \in L_\Gamma && \text{by (2)}
\end{aligned}
$$

Thus, $v_1 \sim_{L_\Gamma} v_2$ so $\eta_3(v_1) = \eta_3(v_2)$.

Assume that $\eta_3(v_1) = \eta_3(v_2)$. Thus, $v_1 \sim_{L_\Gamma} v_2$ so for every $x, y \in (\{0,1\}^k)^+$, $xv_1y \in L_\Gamma$ iff $xv_2y \in L_\Gamma$. Therefore, for every $x, y \in (\{0,1\}^k)^+$,

$$
\begin{aligned}
\gamma^*(x)\gamma(v_1)\gamma^*(y) = s \text{ iff } & \gamma^*(xv_1y) = s && \text{by definition} \\
\text{iff } & xv_1y \in L_\Gamma && \text{by (2)} \\
\text{iff } & xv_2y \in L_\Gamma && \text{by the above} \\
\text{iff } & \gamma^*(xv_2y) = s && \text{by (2)} \\
\text{iff } & \gamma^*(x)\gamma(v_2)\gamma^*(y) = s && \text{by definition}
\end{aligned}
$$

Because S_5 is a group, it is cancellative (cf. Proposition 14); thus, $\gamma(v_1) = \gamma(v_2)$.

We have now shown that $\gamma(v_1) = \gamma(v_2)$ iff $\eta_3(v_1) = \eta_3(v_2)$. ◀

We now construct a typed isomorphism $f = (f_1, f_2, f_3)$ from T to $\mathrm{syn}(L_\Gamma)$. We start with f_3.

For each $t \in S_5$, let $f_3(t) = \eta_3(w)$ for some $w \in \gamma^{-1}(t)$; by Lemma 62, the specific choice of $w \in \gamma^{-1}(t)$ does not matter.

We now prove that f_3 is injective. Let $s_1, s_2 \in S_5$ and assume that $f_3(s_1) = f_3(s_2)$. Then, by construction of f_3, there exists $w_1 \in \gamma^{-1}(s_1)$ and $w_2 \in \gamma^{-1}(s_2)$ such that $f_3(s_1) = \eta_3(w_1) = \eta_3(w_2) = f_3(s_2)$. By Lemma 62, $\gamma(w_1) = \gamma(w_2)$, so $s_1 = s_2$ since $s_i \in \gamma^{-1}(s_i)$.

We now prove that f_3 is surjective. Let $v \in E_\Gamma \subseteq S_\Gamma$ be arbitrary. Because η is the syntactic morphism to $\mathrm{syn}(L_\Gamma)$, it is surjective; therefore, $\eta_3(\{0,1\}^k) = E_\Gamma$ so there exists $w \in \{0,1\}^k$ such that $\eta_3(w) = v$. Let $t = \gamma(w)$. By construction of f_3 and Lemma 62, $f_3(t) = \eta_3(w) = v$ so f_3 is surjective.

Therefore, f_3 is a bijection from S_5 to E_Γ.

Let f_1 be the homomorphism induced by f_3. The proof of f_1's bijectivity is analogous to the proof of f_3's bijectivity. Thus, f_3 is a isomorphism from S_5 to S_Γ.

We now must construct and show that $f_2 : \{\varnothing, \{s\}, S_5 - \{s\}, S_5\} \to B_\Gamma$ is an isomorphism of Boolean algebras. B_Γ will only have four elements—$\varnothing$, $X = \eta_1(L_\Gamma)$, $S_\Gamma - X$, and S_Γ—since $(S_\Gamma, B_\Gamma, E_\Gamma)$ is a syntactic typed semigroup. Let $f_2(\varnothing) = \varnothing$, $f_2(\{s\}) = X$, $f_2(S_5 - \{s\}) = S_\Gamma - X$, and $f_2(S_5) = S_\Gamma$. f_2 is clearly bijective and preserves the Boolean algebra structure.

Lastly, we must prove that $f = (f_1, f_2, f_3)$ is actually a typed homomorphism by proving that $f_1(\{s\}) = f_2(\{s\}) \cap f_1(S_5)$. (The other condition on Definition 39 is trivially satisfied.)

We know that for an element $g \in (\{0,1\})^+$,

$$\eta_1(g) \in \eta_1(L_\Gamma) \text{ iff } \gamma^*(g) = s \qquad (\star)$$

We first prove that $f_1(\{s\}) \subseteq f_2(\{s\}) \cap f_1(S_5)$. Since $s \in S_5$, $f_1(\{s\}) \subseteq f_1(S_5)$. We know $f_1(\{s\}) = \{f_1(s)\}$ by definition so we must show that $f_1(s) \in f_2(\{s\}) = X = \eta_1(L_\Gamma)$. Since $f_1(s) = \eta_3(g) = \eta_1(g)$ for some $g \in \gamma^{-1}(s)$, $\gamma^*(g) = s$ so, by $(\star)$, $\eta_1(g) \in \eta_1(L_\Gamma)$ so $f_1(s) \in \eta_1(L_\Gamma)$.

We now prove that $f_2(\{s\}) \cap f_1(S_5) \subseteq f_1(\{s\})$. Let $\eta_1(g) \in f_2(\{s\}) \cap f_1(S_5)$ be arbitrary. Since $f_2(\{s\}) = \eta_1(L_\Gamma)$, by $(\star)$, $\gamma^*(g) = s$. Therefore, by construction of f_1, $\eta_1(g) = f_1(s) \in f_1(\{s\}) = \{f_1(s)\}$.

All together, we get that f is a typed isomorphism from T to $\mathrm{syn}(L_\Gamma)$ so the typed quantifier semigroup of $\Gamma^{S_5,s}_{1,\gamma}$ is isomorphic to (S_5, s, S_5). ◀

We also know the following from the literature:

▶ **Lemma 63.**
- **(i)** DLogTime-uniform NC^1 *can compute majority.*
- **(ii)** *The quantifiers in* FO *are definable in* $(\mathrm{Maj})[<]$. *([12, Theorem 3.2])*
- **(iii)** *The numerical predicate* $+$ *is definable in* $(\mathrm{Maj})[<]$. *([12, Theorem 4.1])*
- **(iv)** *The numerical predicate* $\times$ *is definable in* $(\{\mathrm{Maj}, \mathrm{Sq}\})[<]$ *and* Sq *is definable in* $(\mathrm{Maj})[<, +, \times]$. *(cf. [15, Theorem 2.3.f] and [11, Section 2.3])*

and, all together, we get our main result:

▶ **Theorem 64.**

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

Proof. Let $\delta : \{0,1\}^c \to S_5$ be as it was defined in Lemma 53.

$$\begin{aligned}
\text{DLogTime-uniform } \mathrm{NC}^1 &= \mathcal{L}((\mathrm{FO} \cup \Gamma^{S_5})[+, \times]) \text{ via [2]} \\
&= \mathcal{L}((\mathrm{FO} \cup \Gamma^{S_5}_{1,\delta})[+, \times]) \text{ via Corollary 58} \\
&= \mathcal{L}((\Gamma^{S_5}_{1,\delta} \cup \{\mathrm{Maj}, \mathrm{Sq}\})[<]) \text{ via Lemma 63} \\
&= \mathcal{L}(\mathrm{sbpc}_<(\{(\mathbb{Z}, \mathbb{Z}^+, \pm 1), (\mathbb{N}, \mathbb{S}, \{0,1\})\} \\
&\qquad\qquad \cup \{(S_5, s, S_5) \mid s \in S_5\})) \\
&\quad \text{via Theorems 52 and 61} \\
&= \mathcal{L}(\mathrm{sbpc}_<(\{(\mathbb{Z}, \mathbb{Z}^+, \pm 1), (\mathbb{N}, \mathbb{S}, \{0,1\}), (S_5, \wp(S_5), S_5)\})) \\
&\quad \text{since } \forall s \in S_5, (S_5, s, S_5) \preceq (S_5, \wp(S_5), S_5) \\
&\quad \text{and } \mathcal{L}((S_5, \wp(S_5), S_5)) \subseteq \mathrm{Reg} \subseteq \mathrm{ALogTime}
\end{aligned}$$

◀

5 Conclusion

[TODO]:

References

1 Scott Aaronson and Avi Wigderson. Algebrization: A new barrier in complexity theory. *ACM Transactions on Computation Theory (TOCT)*, 1(1):1–54, 2009.

2 David A Mix Barrington, Neil Immerman, and Howard Straubing. On uniformity within NC1. *Journal of Computer and System Sciences*, 41(3):274–306, 1990.

3 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.

4 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.

5 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.

6 Stephen A Cook. Characterizations of Pushdown Machines in Terms of Time-Bounded Computers. *Journal of the ACM (JACM)*, 18(1):4–18, 1971.

7 Samuel Eilenberg. *Automata, Languages, and Machines (Vol. B)*. Academic Press, 1976.

8 Ronald Fagin. Generalized first-order spectra and polynomial-time recognizable sets. *Complexity of computation*, 7:43–73, 1974.

9 Fred C Hennie. One-tape, off-line turing machine computations. *Information and Control*, 8(6):553–578, 1965.

10 Andreas Krebs. *Typed semigroups, majority logic, and threshold circuits*. PhD thesis, Tübingen, Univ., Diss., 2008, 2008.

11 Andreas Krebs, Klaus-Jörn Lange, and Stephanie Reifferscheid. Characterizing TC0 in terms of infinite groups. *Theory of Computing Systems*, 40(4):303–325, 2007.

12 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.

13 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.

14 John Rhodes and Bret Tilson. The kernel of monoid morphisms. *J. Pure Appl. Algebra*, 62(3):227–268, 1989.

15 Nicole Schweikardt. *On the Expressive Power of First-order Logic with Built in Predicates*. Logos-Verlag, 2002.

16 Avi Wigderson. *Mathematics and computation: A theory revolutionizing technology and science*. Princeton University Press, 2019.

A Strong Block Product Closure

A.1 Weakly Closed Classes

▶ **Definition 65** (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 66** (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 67** (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 68** (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 69** (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.[6] We now build up to its definition:

▶ **Definition 70** (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 71** (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* $(m_1,n_1),(m_2,n_2)\in M\times N$,

$$(m_1,n_1)\circ(m_2,n_2)=(m_1n_2+n_1m_2,n_1\cdot n_2).$$

▶ **Definition 72** (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* $+$:

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

[6] Historically, the "*wreath product*" was first used for this purpose. Since [14], however, the block product has been the preferred and easier-to-work-with tool of choice.

- *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*

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

A.3 The Typed Block Product

▶ **Definition 73** (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) *$T \leq S \Box S'$ 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 74.** *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 75.** *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*

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