# Numerical Semigroups and the Cayley Semigroup Membership Problem

Jacob Urisman

David A. Mix Barrington, Eric Sommers

## Contents

# 1 Abstract

The Cayley Semigroup Membership problem is a bridge between the world of classical space complexity and semigroup theory. The problem was first found to be NL-complete, after which more specific variants of the problem were gradually determined to be in L using descriptive complexity, circuit complexity, and in this paper, semigroup theory. With help from Dr. Pierre McKenzie, we show that the problem, when restricted to groups, is in L. We investigate the problem for the simplest type of commutative semigroup: monogenic semigroups, and find L equivalence between monogenic semigroups and numerical semigroups. We also connect both variants to the classical algorithmic complexity problem subset sum. From there, we are able to place the problem for monogenic semigroups in L.

# 2 Introduction and Basic Definitions

The goal of this paper is to find new possible methods of solving the classical complexity theory open problem of L vs NL by drawing connections to abstract algebra, specifically semigroup theory. Our hope is to attract pure mathematicians to the field of complexity theory to see if they can offer new insights to the long standing open problem. Many mathematicians have likely heard of complexity theory as a field of theoretical computer science, but it is important to make a few distinctions about the nature of the problem in this paper. First and foremost, this problem is dealing with space complexity. It is common to hear about complexity theory from the angle of time complexity, as P vs NP is a very famous time complexity problem. L vs NL is a similar problem, but instead of caring about how long a computer would need to run to solve a problem, we care about how much space the computer needs to work out the problem. For the following definitions, we assume the reader is familiar with Turing machines and big-O notation.

## 2.1 Space Complexity

**Definition 1.** Given a Turing machine with an $n$ bit input on a read only input tape, a blank read-write work tape, and a blank write only output tape, we say a problem has an $O(f(n))$ *space complexity* if the Turing machine needs $O(f(n))$ bits of the work tape to solve it.

*Remark* 1. For the rest of the paper, the reader may assume that all problems discussed are decision problems, i.e. problems where the answer is "yes" or "no".

**Definition 2.** A problem is in the complexity class of *deterministic logspace* or L if there exists an algorithm to solve using $O(\log n)$ bits of the work tape.

**Definition 3.** A problem is in the complexity class of *nondeterministic logspace* or NL if given an input $x$, there exists an algorithm $A$ (often referred to as a verifier or certifier) and a certificate $c$ (usually a possible solution to the problem on that input), such that $A$ when given $x$ and $c$ checks whether $c$ is a valid solution to the problem in $O(\log n)$ space.

*Note.* "Possible solution" in the context of the certificate means the solution that would be output if the problem was not a decision problem.

The L vs NL question asks whether the two complexity classes are actually the same. In layman's terms the question is "is the set of problems that can be solved when using space efficiently and the set of problems that can be verified when using space efficiently the same set?" The answer seems intuitively to be no, but proving that a problem *cannot* be solved using a certain amount of space, i.e. an algorithm solving the problem must use more than a certain amount of space, is very difficult.

## 2.2 Reductions

The main proof technique when examining the complexity of a problem is to compare it to problem with a known complexity. Say we are examining problem $X$ and we know the complexity of problem $Y$.

**Definition 4.** To show that $X$ is at most as hard as $Y$, we construct a *reduction*, which is a function $f : X \to Y$ such that for every $x \in X$, $f(x)$ has " yes" as an answer if and only if $x$ has "yes" as an answer.

*Note.* $f$ must be a deterministic algorithm and cannot be more complex than solving $f(x)$.

*Remark* 2. The intuition behind reductions is that even if $X$ seems harder than $Y$, one would always just convert $X$ to $Y$ if no easier method is known. This is why we write $X \leq Y$, read as $X$ reduces to $Y$, because $Y$ is as hard or harder than $X$.

*Note.* To signify the complexity of the reduction, i.e. how long it takes to run $f$, we place a subscript on the $\leq$.

## 2.3 Numerical Semigroups

Numerical semigroups have been the subject of recent research by a group of mathematicians at the University of Granada. [Gar06] They are a fairly

simple algebraic object and are easily accessible to any mathematician that has taken a course in abstract algebra. The problem of numerical semigroup membership, which we will define later, is even reminiscent of a standard number theory problem given to students: the Coin Problem (also called the Frobenius Problem and the Postage Stamp Problem). The accessibility of problems regarding numerical semigroups makes them ideal for a wide range of mathematicians to investigate.

# 3 Prior Research: BKLM 2001

[BKLM01]

## 3.1 Problem Definition

We first begin by investigating the complexity of two group theoretic problems.

**Definition 5.**
*Given*: group $G$ as a multiplication table, and some property.
*Question*: does $G$ have that property?

**Definition 6** (Cayley Group Membership)**.**
*Given*: group $G$ as a multiplication table, $X \subseteq G$, $t \in G$.
*Question*: Is $t \in \langle X \rangle$?

CGM for a groupoid was shown to be P-complete [JL74]. CGM for a semigroup was shown to be NL-complete [JLL76]. Barrington and McKenzie investigated the problem in 1991 [BM91].

## 3.2 Descriptive Complexity and First-order Definitions

It is common in complexity theory to analyze complexity classes given by using first-order logic with a constraint on the length of the quantifier blocks and what kinds of predicates we are allowed. In this setting, the input is a finite structure and the elements of the structure are the domain. For example, the input might be a bit string and the elements are the positions of the string. Regardless of structure, we use predicates to test properties of the structure. For example, given an input representing a graph, we might use the predicate $E(x, y)$ to refer to an edge between nodes $x$ and $y$. The predicate is true if and only if an edge exists. This lens of complexity theory is called descriptive complexity and is outlined well in [Imm98]. Descriptive

complexity relates neatly to classical complexity classes. For example, first-order logic defines the class $AC^0$, or the class of problems solveable by circuit families of polynomial size with constant depth and unbounded fan-in. In fact, a first-order formula can express a property if and only if it can be tested by a logtime uniform circuit family of polynomial size, unbounded fan-in, and $O(d(n))$ depth where $d(n)$ is the quantifier depth of the formula. For example, $AC^1$ is equivalent to $\mathrm{FO}(\log n)$.

### 3.2.1 Defining FOLL

The paper introduces a new complexity class FOLL, which stands for $\mathrm{FO}(\log \log n)$. In other words, the class of problems solveable by logtime uniform circuit families of polynomial size, unbounded fan-in, and $O(\log \log n)$ depth. It is also important to note that FOLL is closed under FO reductions.

### 3.2.2 Where is FOLL on the Inclusion Chain

FOLL is only known to include $AC^0$ from the classical circuit complexity classes. $AC^1$ seems to be the smallest classical complexity class not included in FOLL. Furthermore, from the Smolensky circuit lower bound [Smo93], we know that any problem in FOLL is not hard for any class containing PARITY.

## 3.3 Results

### 3.3.1 Main Proof Technique

A helpful method when working with formulae with iterated quantifiers, is to use a recursive definition of a predicate, and show that it will always terminate after a certain number of steps. For example, $a = b^i$ for $a, b \in G$ can be checked for any $i$ up to $n$. We exploit the inductive nature of the power predicate: $P(a, b, i) := a = b^i$. We have two base cases $P(a, b, 0) \iff a = e$, and $P(a, b, 1) \iff a = b$. From here, we can write two inductive cases, which are $P(a, b, i) \to \exists j, k, c, d : (i = j + k) \wedge (c = b^j) \wedge (d = b^k) \wedge (a = cd)$, and $P(a, b, i) \to \exists j, k, c : (i = jk) \wedge (c = b^j) \wedge (a = c^k)$.

Now, we shall explain why this definition closes in $O(\log \log n)$ steps so long as $i \leq n$. If $i$ is a power of two, we can write $i = jk$ for a $j$ and $k$ that we choose such that $j$ and $k$ are powers of two and have logs at most half that of $i$. If $i$ is not a power of two, we can write $i = j + k$ for a $j$ and $k$ that we choose such that $j$ and $k$ each have at most half as many ones in their binary expansions as $i$. Since both $\log i$ and the number of ones in the

binary expansion of $i$ are bounded by $\log \log n$, we can reach the base cases in at most $2 \log \log n$ steps. Furthermore, we can define it using FO. Thus, $\forall a, b \in G, i \leq n : a = b^i \in$ FOLL. It is important to note that this predicate is also in L. We simply compute $b^i$ for all $i \leq n$ and compare each one to $a$.

### 3.3.2 Group Properties

Here are a few results from the paper regarding testing group properties.

**Theorem 1.** *Testing a Cayley Group for cyclicity is in FOLL ∩ L.*

*Proof.* A cyclic group is defined as follows: $\exists g \forall a \exists i : a = g^i$, which is checkable using the power predicate, which is in FOLL ∩ L. □

**Theorem 2.** *Testing a Cayley Group for nilpotency is in FOLL ∩ L.*

*Proof.*

**Lemma 1.** *A finite group $G$ is nilpotent if and only if the set of $p$-elements for every prime $p$ divides $|G|$ is a group.*

*Proof.* There is a proof of theorem 5.39 in [Rot99] and theorem 3 in [DF03] which states that a finite group $G$ is nilpotent if and only if it is the direct product $\Pi$ of Sylow $p$-groups. So, for each prime $p$ divides $|G|$, the $p$-elements of $G$ are exactly the elements whose direct product components outside the $p$-groups are trivial. Therefore, the set of $p$-elements is a group, isomorphic to the direct product of the $p$-groups in $\Pi$.

The converse is slightly simpler. A finite group $G$ with a unique maximal $p$-subgroup for every $p$ divides $|G|$, is the direct product of those maximal subgroups. Thus, if it forms a group, it must be the unique maximal $p$-subgroup of $G$. If this is true for all $p$ divides $|G|$, then $G$ is nilpotent. □

For any $p$, we can define a $p$-element in first-order as follows:

$$o(a) = m \iff (a^m = e) \wedge \forall j : 0 < j < m \rightarrow a^j \neq e$$

This all relies on the power predicate, and is thus in FOLL ∩ L. However, we also need to check that $p$ is a prime smaller than $m$, $m \leq n$, and no prime other than $p$ divides $m$. This can all be done in first-order. Thus, our nilpotency test simply follows from the above lemma. □

### 3.3.3 Cayley Group Membership

**Theorem 3.** *CGM(cyclic groups) is in FOLL ∩ L.*

*Proof.* We simply need to check which generator in $\langle X \rangle$ is the greatest common divisor $i$ such that $g^i = h$ for all $h \in X$. Then, for the $j$ such that $g^j = t$, we just need to check that $i$ divides $j$. □

**Theorem 4.** *CGM(abelian) is in FOLL.*

*Proof.* In order to prove this, we will need the following lemma.

**Lemma 2.** *Abelian groups have the* $\log n$ *power basis property, meaning that for a group G, it is abelian if any set of generators X for G, any element of G is a product of at most* $\log n$ *powers of elements of X.*

*Proof.* Take an arbitrary set of generators for $G$, called $B = \{b_1, \ldots, b_k\}$. Now let $G_i$ be the set of elements generated by $\{b_1, \ldots, b_i\}$. Set $X$ to be all $b_i$ such that $G_i$ and $G_{i-1}$ are different. It is fairly straightforward to see that $X$ is the minimal set of generators for $G$. To complete the proof, note that $G_i$ will have at least twice as many elements as $G_{i-1}$, thus $|X|$ is at most $\log n$. □

Consider the following sets.

$$Y_0 = \{x^i : x \in X, i \leq n\}$$

$$Y_{i+1} = \{yz : y, z \in Y_i\}$$

Since $G$ has the $\log n$ power basis property, all we need to do is check that $t \in Y_{\log \log n}$. Finally, since membership in that group is FO definable, this problem is in FOLL. □

*Note.* The above problem was not known to be in L at the time of the paper.

**Theorem 5.** *CGM for solvable groups of solvability class* $d(n)$ *is in FO(*$d(n) \log \log n$*). Specifically, FOLL if* $d(n) = O(1)$*.*

*Proof.* Solvability of a group is defined in terms of a group's derived series, wherein we take successive commutator subgroups and a group is solveable if and only if the series of commutator subgroups eventually terminates in the trivial group. The number of times $d$ we take commutators is the solvability class of $G$.

The authors extend the method for CGM(abelian) wherein we close the current generating set under powering and then under products, but this time,

do it $d$ times. Let us prove this inductively. Our base case is CGM(abelian). Assume that CGM($H$) is FO($(d(n) - 1)(\log \log n)$). We shall prove that CGM($G$) is FO($d(n)(\log \log n)$). Let $H$ be the commutator subgroup of $G$. We can generate all elements of $G$ from $H$ as follows. Every element of $G$ is a product of some element of a coset of $G/H$ and an element of $H$. Since the elements $xH$ for every $x \in X$ generates $G/H$. This means we can find an element of every coset of $G/H$ by taking one round of closure under powers and $O(\log n)$ length products (which takes $O(\log \log n)$ time by CGM(abelian)), and we are done. □

**Theorem 6.** *This theorem is due to [Thé80]. Any nilpotent group of order $n$ has solvability class $O(\log \log n)$.*

Thus, CGM for nilpotent groups is in FO($(\log \log n)^2$) and thus not hard for any class that contains PARITY.

### 3.4 The Division Problem

This paper played a seminal role in understanding the complexity of three problems, with one of specific interest. *Iterated multiplication* of $n$ $n$-bit integers, *powering* of an $n$-bit integer by a $\log n$-bit exponent, and most interestingly, *division* of one $n$-bit integer by another. In 2000, Chiu, Davida, and Litow [CDL01] constructed an L-uniform circuit family which placed these three problems in L. However, analysis of that family equated it to a first order formula with majority quantifiers and only one additional predicate, that being the powering predicate. The results in this paper looked into the complexity of this predicate. This added power from the powering predicate allows for iterated multiplication of numbers getting output into Chinese remainder representation. Thus, the complexity of the three problems is equivalent to converting a number from Chinese remainder form to binary form. This problem was solved only two years later by William Hesse, which culminated in the result that $n$-bit integer division is in fully uniform $TC^0$ [HAB02].

## 4 Prior Research: Fleischer 2018

[Fle18]

### 4.1 Problem Definition

Similar to the Cayley Group Membership problem, the Cayley Semigroup Membership problem is defined as follows.

**Definition 7** (Cayley Semigroup Membership)**.**
*Given*: semigroup $S$ as a multiplication table, $X \subseteq S$, $t \in S$.
*Question*: Is $t \in \langle X \rangle$?

The paper analyzed the complexity of this problem for various *varieties* of semigroups. The motivation for this is firstly the direct connection between this problem and regular languages, and secondly the goal of better understanding the connection between algebra and complexity theory. As stated in the previous chapter, previous work of note includes [JL74],[JLL76],[BM91],and [BKLM01].

Let us define several *varieties* of semigroups that we will be dealing with.

**Definition 8. G**: The class of all finite groups

**Definition 9. Ab**: The class of all finite abelian groups

**Definition 10. Com**: The class of all commutative semigroups

**Definition 11. N**: The class of finite nilpotent semigroups

Let us also define the notion of a *join*.

**Definition 12.** The *join* of two varieties **V** and **W**, denoted as **V**$\vee$**W** is the smallest variety containing both **V** and **W**.

This paper places various varieties of semigroups into the quasipolynomial size circuit class $qAC^0$. [Bar92] surveys the complexity of quasipolynomial size circuits, or circuits whose size is $O(2^{\log^c n})$ where $n$ is the size of the input and $c$ is a constant. $qAC^0$ is important because like FOLL, anything in $qAC^0$ cannot be hard for L or any higher class.

## 4.2 Main Results

### 4.2.1 Cayley Circuits

**Definition 13.** We define *Cayley circuits* as non-classical circuits where each vertex has in-degree 0 or 2. In-degree 0 vertices are input gates, and in-degree 2 vertices are product gates. Let there be a topologically maximal product gate called the output gate. The size of a circuit $\mathcal{C}$ is denoted by $|\mathcal{C}|$. An input to such a circuit is a list of $k$ elements $x_1 \dots x_k$. Parenthesization does not matter as semigroups are associative.

**Theorem 7.** *Let $\mathcal{C}$ be a Cayley circuit of size $m$. $\mathcal{C}$ can be simulated by an $AC^0$ circuit of size $n^m$.*

*Proof.* The input is $O((|S|^2 + k) \log n)$ bits. For each gate $i \in \{1, \ldots, m\}$ of $\mathcal{C}$, we add $\lceil \log |S| \rceil$ incoming wires to an AND gate (which checks if a fixed vector $(y_1, \ldots, y_m)$ is a valid sequence in $\mathcal{C}$ under the given inputs): if the $i$-th gate of $\mathcal{C}$ is an input gate, we feed the bits of the corresponding input value into the AND gate, complementing the $j$-th bit if the $j$-th bit of $y_i$ is 0. If the $i$-th gate is a product gate and has incoming wires from gates $l$ and $r$, we connect the entry $(y_l, y_r)$ of the multiplication table to the AND gate, again complementing bits corresponding to 0-bits of $y_i$. To obtain a Boolean circuit simulating $\mathcal{C}$, we put such AND gates for all vectors in parallel. In a second layer, we create $\lceil \log |S| \rceil$ OR gates and connect the AND gate for a vector to the $j$-th OR gate if and only if the $j$-th bit of $y_m$ is one. The idea is that exactly one of the AND gates (the gate corresponding to the vector of correct guesses of the gate values of $\mathcal{C}$) evaluates to 1 and the corresponding output value ym then occurs as output value of the OR gates. This circuit has depth 2 and size $|S|^m + \lceil \log |S| \rceil$. □

### 4.2.2 Poly-Logarithmic Circuits

As seen in the previous chapter, [BKLM01] introduced the logarithmic power basis property. Fleischer builds on this and introduce a new property called the poly-logarithmic circuits property. He then proves that both **Com** and **G** have this property.

**Lemma 3.** *The variety **Com** has the logarithmic power basis property.*

*Proof.* Let $t = x_1^{i_1}, \ldots, x_k^{i_k}$ for $i_n \in \mathbb{N}$. Assume for the sake of contradiction that $k > \log |S|$. The power set $\mathcal{P}(\{1, \ldots, k\})$ forms a semigroup under the set union operation. Consider the morphism $h : \mathcal{P}(\{1, \ldots, k\}) \to S$ defined by $h(\{j\}) = x_j^{i_j}$ for all $j \in \{1, \ldots, k\}$. This morphism is well-defined since $S$ is commutative. This means that $|\mathcal{P}(\{1, \ldots, k\})| = 2^k > 2^{\log |S|} = |S|$. However, by the pigeonhole principle, there exists some element which is not needed in the product $x_1^{i_1}, \ldots, x_k^{i_k}$ to produce $t$, which is a contradiction. □

**Proposition 1.** *Let **V** be a class of semigroups which is closed under sub-semigroups and has the logarithmic power basis property. Then **V** has the poly-logarithmic circuits property.*

*Proof.* We can construct Cayley circuits $\mathcal{C}_1, \ldots, \mathcal{C}_k$ of size at most $2\lceil \log |S| \rceil$. We can then use an additional $k-1$ product gates to combine each circuit into a single circuit with $k \cdot 2\lceil \log |S| \rceil + k - 1 < 5 \log^2 |S|$ gates since $k \leq \log |S|$. □

**Definition 14.** A *straight line program* over a finite group $G$ and a subset $X$ is a sequence of elements $(g_1, \ldots, g_l)$ where each $g_i \in X$ or $g_i = g_p^{-1}$ or $g_i = g_p g_q$ for some $p, q < i$. $l$ is called the length.

**Lemma 4.** *Reachability Lemma Let $G$ be a finite group and let $X$ be a set of generators for $G$. Then, for each element $t \in G$, there exists a straight-line program over $X$ generating $t$ of length at most $(\log |G| + 1)^2$. Proof due to [Bab91].*

*Proof.* We create sets of increasing size $C_i = \{h_1^{e_1} h_2^{e^2} \cdots h_i^{e_i} \mid e_j \in \{0, 1\}\}$. Let $C_0 = \{1\}$. We then find an $h_{i+1} \in C_i^{-1} C_i X$ such that $C_i \cap C_i h_{i+1} = \emptyset$. If no such $h_{i+1}$ is found, our $C_i$ is our straight line program. Let $i = t$ at termination. $C_{i+1} = C_i \cup C_i h_i$ by definition, so $t \leq \log |G|$. Furthermore, if there is no such $h_{i+1}$, then $C_i \cap C_i h_{i+1} \neq \emptyset$, meaning every element was in $C_i$, making $C_i^{-1} C_i X \subseteq C_i^{-1} C_i$, and therefore $G = C_i^{-1} C_i X^N \subseteq C_i^{-1} C_i$ making $C_i^{-1} C_i = G$. Thus, every element of $G$ can be written as $x^{-1} y$ for $x, y \in C_t$. Finally, the cost of adding $h_{i+1}$ to $X \cup h_1, \ldots, h_i$ is $\leq (2i - 1)$ making the total cost of the straight line program $\leq \sum_{i=1}^{t} (2i - 1) = t^2 \leq (\log |G|)^2$ □

**Lemma 5.** *The variety $\boldsymbol{G}$ has the poly-logarithmic circuits property.*

*Proof.* By the Reachability Lemma, there exists a straight line program for any finite group with $l \leq (\log |G| + 1)^2$. We construct a Cayley circuit $\mathcal{C}$ by adding one of three types of gates for each element in the straight line program. If $g_i \in X$, we add a new input gate and let the next element in the sequence $x_k = g_i$. If $g_i = g_p g_q$, then we add a new product gate. If $g_i = g_p^{-1}$, we can add a Cayley circuit $\mathcal{C}'$ with maximum $2\lceil \log |G| \rceil$ gates. The resulting circuit will have a maximum of $(\log |G| + 1)^2 \cdot 2\lceil \log |G| \rceil \leq 2(\log |G| + 1)^3$ gates. □

**Theorem 8.** *Let $\boldsymbol{V}$ be a class of semigroups with the poly-logarithmic circuits property. Then CSM($\boldsymbol{V}$) is in $qAC^0$.*

*Proof.* Since $\mathbf{V}$ has the poly-logarithmic circuits property, we know that, for some constant $c$, the element $t$ is in the subsemigroup generated by $X$ if and only if there exist a Cayley circuit $\mathcal{C}$ of size $\log^c n$ and inputs $x_1, \ldots, x_k \in X$ such that $\mathcal{C}(S, x_1, \ldots, x_k) = t$. There are at most $(\log^c n \cdot \log^c n)^{\log^c n} = 2^{\log^c n \log(2c \log n)}$ different Cayley circuits of this size. For any of these Cayley circuits, by Proposition 3, there exists an $AC^0$ circuit of size $n^{\log^c n} = 2^{\log^{c+1} n}$ which simulates it. There are at most $n^k \leq n^{\log^c n} = 2^{\log^{c+1} n}$ possibilities of connecting (not necessarily all) input gates corresponding to the elements of $X$ to this simulation circuit. Thus, we can check for all Cayley circuits of the given size and all possible input assignments in parallel, whether the value of the corresponding circuit is $t$, and feed the results of all these checks into a single OR gate to obtain a quasi-polynomial-size Boolean circuit. □

### 4.2.3 Looking at the Complexity of Some Varieties

Fascinatingly, while CSM(**G**) and CSM(**Com**) are both in $qAC^0$, CSM(**N**)$\subseteq$CSM(**G**$\vee$**Com**) is NL-complete [JLL76]. Fleischer refers to this as "strange complexity". What this means is that for any well defined variety of semigroup that is not NL-complete, it will likely not be complete for any class that contains parity. Furthermore, since we now know that CSM(**G**) is in L, there is likely no variety that we can find that is L-complete.

# 5 New Research

## 5.1 An Observation

As Dr. Pierre McKenzie recently pointed out to us, undirected graph reachability is reducible to the Cayley Group Membership problem for an arbitrary finite group, putting the entirety of CGM in L.

*Remark* 3. CSM(**G**) $\leq_L$ UReach

*Proof.* Let $G, X, t$ be an arbitrary input to CGM as defined above. We construct an undirected graph with nodes for each element $a \in G$. Then, $\forall y \in G, x \in X$ we add the undirected edge $(y, z)$ where $z = yx$. Let $s$ be the node constructed from the identity element and let $t$ be the node constructed from the target element.

Now all that's left to check is that if an edge is able to be taken, we must be able to take it back, which corresponds to inverses of every $x \in X$ being able to be generated by $X$. We exploit the property of groups that $x^{-1} = x^{n-1}$ where $n = |G|$ to see that the inverse of every $x \in X$ is able to be generated by using only $x$.

Thus, if there is a path from $s$ to $t$, then we can write $t$ as a product of each edge taken to reach it, making it able to be generated by $X$. Conversely, if $t$ can be written as a product of elements of $X$, we simply follow edges associated with each element of the product to reach $t$ from $s$. □

This observation, coupled with the 2004 result that undirected graph reachability is in L [Rei08] shows that CSM(**G**) is in L.

## 5.2 Results

### 5.2.1 Numerical Semigroups Again

The following algebraic object will be our central focus for the rest of this paper, as it is simple to work with and there is a substantial amount of prior research on it.

**Definition 15.** A set $S \subset \mathbb{N}$ is a *numerical semigroup* if

1. $0 \in S$
2. $S$ is cofinite
3. $x, y \in S \to x + y \in S$

*Remark* 4. Note that it is simple to construct a numerical semigroup by starting with a set of naturals and adding every linear combination of elements in that set. More formally, for a set $\{n_1, \ldots, n_k\}$, if a natural can be represented as $x_1 n_1 + \cdots + x_k n_k$, then it is in $S$.

**Theorem 9.** *CSM(Monogenic) is L-equivalent to CSM(Numerical)*

*Proof.* For the first direction, i.e. CSM(Monogenic) $\leq_{\mathrm{L}}$ CSM (Numerical), let $M$ be an arbitrary monogenic semigroup generated by $g$. Let $m$ be the index of the monogenic semigroup and let $r$ be the period, i.e. the smallest positive integers such that $g^m = g^{m+r}$. Finally, let $j$ be the discrete log of the target element $t$, i.e. $g^j = t$. If $j \geq m$, then we will show that we can check in logspace whether $\langle X \rangle = t$. This case is equivalent to the cyclic group case, so we simply need to take the gcd of the discrete logs of all the elements of $\langle X \rangle$ and $r$ and see if that number divides $j$. To verify that this works, we note that if the target element is in the loop, our generators will generate everything in the loop that is a multiple of the gcd of the discrete logs. However, if the loop is coprime to the gcd of the generators, then everything in the loop will necessarily be generated. The only thing left to worry about is whether we need to adjust the generators that might be in the loop by $m$, but we need not worry, since addition of two elements still works the same as in the cyclic group case, i.e. $g^x + g^y = g^{x+y}$ up to equivalence in the loop.

Otherwise, we have the case where the target lies in the stem of the semigroup. In this case, it is not immediately obvious how to solve this in logspace. However, it is equivalent to the case where we ignore the cycle of the semigroup, leaving elements 1 to $m-1$. We can rephrase this problem as the case where $M$ now becomes the numerical semigroup $\langle X \rangle$, i.e. the

numerical semigroup generated by our set of elements $X$. If there is a way to solve this problem for a monogenic semigroup in L, then we can use the method for solving the case where the target lies in the stem of the semigroup on a numerical semigroup. Conversely, if there is a way to solve this problem for a numerical semigroup in L, then we can use that method as the way to solve the case where the target lies in the stem of a monogenic semigroup.

Conversely, it is trivially true that CSM(Numerical) $\leq_{\mathrm{L}}$ CSM(Monogenic), as an input to CSM(Numerical) is just the generators of the monogenic semigroup of order $t$ with period 0.

It is important to note with this proof that all of these reductions are Turing reductions. Each of these reductions can be computed on a logspace transducer rather easily. □

### 5.2.2 Subset Sum

Another method of attack for this problem could be one inspired by the subset sum problem. Finding a combination of elements to try to reach a target is simply subset sum with the ability to reuse elements multiple times. Unary subset sum has been proven to be in L [Kan17]. However, the algorithm for solving unary subset sum in L allows our set of elements to be a multiset. This puts CSM(Numerical) in L and by extension CSM(Monogenic).

The algorithm takes advantage of an interesting number theoretic property. If we start with the generating function sum values of subsets: $sum_{S\subseteq[n]}x^{\sum_{i\in S}X_i}$ where $X_i$ is an element of $X$ and $x$ is the standard polynomial variable. With some basic combinatorics, we can turn this sum notation into product notation: $\prod_{i=1}^{|X|}(1+x^{X_i})$. The important thing is that this product is easy to compute in L. From there, taking the sum of these products multiplied by $x^{-t}$ over some prime has an interesting property. If $\sum_{x=1}^{p-1}x^{-t}\prod_{i=1}^{|X|}(1+x^{X_i})\not\equiv 0 \pmod p$ for some prime $p$, then there exists a solution to subset sum. Note that this formula does not tell us anything about what the subset is, only that it exists. Checking this formula over every prime between the length $C$ of the input and $2^C$ ensures that the algorithm outputs the correct answer. The fact that our input is in unary allows this to be in L.

### 5.2.3 Preprocessing and Possible Directions of Future Research

CSM for the simplest case of commutative semigroups, monogenic semigroups, is in L by the two previous theorems. We can extend these results

to CSM for semigroups generated by a constant number of elements.

There is no overarching theorem for finitely generated commutative semigroups as there are with groups. Can we extend these results to all commutative semigroups being in L?

# 6 Conclusion

As stated previously, CSM($\mathbf{G}$) is now known to be in L. Monogenic semigroups are the simplest finite commutative semigroups, being cyclic groups without necessitating the identity element or inverses. Furthermore, now that we have drawn this connection between CSM(Monogenic), CSM(Numerical), and subset sum, this now seems like a problem that can be attacked both from an algorithmic perspective, and a semigroup theoretical perspective.

# 7 Acknowledgements

Thank you to my incredible parents Dr. Anatoly Urisman and Tatiana Urisman, my lovely sibling Hannah Urisman, my amazing grandparents, uncle, aunt, and cousins, and my wonderful girlfriend Ashley Sage Ginzburg for their unconditional love and support. Thank you to Dr. David Mix Barrington for his extraordinary mentorship and advice. Thank you to Dr. Eric Sommers for his help and perspective on this project. Thank you to Dr. Pierre McKenzie for his astute and eye-opening observation. Thank you to Dr. William Hoza for pointing us in the direction of numerical semigroups.

# References

Babai, László. "Local expansion of vertex-transitive graphs and random generation in finite groups". In: *Proceedings of the twenty-third annual ACM symposium on Theory of computing - STOC '91*. ACM Press, 1991. DOI: 10.1145/103418.103440. URL: https://doi.org/10.1145/103418.103440.

Barrington, D.A.M. "Quasipolynomial size circuit classes". In: *[1992] Proceedings of the Seventh Annual Structure in Complexity Theory Conference*. 1992, pp. 86–93. DOI: 10.1109/SCT.1992.215383.

Barrington, David A.Mix and Pierre McKenzie. "Oracle branching programs and Logspace versus P". In: *Information and Computation* 95.1 (Nov. 1991), pp. 96–115. DOI: 10.1016/0890-5401(91)90017-v. URL: https://doi.org/10.1016/0890-5401(91)90017-v.

Barrington, David Mix, Peter Kadau, Klaus-Jörn Lange, and Pierre McKenzie. "On the Complexity of Some Problems on Groups Input as Multiplication Tables". In: *Journal of Computer and System Sciences* 63.2 (Sept. 2001), pp. 186–200. DOI: 10.1006/jcss.2001.1764. URL: https://doi.org/10.1006/jcss.2001.1764.

Chiu, Andrew, George Davida, and Bruce Litow. "Division in logspace-uniform $NC^1$". In: *RAIRO - Theoretical Informatics and Applications* 35.3 (May 2001), pp. 259–275. DOI: 10.1051/ita:2001119. URL: https://doi.org/10.1051/ita:2001119.

Dummit, David S and Richard M Foote. *Abstract Algebra*. en. 3rd ed. Nashville, TN: John Wiley & Sons, June 2003.

Fleischer, Lukas. "On The Complexity of the Cayley Semigroup Membership Problem". In: *CoRR* abs/1802.00659 (2018). arXiv: 1802.00659. URL: http://arxiv.org/abs/1802.00659.

García-Sánchez, PA. *Numerical semigroups mini-course*. 2006. URL: https://www.ugr.es/~pedro/minicurso-porto.

Hesse, William, Eric Allender, and David A. Mix Barrington. "Uniform constant-depth threshold circuits for division and iterated multiplication". In: *Journal of Computer and System Sciences* 65.4 (Dec. 2002), pp. 695–716. DOI: 10.1016/s0022-0000(02)00025-9. URL: https://doi.org/10.1016/s0022-0000(02)00025-9.

Immerman, Neil. *Descriptive Complexity*. en. 1999th ed. Texts in Computer Science. New York, NY: Springer, Nov. 1998.

Jones, Neil D. and William T. Laaser. "Complete Problems for Deterministic Polynomial Time". In: *Proceedings of the Sixth Annual ACM Symposium on Theory of Computing*. STOC '74. Seattle, Washington, USA: Association for Computing Machinery, 1974, pp. 40–46. ISBN: 9781450374231. DOI: 10.1145/800119.803883. URL: https://doi.org/10.1145/800119.803883.

Jones, Neil D., Y. Edmund Lien, and William T. Laaser. "New problems complete for nondeterministic log space". In: *Mathematical Systems Theory* 10.1 (Dec. 1976), pp. 1–17. DOI: 10.1007/bf01683259. URL: https://doi.org/10.1007/bf01683259.

Kane, Daniel M. *Unary Subset-Sum is in Logspace*. 2017. arXiv: 1012.1336 [cs.CC].

Reingold, Omer. "Undirected Connectivity in Log-Space". In: *J. ACM* 55.4 (Sept. 2008). ISSN: 0004-5411. DOI: 10.1145/1391289.1391291. URL: https://doi.org/10.1145/1391289.1391291.

Rotman, Joseph J. *An introduction to the theory of groups*. en. 4th ed. Graduate Texts in Mathematics. New York, NY: Springer, Aug. 1999.

Smolensky, Roman. "On representations by low-degree polynomials". In: *Proceedings of 1993 IEEE 34th Annual Foundations of Computer Science* (1993), pp. 130–138.

Thérien, Denis. "Classification of regular languages by congruences". CS-80-19. PhD thesis. Ontario, Canada: University of Waterloo, 1980.