Space Complexity of Abelian Groups

Douglas Cenzer*
Department of Mathematics, University of Florida,
P.O. Box 118105, Gainesville, Florida 32611
cenzer@math.ufl.edu
Rodney G. Downey,
Department of Mathematics, Victoria University,
Wellington 6140, New Zealand
Rod.Downey@mcs.vuw.ac.nz
Jeffrey B. Remmel†
Department of Mathematics, University of California–San Diego,
La Jolla, CA 92093
jremmel@ucsd.edu
Zia Uddin
Department of Mathematics, University of Wisconsin–Platteville,
Platteville, WI 53818

July 25, 2008

Abstract

We develop a theory of $LOGSPACE$ structures and apply it to construct a number of examples of Abelian Groups which have $LOGSPACE$ presentations. We show that all computable torsion Abelian groups have $LOGSPACE$ presentations and we show that the groups $\mathbb{Z}$, $Z(p^\infty)$, and the additive group of the rationals have $LOGSPACE$ presentations over a standard universe such as the tally representation and the binary representation of the natural numbers. We also study the effective categoricity of such groups. For example, we give conditions are given under which two isomorphic $LOGSPACE$ structures will have a linear space isomorphism.

1 Introduction

This paper continues the study of complexity theoretic model theory and algebra. Complexity theoretic or feasible model theory is the study of resource-bounded structures and isomorphisms and their relation to computable structures and computable isomorphisms. Complexity theoretic model theory and

*Partially supported by NSF grants DMS 0554841, DMS 0532644 and DMS 00652732
†Partially supported by NSF grant DMS 065060.
Keywords: Computability, Complexity Theory, Computable Model Theory

algebra was developed by Nerode, Remmel and Cenzer [16, 17, 18, 4, 5]; see the handbook article [9] for details. Much of the work of those authors focused on polynomial time models. In this paper, we shall develop the theory of $LOGSPACE$ structures and apply it to the study of $LOGSPACE$ Abelian groups.

Complexity theory has been a central theme of computer science and related areas of mathematics since the middle of the last century. Much work has been done on the time complexity of sets, functions and structures. The practical goal is to find efficient algorithms for computing functions and solving problems. In his encyclopedic book [14], Knuth examines in detail the quest for fast multiplication and also considers the problem of radix conversion of numbers between binary and decimal representation. Recent striking advances include the proof that primality is polynomial-time decidable [1] and the result that division of integers can be computed in $LOGSPACE$ [11]. The latter result will be used below in our construction of a $LOGSPACE$ model for the additive group of rationals.

Complexity theoretic model theory is concerned with infinite models whose universe, functions, and relations are in some well known complexity class $\mathcal{C}$ such as polynomial time, exponential time, polynomial space, $LOGSPACE$ etc. That is, let $M_0, M_1, \ldots$ be an effective list of all Turing machines and $\phi_e(x_1, \ldots, x_n)$ denote the function of n-variables computed by the e-th Turing machine M_e. Let

$$\mathcal{A} = (A, \{R_i^{\mathcal{A}}\}_{i\in S}, \{f_i^{\mathcal{A}}\}_{i\in T}, \{c_i^{\mathcal{A}}\}_{i\in U}),$$

be a structure where the universe of $\mathcal{A}$, A, is a subset of $\{0,1\}^*$ and S, T, and U are finite initial segments of of the natural numbers $\mathbb{N}$. Then we say that $\mathcal{A}$ is *computable* if A is a computable subset of $\{0,1\}^*$, each relation $R_i^{\mathcal{A}}$ is computable, and each function $f_i^{\mathcal{A}}$ is computable. For any given complexity class $\mathcal{C}$, we say that $\mathcal{A}$ is $\mathcal{C}$ *structure* if A is in $\mathcal{C}$, each relation $R_i^{\mathcal{A}}$ is in $\mathcal{C}$, and each function $f_i^{\mathcal{A}}$ is in $\mathcal{C}$. If

$$\mathcal{B} = (B, \{R_i^{\mathcal{B}}\}_{i\in S}, \{f_i^{\mathcal{B}}\}_{i\in B}, \{c_i^{\mathcal{B}}\}_{i\in U}),$$

is another $\mathcal{C}$ structure, then we say that $\mathcal{A}$ is $\mathcal{C}$-isomorphic to $\mathcal{B}$ if there is an isomorphism $\Theta : A \to B$ such that both Θ and Θ^{-1} are in $\mathcal{C}$.

By far, the complexity class that has received the most attention is polynomial time. One immediate difference between computable model theory and complexity theoretic model theory is that, in general, it is not the case that all infinite $\mathcal{C}$ sets are $\mathcal{C}$-isomorphic. For example, there is no polynomial isomorphism f with a polynomial time inverse f^{-1} which maps the binary representation of the natural numbers $Bin(\mathbb{N}) = \{0\} \cup \{1\}\{0,1\}^*$ onto the tally representation of the natural numbers $Tal(\mathbb{N}) = \{0\} \cup 1\{1\}^*$. This is in contrast with computable model theory where all infinite computable sets are computably isomorphic, so that one usually only considers computable structures whose universe is the set of natural numbers $\mathbb{N}$.

There are two basic types of questions which have been studied in complexity theoretic model theory. First, there is the basic existence problem, i.e. whether a given infinite computable structure $\mathcal{A}$ is isomorphic or computably isomorphic to a model that lies in a given complexity class $\mathcal{C}$. That is, when we are given a class of structures $\mathcal{S}$ such as a linear orderings, Abelian groups, etc., the following natural questions arise.

(1) Is every computable structure in $\mathcal{S}$ isomorphic to a $\mathcal{C}$ structure?

(2) Is every computable structure in $\mathcal{S}$ computably isomorphic to a $\mathcal{C}$ structure?

For example, Cenzer and Remmel showed in [4] that every computable relational structure is computably isomorphic to a polynomial time structure and that the standard model of arithmetic $(\omega, +, -, \cdot, <, 2^x)$ with addition, subtraction, multiplication, order and the 1-place exponential function is isomorphic to a polynomial time structure. The fundamental effective completeness theorem says that any decidable theory has a decidable model. It follows that any decidable relational theory has a polynomial time model. These results are examples of answers to questions (1) and (2) above. However, one can consider more refined existence questions. For example, we can ask whether a given computable structure $\mathcal{A}$ is isomorphic or computably isomorphic to a polynomial time structure with a standard universe such as the binary representation of the natural numbers, $Bin(\mathbb{N})$, or the tally representation of the natural numbers, $Tal(\mathbb{N})$. That is, when we are given a class of structures $\mathcal{S}$, we can ask the following questions.

(3) Is every computable structure in $\mathcal{S}$ isomorphic to a $\mathcal{C}$ structure with universe $Bin(\mathbb{N})$ or $Tal(\mathbb{N})$?

(4) Is every computable structure in $\mathcal{S}$ computably isomorphic to a $\mathcal{C}$ structure with universe $Bin(\mathbb{N})$ or $Tal(\mathbb{N})$?

It is often the case that when one attempts to answer questions of type (3) and (4) that the contrasts between computable model theory and complexity theoretic model theory become more apparent. For example, Grigorieff [12] proved that every computable linear order is isomorphic to a linear time linear order which has universe $Bin(\mathbb{N})$. However Grigorieff's result can not be improved to the result that every computable linear order is computably isomorphic to a linear time linear order over $Bin(\mathbb{N})$. For example, Cenzer and Remmel [4] proved that for any infinite polynomial time set $A \subseteq \{0,1\}^*$, there exists a computable copy of the linear order $\omega + \omega^*$ which is not computably isomorphic to any polynomial time linear order which has universe A. Here $\omega + \omega^*$ is the order obtained by taking a copy of $\omega = \{0, 1, 2, \ldots\}$ under the usual ordering followed by a copy of the negative integers under the usual ordering.

The main goal of this paper is the study $LOGSPACE$ Abelian groups. It was shown in [5] that there is a family of Abelian p-groups, including the computably categorical p-groups of [20] which are computably isomorphic to polynomial time groups with a standard universe. At the same time, Abelian

p-groups were constructed in [5] which are not computably isomorphic to polynomial time groups with a standard universe. The question of uniqueness of representation, that is, categoricity, was studied further in [6, 8]

It was established by Hopcroft and Ullman [13] that an appropriate model for function calculation is a Turing machine with read-only input and write-only output. The motivation for the input/output approach is that simple functions such as addition can be performed in logarithmic space (in fact in zero space) whereas including the input and/or output would automatically require at least space n.

In particular, addition of integers can be computed with zero space and multiplication can be computed in $LOGSPACE$. Recent work of Chiu et al [11] has shown that division can also be computed in $LOGSPACE$. It then follows from [2] that powering and iterated multiplication can also be computed in $LOGSPACE$. On the other hand, the best upper bound for radix conversion seems to require space $log\ n\ log\ log\ n$. (see [2]). We show that, nevertheless, for each k, there is a $LOGSPACE$ isomorphism between the binary and k-ary representations of natural numbers.

The outline of this paper is as follows. In section 2, we shall define the various complexity classes that we shall need for our developments as well as prove some basic lemmas about the closure properties of functions from these classes under composition. For example, it is well known that the family of $LOGSPACE$ functions is closed under composition and therefore this notion of $LOGSPACE$ computation is robust. We give a generalization of this result which gives upper bounds for the complexity of the composition of functions of arbitrary space complexity. In section 3, we improve some results of Cenzer and Remmel [5] by characterizing the sets of natural numbers which are $LOGSPACE$ isomorphic to $\{1\}^*$ and by giving various lemmas which ensure that a given sum or product of $LOGSPACE$ sets is $LOGSPACE$ isomorphic to $Tal(\mathbb{N})$ or to $Bin(\mathbb{N})$. Section 4 is devoted to the construction of $LOGSPACE$ models for certain standard Abelian groups. For example, we show that any computable torsion group is computably isomorphic to a $LOGSPACE$ group. Also, we show that the additive groups $\mathbb{Q}_p$ of p-adic rationals and $\mathbb{Z}(p^\infty)$ of p-adic rationals modulo 1 where p is a prime, the additive groups $\mathbb{Q}$ mod 1 and the additive group $\mathbb{Q}$ of the rationals have $LOGSPACE$ presentations over either $Bin(\mathbb{N})$ or $Tal(\mathbb{N})$. Note that the standard model of $\mathbb{Q}$ is certainly $LINSPACE$ and $PTIME$. The difficulty in obtaining a $LOGSPACE$ model is that one needs a unique representative for each rational, which seems to require finding the least common denominator of two integers. In section 5, we construct $LOGSPACE$ presentations for a general family of torsion-free Abelian groups of rank one. Finally, in section 6, we study the effective categoricity of $LOGSPACE$ groups. For example, we show that $\mathbb{Q}$, $\oplus_{i<n}\mathbb{Z}(p^\infty)$ for any $n > 0$, and any computable torsion Abelian groups have $LOGSPACE$ presentations which are not even primitive recursively isomorphic. We also give conditions under which two isomorphic $LOGSPACE$ structures are linear space isomorphic.

2 Preliminaries

Our model of computation is the multi-tape Turing machine of Papadimitriou [19]. The cursor of each tape can move independently of the cursors of other tapes. Our Turing machines are both *read-only* (input tape symbols are never overwritten) and *write-only* (the output-string cursor never moves left).

Let $\mathbb{N}$ denote the set $\{0, 1, 2, \dots\}$ of natural numbers and $\mathbb{N}^+ = \mathbb{N} - \{0\}$. A function $F(x) : \mathbb{N}^+ \to \mathbb{N}^+$ is a *proper complexity function* if F is nondecreasing and furthermore, there is a Turing machine M with input and output which, on any input x, computes the string $1^{F(|x|)}$ in $\leq \mathcal{O}(|x| + F(|x|)$ steps and uses space $\leq \mathcal{O}(F(|x|)$. Some examples are constant functions, $k \log x$, $(\log x)^k$, kx, x^k, $2^{(\log x)^k}$, 2^{kx}, 2^{x^k}, or $2^{2^{kx}}$. (We use $\log x$ as an abbreviation for $\log_2 x$.)

Fix a finite alphabet Σ and a proper complexity function G. Then a function $f : (\Sigma^*)^k \to \Sigma^*$ is computable in SPACE(G) if there is a Turing machine M with input and output which computes $f(x_1, \dots, x_k)$ using space $\leq G(|x|)$; f is computable in TIME(G) if there is a Turing machine M with input and output which computes $f(x_1, \dots, x_k)$ using time $\leq G(|x|)$. For time complexity, the restriction on input and output does not change the capability of the Turing machine, by Proposition 2.2 of [19].

We are primarily interested in the following families

$LOG = LOGSPACE = \cup_{c\in\mathbb{N}} SPACE(c\ log\ n)$;

$PLOGSPACE = \cup_{c\in\mathbb{N}} SPACE((log\ n)^c)$;

$LINSPACE = \cup_{c\in\mathbb{N}} SPACE(c\ n)$;

$SUPERSPACE = \cup_{c\in\mathbb{N}} SPACE(2^{(log\ n)^c})$;

$EXSPACE = \cup_{c\in\mathbb{N}} SPACE(2^{cn})$;

$EXPSPACE = \cup_{c\in\mathbb{N}} SPACE(2^{n^c})$;

$P = PTIME = \cup_{c\in\mathbb{N}} TIME(n^c)$.

A function mapping Σ^* to Σ^* is sometimes said to be $FLOG$ computable, or simply $FLOG$ if it is in LOG. The following is part of Theorem 7.4 of [19].

Lemma 1. *For any proper complexity function G:*

(a) $TIME(G) \subseteq SPACE(G)$;

(b) $SPACE(G) \subseteq TIME(k^{G(n)+log\ n})$ *for some* k. □

This implies in particular that $LOG \subseteq P$ and hence the following fact.

Lemma 2. *For any function f in $FLOG$, there is a constant k such that $|f(x)| \leq |x|^k$ for all inputs x.*

The standard universes for computation are the following. Let $Tal(0) = 0$ and for $n \geq 1$, let $Tal(n) = 1^n$. Then $Tal(\mathbb{N}) = \{0\} \cup 1\{1\}^* = \{Tal(n) : n \in \mathbb{N}\}$. For each $n \in \mathbb{N}^+$ and each $k \geq 2$, let $B_k(n) = b_r b_{r-1} \cdots b_0 \in \{0, 1, \ldots, k-1\}^*$ be the standard k-ary representation where $b_r \geq 0$ and $n = b_0 + b_1 k + \cdots + b_r k^r$. Let $B_k(0) = 0$ for all $k \geq 2$. Then

$$B_k(\mathbb{N}) = \{0\} \cup \{B_k(n) : n \in \mathbb{N}^+\} = \{0\} \cup \{b_r \cdots b_0 \in \{0, 1, \ldots, k-1\}^* : b_r \neq 0\}.$$

In particular, let $Bin(n) = B_2(n)$ and $Bin(\mathbb{N}) = B_2(\mathbb{N})$.

Next we consider the space complexity of composite functions. We start by giving a general result which provides an upper bound on the complexity of a composition of functions and some specific corollaries which we will need for our study of resource-bounded structures.

Theorem 1. *Let $F, G : \mathbb{N}^+ \to \mathbb{N}^+$ be nonconstant proper complexity functions such that $F(n), G(n) \geq log\ n$ for all $n \in \mathbb{N}^+$. Let g be a unary function in $SPACE(G)$ and f an n-ary function in $SPACE(F)$. Then the composition $g \circ f$ can be computed in $SPACE \leq G(2^{kF})$ for some constant k.*

Proof. We let $|x| = \max(|x_1|, \ldots, |x_n|)$ if $x = (x_1, \ldots, x_n)$ is a n-tuple of strings in $\{0, 1\}^*$.

Our proof is a generalization of the standard proof ([19], p. 164) that the composition of two $LOGSPACE$ functions is in $LOGSPACE$. That is, suppose that M is an input/output Turing machine that witnesses that $f \in SPACE(F)$ and M' is an input/output Turing machine that witnesses that $g \in SPACE(G)$. Then the input/output Turing Q which computes $g \circ f$ operates as follows. We do not explicitly store the string $f(x_1, \ldots, x_n)$ on one of the work tapes of Q. Instead, we simulate M' on input $(x_1, \ldots, x_n)$ by remembering at all times the cursor position i of the input string $f(x_1, \ldots, x_n)$ which is the output string of M. We store the string i in binary on one of the work tapes of Q. Initially, we set $i = 1$ and on a separate set of work tapes, we begin to simulate the computation of M on input $(x_1, \ldots, x_n)$. Whenever the cursor of M''s input string moves to the right, then we increment i and continue the computation of M long enough for it to produce the next output symbol. That symbol then becomes the symbol currently scanned by M'. If the cursor for M' stays in the same position, we just remember the last input symbol for M' that we scanned. If the cursor of M' moves left, then we decrement i by one and then run M on input $(x_1, \ldots, x_n)$ from the beginning, counting on a separate work tape the symbols output by M and stopping when the i-th symbol of M on $(x_1, \ldots, x_n)$ is output. Once we know this symbol, then the simulation of M' is continued.

In particular, it follows from Lemma 1 that for $x = (x_1, \ldots, x_n)$, $f(x)$ can be computed in time $c|x|^c 2^{cF(|x|)}$ for some constant c, which bounds the length of $f(x)$. Since $F(n) \geq log\ n$, it follows that $|f(x)| \leq 2^{aF(|x|)}$ for some constant a so that the space required for the simulation of the computation of M' on input $f(x_1, \ldots, x_n)$ is $\leq G(2^{aF})$. Similarly the space required for the simulation of M on input $(x_1, \ldots, x_n)$ is at most $bF(|x|)$ for some constant b and $bF(|x|) \leq G(2^{bF(|x|)})$ since $G(n) \geq log\ n$. The the total space required for the computation of Q on input $(x_1, \ldots, x_n)$ is $\leq G(2^{kF(|x|)})$ for some constant k. □

Corollary 1. *(i)* $LOGSPACE \circ LOGSPACE = LOGSPACE.$

(ii) $LINSPACE \circ LOGSPACE \subseteq PSPACE.$

(iii) $PLOGSPACE \circ LINSPACE = LINSPACE.$

(iv) $PLOGSPACE \circ PLOGSPACE = PLOGSPACE.$

(v) $SUPERSPACE \circ LINSPACE \subseteq EXPSPACE.$

(vi) $SUPERSPACE \circ PLOGSPACE = SUPERSPACE.$

(vii) $EXPSPACE \circ LOGSPACE \subseteq EXPSPACE.$

3 *LOGSPACE* Sets and Radix Representation

In this section, we establish a few lemmas about $LOGSPACE$ isomorphisms of sets which will be needed for the discussion of $LOGSPACE$ structures.

The first lemma characterizes sets isomorphic to $Tal(\mathbb{N})$ and is similar to Lemma 2.4 of [5].

Lemma 3. *Let A be a $LOGSPACE$ subset of $Tal(\mathbb{N})$ and suppose that $a_0, a_1, a_2 \ldots$ is an increasing list of the elements of A in the standard ordering. Then the following are equivalent:*

(a) A is $LOGSPACE$ isomorphic to $Tal(\mathbb{N})$.

(b) There exists a k such that for all $n \geq 2$, $|a_n| \leq n^k$.

(c) The canonical bijection between $Tal(\mathbb{N})$ and A that associates 1^n with a_n, $n \geq 0$ is in $LOGSPACE$.

Proof. Clearly, (a) implies (c) which, in turn, implies (b) by Lemma 2. Thus we need only show that (b) implies (a).

The map taking a_n to 1^n is $FLOG$ even without assumption (b). That is, given tally input $a = a_n$, one proceeds as follows. First convert a to binary b and write this on a worktape. Now a second tape will begin with $Bin(0)$ and increment at stage $t+1$ from $Bin(t)$ to $Bin(t+1)$ as long as $Bin(t) \leq b$. The output tape will begin with 0. Then at stage t, we will simulate testing whether $Tal(t) \in A$ as follows. Use the standard $LINSPACE$ conversion $Bin(t)$ into $Tal(t)$ and then the $LOGSPACE$ test of whether $Tal(T) \in A$. It follows from Corollary 1 that this computation is $LINSPACE$ in the input $Bin(t)$ and since $Tal(t) \leq a_n$, the computation can be done in $LOGSPACE$ with respect to input a_n. If the test is positive, then a "1" is appended to the output tape.

For the map taking 1^n to $a_n = Tal(m)$, assume (b) and use the following procedure. As above, at stage $t \leq n$, we will have $Bin(t)$ on one work tape and test whether $Tal(t) \in A$. If the test is positive, then we move the cursor on the input tape to the right and otherwise not. Once the end of the input tape is reached, we will have $Bin(m)$ on the work tape. The final step is to convert this to $a_n = Tal(m)$. Since $a_n \leq n^k$, it follows that $|Bin(m)| \leq log(n^k)$, so that the computation can be done in $LOGSPACE$. □

Next we consider the translation between tally to binary.

Lemma 4. *There is a LOGSPACE algorithm which takes $Tal(n)$ to $Bin(n)$ and there is a linear space algorithm which takes $Bin(n)$ to $Tal(n)$.*

Proof. On input $Tal(n) = 1^n$, simply write 1 on the worktape after reading the first 1 of 1^n and then after each additional 1 increment the binary number on the worktape. Thus after reading the first k bits of 1^n, the work tape will contain $Bin(k)$. After finishing the input, transfer $Bin(n)$ to the output tape. Since $Bin(n)$ has length $log\ n$, this requires only space $log\ n$. For the reverse translation, copy $Bin(n)$ onto the work tape and decrement it repeatedly by 1 while writing 1's on the output tape. The maximum space used on the work tape is the initial copying of the input, so this uses linear space $|Bin(n)|$. □

The next lemma is also crucial for building structures with a standard universe.

Lemma 5. *For each $k \geq 2$, the following sets are LOGSPACE isomorphic:*

(a) $Bin(\mathbb{N})$*;*

(b) $B_k(\mathbb{N})$*;*

(c) $\{0, 1, \ldots, k-1\}^*$.

Furthermore, there exists a LOGSPACE bijection $f : Bin(\mathbb{N}) \to B_k(\mathbb{N})$ and constants $c_1, c_2 \geq 0$ such that, for every $n \in \mathbb{N}$:

(i) $|f(Bin(n))| \leq c_1|Bin(n)|$ *and*

(ii) $|f^{-1}(B_k(n))| \leq c_2|B_k(n)|$.

Proof. It is easy to see that $Bin(\mathbb{N})$ is $LOGSPACE$ isomorphic to $\{0,1\}^*$. That is, the isomorphism f sends $bin(n)$ to $bin(n+1)$ and then strips off the leading 1. Clearly f and f^{-1} are $LOGSPACE$ computable.

Next for any $k \geq 2$, consider the map g_k which send $B_k(n)$ to $B_k(n+k)$ and the strips off the leading bit in the string. g_k maps $B_k(\mathbb{N})$ onto $k-1$ copies of $\{0, \ldots, k-1\}^* - \{\emptyset\}$. Clearly, both g_k and g_k^{-1} are $LOGSPACE$ computable. We claim that $\{0, 1 \ldots, k-1\}^* - \{\emptyset\}$ is $LOGSPACE$ isomorphic to $k-1$ copies of itself. That is, below we denote the elements of $\{0, 1 \ldots, k-1\}^* - \{\emptyset\}$ by σ and elements of the $k-1$ fold disjoint union by $\langle j, \sigma \rangle$, and arbitrary elements of $\{0, 1 \ldots, k-1\}^*$ are denoted by τ. The mapping is defined by the following sets of rules. For strings not beginning with 0 or 1, we have:

$$
\begin{array}{lll}
2 \to \langle 1, 0 \rangle & 2^\frown 0^n \to \langle 1, 0^{n+1} \rangle & 2^\frown \sigma \to \langle 1, \sigma \rangle \\
3 \to \langle 2, 0 \rangle & 3^\frown 0^n \to \langle 2, 0^{n+1} \rangle & 3^\frown \sigma \to \langle 2, \sigma \rangle \\
\vdots & \vdots & \vdots \\
k-2 \to \langle k-3, 0 \rangle & (k-2)^\frown 0^n \to \langle k-3, 0^{n+1} \rangle & (k-2)^\frown \sigma \to \langle k-3, \sigma \rangle \\
k-1 \to \langle k-2, 0 \rangle & (k-1)^\frown 0^n \to \langle k-2, 0^{n+1} \rangle & (k-1)^\frown \sigma \to \langle k-2, \sigma \rangle
\end{array}
$$

For strings beginning with 1, we have:

$$1 \to \langle 0, (k-1)^\frown 0\rangle \qquad 1^\frown 0^n \to \langle 0, (k-1)^\frown 0^{n+1}\rangle \qquad 1^\frown \sigma \to \langle 0, (k-1)^\frown \sigma\rangle$$

For strings beginning with 0, we have:

$$0^n \ \to \ \langle 0, 0^n\rangle$$

$$\begin{array}{ll}
0^\frown(k-1)^\frown\tau \ \to \ \langle 0, (k-2)^\frown\tau\rangle & 0^{n+1^\frown}(k-1)^\frown\tau \ \to \ \langle 0, 0^{n^\frown}(k-2)^\frown\tau\rangle \\
0^\frown(k-2)^\frown\tau \ \to \ \langle 0, (k-3)^\frown\tau\rangle & 0^{n+1^\frown}(k-2)^\frown\tau \ \to \ \langle 0, 0^{n^\frown}(k-3)^\frown\tau\rangle \\
\vdots & \vdots \\
02^\frown\tau \ \to \ \langle 0, 1^\frown\tau\rangle & 0^{n+1^\frown}2^\frown\tau \ \to \ \langle 0, 0^{n^\frown}1^\frown\tau\rangle \\
01^\frown\tau \ \to \ \langle 0, 0^\frown(k-1)^\frown\tau\rangle & 0^{n+1^\frown}1^\frown\tau \ \to \ \langle 0, 0^{n+1^\frown}(k-1)^\frown\tau\rangle
\end{array}$$

It is not hard to see that this defines a bijection f and that both f and f^{-1} are $LOGSPACE$ computable, in fact, they can be computed without using any space. Since $LOGSPACE$ functions are closed under composition, it follows that $B_k(\mathbb{N})$ is $LOGSPACE$ isomorphic to $\{0,\ldots,k-1\}-\{\emptyset\}$. However, the map h which takes $B_k(n)$ to $B_k(n+1)$ shows that $B_k(\mathbb{N})$ is $LOGSPACE$ isomorphic to $B_k(\mathbb{N})-\{0\}$ and, hence, $B_k(\mathbb{N})-\{0\}$ is $LOGSPACE$ isomorphic to $\{0,\ldots,k-1\}-\{\emptyset\}$. Thus $B_k(\mathbb{N})$ is $LOGSPACE$ isomorphic to $\{0,\ldots,k-1\}^*$

Thus, to complete the proof, we need only show that for any $k \geq 2$, $\{0,1,\ldots,k-1\}^*$ is $LOGSPACE$ isomorphic to $\{0,1\}^*$. First define

$$g : \{0,1,\ldots,k-1\} \longrightarrow \{0,1\}^*$$

by $g(0) = 0^{k-1}$ and $g(i) = 10^{i-1}$ for $1 \leq i \leq k-1$. Then for $k \geq 3$, consider the function $f_k : \{0,\ldots,k-1\}^* \to \{0,1\}^*$ which is defined as follows. First $f_k(\emptyset) = \emptyset$. If $\sigma = \sigma_1 \ldots \sigma_n \in \{0,\ldots,k-1\}^* - \{\emptyset\}$, then we scan σ and as long as we scan a sequence of 0's, we write the corresponding sequence of 0's on the output tape. If $\sigma_1 = \cdots = \sigma_{i-1} = 0$ and $\sigma_i \neq 0$, then for $j \geq i$, we add $g(\sigma_j)$ to the output tape when we scan σ_j. Thus if $\sigma = 0^n$, then $f_k(\sigma) = 0^n$ and if $\sigma = 0^r\tau_1\cdots\tau_s$ where $\tau_1 \neq 0$, then $f_k(\sigma) = 0^n g(\tau_1)\cdots g(\tau_s)$. Clearly f_k is in $LOGSPACE$. To compute f_k^{-1} on a string $\gamma = \gamma_1\cdots\gamma_n$, we scan γ and as long as we see a 0, we add a corresponding 0 on the output tape. We also have a work tape which is recording in binary, the number of elements that we have scanned. Once we see a 1, then we record on another work tape when we saw that 1 and we have another work tape which begins to record the length in binary until we see another 1 or we reach the end to the string. Clearly if we know the length s between the last 1 that we have recorded and the next 1 that we see, then we can divide s by $k-1$ and express $s = a(k-1)+j$ where $0 \leq j \leq k-2$. Once we have a and j, then we know to add $j+1$ followed by a 0's to the output tape. If we have not reached the end of the string, we replace the

position of the previous 1 with the position of the next 1 and start the process over again. It is clear that this process can be carried out in $LOGSPACE$ so that f_k^{-1} is in $LOGSPACE$. □

Lemma 6. *Let A be a nonempty $LOGSPACE$ subset of $Tal(\mathbb{N})$. Then*

(a) The set $A \oplus Tal(\mathbb{N})$ is $LOGSPACE$ isomorphic to $Tal(\mathbb{N})$ and the set $A \oplus Bin(\mathbb{N})$ is $LOGSPACE$ isomorphic to $Bin(\mathbb{N})$.

(b) The set $A \times Tal(\mathbb{N})$ is $LOGSPACE$ isomorphic to $Tal(\mathbb{N})$ and the set $A \times Bin(\mathbb{N})$ is $LOGSPACE$ isomorphic to $Bin(\mathbb{N})$.

(c) Both $Bin(\mathbb{N}) \oplus Bin(\mathbb{N})$ and $Bin(\mathbb{N}) \times Bin(\mathbb{N})$ are $LOGSPACE$ isomorphic to $Bin(\mathbb{N})$.

(d) If B is a nonempty finite subset of $Bin(\mathbb{N})$, then both $B \oplus Bin(\mathbb{N})$ and $B \times Bin(\mathbb{N})$ are $LOGSPACE$ isomorphic to $Bin(\mathbb{N})$.

Proof. The tally cases of parts (a) and (b) follow from Lemma 3. That is, for example, $A \oplus Tal(\mathbb{N})$ contains all odd numbers and therefore the nth element is certainly $\leq 2n + 1$.

For the binary cases of (a) and (b), first observe that $Bin(\mathbb{N}) - Tal(\mathbb{N})$ is $LOGSPACE$ isomorphic to $Bin(\mathbb{N})$ via the map $f(x) = x + 1 - |x|$ and $Tal(\mathbb{N}) \times Bin(\mathbb{N})$ is $LOGSPACE$ isomorphic to $Bin(\mathbb{N})$ via the map g defined as follows: $g(\langle 0, 0\rangle) = 0$,
$g(\langle 1^m, 0\rangle) = 10^{m-1}$ if $m \geq 1$,
$g(\langle 0, Bin(n)\rangle) = 1^\frown Bin(n)$ if $n \geq 1$, $g(\langle 1^m, Bin(n)\rangle) = 10^m{}^\frown Bin(n)$, and
if $m \geq 1$ and $n \geq 1$.
Then $A \oplus Bin(\mathbb{N})$ is $LOGSPACE$ isomorphic to $A \oplus Tal(\mathbb{N}) \oplus (Bin(\mathbb{N}) - Tal(\mathbb{N}))$, which is $LOGSPACE$ isomorphic to $Tal(\mathbb{N}) \oplus Bin(\mathbb{N}) - Tal(\mathbb{N})$ by the tally case and thus is $LOGSPACE$ isomorphic to $Bin(\mathbb{N})$. Finally, $A \times Bin(\mathbb{N})$ is $LOGSPACE$ isomorphic to $A \times Tal(\mathbb{N}) \times Bin(\mathbb{N})$, which is $LOGSPACE$ isomorphic to $Tal(\mathbb{N}) \times Bin(\mathbb{N})$ by the tally case and thus is $LOGSPACE$ isomorphic to $Bin(\mathbb{N})$.

For part (c), partition $\mathbb{N} \times \mathbb{N}$ into an infinite disjoint union as follows.
For each $n \geqslant 1$, define

$$\begin{aligned} A_n &= \{0, 1, \ldots, 2^n - 1\} \times \{2^n, 2^n + 1, \ldots, 2^{n+1} - 1\}, \\ B_n &= \{2^n, 2^n + 1, \ldots, 2^{n+1} - 1\} \times \{0, 1, \ldots, 2^n - 1\}, \text{ and} \\ C_n &= \{2^n, 2^n + 1, \ldots, 2^{n+1} - 1\} \times \{2^n, 2^n + 1, \ldots, 2^{n+1} - 1\}. \end{aligned}$$

Define the map f from $\mathbb{N} \times \mathbb{N}$ to $\mathbb{N}$ by $f(0, 0) = 0$, $f(1, 0) = 2$, $f(1, 1) = 3$ and for each $n \geq 1$,

$$\begin{aligned} (x, y) &\mapsto 2^n x + y + 2^{2n} - 2^n \quad \text{if} \quad (x, y) \in A_n, \\ (x, y) &\mapsto 2^n x + y + 2^{2n} \quad \text{if} \quad (x, y) \in B_n, \\ (x, y) &\mapsto 2^n x + y + 2^{2n+1} - 2^n \quad \text{if} \quad (x, y) \in C_n. \end{aligned}$$

Then it can be shown that the corresponding map from $Bin(\mathbb{N}) \times Bin(\mathbb{N})$ to $Bin(\mathbb{N})$ is a $LOGSPACE$ isomorphism.

Part (d) is straightforward so we will not give the details. □

4 *LOGSPACE* Abelian Groups

Let $\mathbb{Z}$ denote the group of integers with the usual addition. For any natural number $n > 1$, $\mathbb{Z}_n$ denote the cyclic group of order n. For a prime number p, the group $\mathbb{Z}(p^\infty)$ is the inverse limit of the sequence $\mathbb{Z}(p^n)$, or more concretely, the set of rational numbers with denominator equal to a power of p and addition modulo 1. The additive group of rational numbers is denoted by $\mathbb{Q}$. A group is said to be *torsion* if all elements have finite order and *torsion-free* if all elements (except the identity) have infinite order. The main results of this section is to show that the groups $\mathbb{Z}$, $\mathbb{Z}(p^\infty)$, $\mathbb{Q}$, $\mathbb{Q}$ mod 1, and any computable torsion Abelian groups are isomorphic to $LOGSPACE$ groups with universe equal to $Bin(\mathbb{N})$ or $Tal(\mathbb{N})$.

Before giving our constructions of $LOGSPACE$ groups isomorphic to $\mathbb{Z}$, $\mathbb{Z}(p^\infty)$, $\mathbb{Q}$, and $\mathbb{Q}$ mod 1, we need to establish a few basic lemmas. Our first lemma is completely straightforward so we state it without proof.

Lemma 7. *Let $\mathcal{A}$ be a $LOGSPACE$ structure and let ϕ be a $LOGSPACE$ bijection from A (the universe of $\mathcal{A}$) onto a set B. Then $\mathcal{B}$ is a $LOGSPACE$ structure, where the functions and relations on its universe B are defined to make ϕ an isomorphism of the structures.*

Let

$$\mathcal{A} = (A, \{R_i^{\mathcal{A}}\}_{i\in S}, \{f_i^{\mathcal{A}}\}_{i\in T}, \{c_i^{\mathcal{A}}\}_{i\in U}),$$

be a structure where the universe of $\mathcal{A}$, A, is a subset of $\mathbb{N}$ and S, T, and U are finite initial segments of $\mathbb{N}$. Then If $A = \{a_0 < a_1 < \cdots\}$, then we let $Tal(\mathcal{A})$ ($B_k(\mathcal{A})$) denote the structure whose universe is $\{Tal(a_0), Tal(a_1), \ldots\}$ ($\{Bin_k(a_0), Bin_k(a_1), \ldots\}$) and whose relations and functions are defined so that the map f which sends a_i to $Tal(a_i)$ (a_i to $Bin_k(a_i)$) for $i \geq 0$ is an isomorphism.

Lemma 8. *Let $\mathcal{M}$ be a structure with universe $M \subseteq \mathbb{N}$, and let $\mathcal{A} = Tal(\mathcal{M})$ and $\mathcal{B} = \mathrm{B}_k(\mathcal{M})$, where $k \geq 2$. Then we have*

(a) If $\mathcal{B} \in LOGSPACE$, then $\mathcal{A} \in PLOGSPACE$.

(b) If $\mathcal{B} \in LINSPACE$ and for all functions $f^{\mathcal{B}}$, it is the case that there is a constant depending on $f^{\mathcal{B}}$ such that for all but finitely many n-tuples $(m_1, \ldots, m_n)$, $|f^{\mathcal{B}}(m_1, \ldots, m_n)| \leqslant c(|m_1| + \cdots + |m_n|)$, then $\mathcal{A} \in LOGSPACE$.

Proof. (a) It easily follows from Corollary 1 and Lemma 4 that the domain $Tal(A)$ and all relations on $\mathcal{A}$ are in $LOGSPACE$. For any function symbol $f^{\mathcal{A}}$ and any $y = f^{\mathcal{A}}(x_1, \ldots, x_n)$, we can compute $Tal(y)$ from $Tal(x_1), \ldots, Tal(x_n)$ in 3 stages. First convert each $Tal(x_i)$ to $Bin_k(x_i)$, then compute $Bin_k(y)$ in $\mathcal{B}$, and finally convert $Bin_k(y)$ to $Tal(y)$. Again by Corollary 1, the first two steps can be done in $LOGSPACE$. Since, for each i, $|Bin_k(x_i)| \leq log|Tal(x_i)|$, it follows that $|Bin(y)| \leq max_i[log|Tal(x_i|]^c$ for some constant c. Thus the final step of converting $Bin_k(y)$ to $Tal(y)$ can be carried out in $PLOGSPACE$.

(b) To check whether $Tal(n) \in Tal(A)$, we first compute $Bin_k(n)$ and then test $Bin_k(n) \in Bin(A)$. Clearly we can compute $Bin_k(n)$ in $LOGSPACE$ and since $|Bin_k(n)| \leq log|Tal(n)|$, we can test whether $Bin_k(n) \in Bin(A)$ in $LOGSPACE$ as well. A similar argument applies to relations. For any function $f^{\mathcal{B}}$ and almost all $(x_1, \ldots, x_n)$, if $y = f^{\mathcal{B}}(x_1, \ldots, x_n)$, then $|Bin_k(y)| \leq c(log|Tal(x_1)|+\cdots+log|Tal(x_n)|)$ so that we can compute $Tal(y)$ in $LOGSPACE$ as well. □

The direct sum, or external weak product, of a sequence $\mathcal{A}_i = (A_i, +_i, -_i, e_i)$ of groups is defined as usual to have elements $(a_0, a_1, \ldots, a_n)$ where, for all but finitely many i, $a_i = e_i$ and the operations are coordinatewise. Thus every element of the direct sum other than the zero can be thought of as a finite sequence $(a_1, \ldots, a_n)$ where $a_i \in A_i$ for $1 \leq i < n$ and $a_n \in A_n - \{e_n\}$.

For now on, we shall fix the following $LOGSPACE$ pairing function from $\{0,1\}^* \times \{0,1\}^* \to \{0,1\}^*$. Given two strings $a_1 \cdots a_n$ and $b_1 \cdots b_m \in \{0,1\}^*$, we define the pair $\langle a_1 \cdots a_n, b_1 \cdots b_m \rangle$ as follows. First we pad the shorter string with 2's at the end so that the two strings have the same length $p = max(m, n)$. This results in two string $\bar{a}_1 \cdots \bar{a}_p$ and $\bar{b}_1 \cdots \bar{b}_p$. Then we define define $c(1) = 11$, $c(0) = 00$ and $c(2) = 10$. Then define

$$\langle a_1 \cdots a_n, b_1 \cdots b_m \rangle = c(\bar{a}_1)c(\bar{b}_1) \cdots c(\bar{a}_p)c(\bar{b}_p).$$

It is easy to see that for any i, we can recover either a_i or b_i in $LOGSPACE$.

Then we say that the sequence is *fully uniformly* $LOGSPACE$ over B where B is either $Bin(\mathbb{N})$ or $Tal(\mathbb{N})$) if the following hold:

(i) the set $\{\langle B(n), a\rangle : a \in A_n\}$ is a $LOGSPACE$ subset of $B \otimes B$, where $B(n) = Tal(n)$ if $B = Tal(\mathbb{N})$ and $B(n) = Bin(n)$ if $B = Bin(\mathbb{N})$,

(ii) the functions F and G, defined by $F(B(n), a, b) = a +_n b$ and $G(B(n), a, b) = a -_n b$, are in $LOGSPACE$,

(iii) the function $e : Tal(\mathbb{N}) \to B$, defined by $e(Tal(i)) = e_i$, is in $LOGSPACE$.

Lemma 9. *Let B be either $Tal(\mathbb{N})$ or $Bin(\mathbb{N})$. Suppose that the sequence $\{\mathcal{A}_i\}_{i\in\mathbb{N}}$ of groups is fully uniformly $LOGSPACE$ over B. Then*

(a) The direct sum $\oplus_i \mathcal{A}_i$ is computably isomorphic to a $LOGSPACE$ group with universe contained in $Bin(\mathbb{N})$.

(b) If the sequence is infinite and the universe of each A_i is equal to $Tal(\mathbb{N})$ and $e_i = 0$ for all i, then $\oplus_i \mathcal{A}_i$ is computably isomorphic to a $LOGSPACE$ group with universe equal to $Bin(\mathbb{N})$.

(c) If A_i is a subgroup of A_{i+1} for all i, and if there is a $LOGSPACE$ function $f : \{0,1\}^* \to B$ such that for all $a \in \bigcup_i A_i$, we have $a \in A_{f(a)}$, then the union $\bigcup_i \mathcal{A}_i$ is a $LOGSPACE$ group with universe contained in B.

(d) *If one of the components has universe B, and the remaining components have universes that are LOGSPACE subsets of $Tal(\mathbb{N})$, then the direct sum is computably isomorphic to a LOGSPACE group with universe B.*

(e) *If the sequence is infinite and if each component has universe $Bin(\mathbb{N})$, then the direct sum is computably isomorphic to a LOGSPACE group with universe $Bin(\mathbb{N})$.*

(f) *If each component has universe $Tal(\mathbb{N})$ and there is a uniform constant c such that for each i and any $a, b \in A_i$, we have both $|a +_i b| \leqslant c(|a| +_i |b|)$ and $|a -_i b| \leqslant c(|a| +_i |b|)$, then the direct sum is computably isomorphic to a LOGSPACE group with universe $Tal(\mathbb{N})$.*

Proof. For (a), first consider the case when $B = Tal(\mathbb{N})$. Thus we are assuming that the universe of each A_i is a subset of $Tal(\mathbb{N})$. Then we simply map the zero element of $\oplus_i \mathcal{A}_i$ to 0 and map any sequence $(a_1, \ldots, a_n)$ where $a_i \in A_i$ for $1 \leq i < n$ and $a_n \in A_n - \{e_n\}$ to $1a_10a_20 \cdots 0a_n0$. It is then easy to see that our definitions ensure that we can carry out the group operations in $LOGSPACE$. Similarly, if $B = Bin(\mathbb{N})$, then we can map the n-tuple $(a_1, \ldots, a_n)$ to the string that results by taking $1a_12a_22 \cdots 2a_n2$ and replacing each 0 in the string by 00, each 1 in the string by 11, and each 2 in the string by 10. Again, it is easy to see that our definitions ensure that we can carry out the group operations in $LOGSPACE$.

For (b), we can map the zero element of $\oplus_i \mathcal{A}_i$ to 0 and map any sequence $(a_1, \ldots, a_n)$ where $a_i \in A_i$ for $1 \leq i < n$ and $a_n \in A_n - \{e_n\}$ to $d(a_n)0d(a_{n-1})0 \cdots d(a_1)$ where $d(a_i) = \emptyset$ if $a_i = e_i$ and $d(a_i) = a_i$ otherwise. It is then easy to see we can recover the j-th bit of any a_i in $LOGSPACE$ so that our our definitions ensure that we can carry out the group operations in $LOGSPACE$. In this case, we have mapped $\oplus_i \mathcal{A}_i$ bijectively onto $Bin(\mathbb{N})$ so that $\oplus_i \mathcal{A}_i$ is computably isomorphic to a $LOGSPACE$ group with universe $Bin(\mathbb{N})$.

Part (c) easily follows from our definitions so we will not give the proof. Part (d) immediately follows from part (a) and Lemma 6.

We just sketch the proofs of (e) and (f). We let the zero element of the direct sum be 0 and the remaining elements of the sum may be viewed as finite sequences $(a_1, \ldots, a_n)$ from $Bin(\mathbb{N})$ such that $a_i \in A_i$ for $1 \leq i < n$ and $a_n \in A_n - \{e_n\}$. The operations of plus and minus are the corresponding coordinatewise operations and, hence, they can be carried out in $LOGSPACE$. For the isomorphism, sequences of length $n \geq 0$ may be mapped to $\langle n, m \rangle \in Bin(\mathbb{N}) \oplus Bin(\mathbb{N})$ since the set of sequences of length n is a finite sum of $n - 1$ copies of $Bin(\mathbb{N})$ with one copy of $Bin(\mathbb{N}) - \{0\}$ and hence isomorphic to $Bin(\mathbb{N})$ by Lemma 6. This will give a $LOGSPACE$ isomorphism of the universe which then leads to a group isomorphism by Lemma 7. This group can be converted into a $LOGSPACE$ group with universe $Tal(\mathbb{N})$ using Lemma 8 under the conditions given in (e).

□

For a given prime p, let $\mathbb{Q}_p$ denote the additive group of all p-adic rationals

Theorem 2. *Let $k \geq 1$ be in $\mathbb{N}$ and let p be a prime. Each of the groups $\mathbb{Z}$, $\bigoplus_\omega \mathbb{Z}_k$, $\mathbb{Z}(p^\infty)$, and $\mathbb{Q}_p$ are computably isomorphic to a LOGSPACE groups $\mathcal{A}$ with universe $Bin(\mathbb{N})$ and a LOGSPACE group $\mathcal{B}$ with universe $Tal(\mathbb{N})$.*

Proof. The standard structure for $\mathbb{Z}$ is clearly $LOGSPACE$ and can be made to have universe $Bin(\mathbb{N})$ or $Tal(\mathbb{N})$ by mapping n to $2n$ for $n \geq 0$ and mapping $-n$ to $2n+1$ for $n > 0$. For any $k \geq 2$, we can think of Z_k as having universe $\{0, \ldots, k-1\}$ where the operations are addition modulo k. It then easily follows that $\bigoplus_\omega \mathbb{Z}_k$ can be identified with $B_k(\mathbb{N})$ where the 0 of $\bigoplus_\omega \mathbb{Z}_k$ is sent to 0 and a sequence $(a_1, \ldots, a_n)$ with $a_i \in \{0, \ldots, k-1\}$ for $i < n$ and $a_n \in \{1, \ldots, k-1\}$ is sent to $a_n \ldots a_1$. It is then easy to see that operations will be in $LOGSPACE$ so that our result follows from Lemmas 4 and 5.

For a fixed prime number p, the group $\mathbb{Z}(p^\infty)$ consists of rational numbers of the form a/p^i where $a, i \in \mathbb{N}$, $0 \leq a \leq p^i$ and $i \geq 0$ with addition modulo 1. For our $LOGSPACE$ model $\mathcal{G}(p^\infty)$, we let the string $e_0 e_1 \ldots e_{n-1} \in B_p(\mathbb{N})$ represent the p-adic rational

$$\frac{e_0}{p} + \frac{e_1}{p^2} + \cdots + \frac{e_{n-1}}{p^n}.$$

It can be verified that the addition operation on these strings is indeed $FLOG$ computable so that $(\mathcal{G}(p^\infty), +^G)$ is a $LOGSPACE$ model of $\mathbb{Z}(p^\infty)$ with universe $B_p(\mathbb{N})$. Note that in $Z(p^\infty)$, the sum $x +^G y$ of two rationals either equals $x + y$ (if $x + y \leq 1$) or equals $x + y - 1$ (if $x + y \geq 1$), and these cases can be determined in $LOGSPACE$. Now Lemma 8 implies that there is a $LOGSPACE$ model with universe $Bin(\mathbb{N})$. Furthermore, $|a \oplus b| \leqslant \max(|a|, |b|)$, so that by Lemmas 5 and 8, there is a $LOGSPACE$ model with universe $Tal(\mathbb{N})$.

The group $\mathbb{Q}_p$ is almost the direct sum of the groups $\mathbb{Z}$ and $Z(p^\infty)$. That is, the universe of $\mathbb{Q}_p$ is the product of the universes of the two groups, but for the addition, we have to check as in the remarks above, whether the elements of $Z(p^\infty)$, viewed as rational numbers, have a sum less than 1, or not. Now let $(\mathcal{B}_1, +_1)$ be our $LOGSPACE$ model of $\mathbb{Z}$ and let $(\mathcal{B}_2, +_2)$ be our $LOGSPACE$ model of $Z(p^\infty)$ and let 1^B denote the element of B_1 corresponding to the integer 1. The desired model of $\mathbb{Q}_p$ will have elements $\langle b_1, b_2 \rangle$ with $b_1 \in B_1$ and $b_2 \in B_2$. To compute $(b_1, b_2) + (c_1, c_2)$, first compute $b_1 +_1 c_1$ and $b_2 +_2 c_2$. Note from the remarks above that we can also decide in $LOGSPACE$ whether the $b_2 +_2 c_2 = b_2 + c_2$ or equals $b_2 + c_2 - 1$. In the former case, $(b_1, b_2) + (c_1, c_2) = (b_1 +_1 c_1, b_2 +_2 c_2)$ and in the latter case, $(b_1, b_2) + (c_1, c_2) = (b_1 +_1 c_1 +_1 1, b_2 +_2 c_2)$. This construction will carry over to the models with binary and tally universes. □

Theorem 3. *The additive group $\mathbb{Q}$ of rationals and the additive group $\mathbb{Q}$ mod 1, are computably isomorphic to LOGSPACE groups with universe $Bin(\mathbb{N})$, and to LOGSPACE groups with universe $Tal(\mathbb{N})$.*

Proof. The group $\mathbb{Q}$ *mod* 1 can be represented as the infinite sum of the groups $\mathbb{Z}(p^\infty)$ over all primes p. Lemma 9 implies that there are $LOGSPACE$ models

of these groups with universe $Bin(\mathbb{N})$ and with universe $Tal(\mathbb{N})$. We will briefly explain how this direct sum can be obtained in a fully uniformly $LOGSPACE$ fashion. Let $\mathcal{A}_p$ be a $LOGSPACE$ group isomorphic to $\mathbb{Z}(p^\infty)$with universe $B = Tal(\mathbb{N})$ and define $\mathcal{C}_p$ to be a copy of $\mathcal{A}_p$ with the element a replaced by $\langle Tal(p), a\rangle$. Given $x = \langle Tal(n), \langle Tal(p), Tal(a)\rangle\rangle$, $x \in \mathcal{C}_{p_n}$ if and only if $p = p_n$, the nth prime. Since the set of primes is polynomial time in Binary, it is $LOGSPACE$ in Tally and therefore we can check whether $p = p_n$ in $LOGSPACE$. That is, given $Tal(n)$ and $Tal(p)$, convert them into $Bin(n)$ and $Bin(p)$. Then test whether $Bin(2), Bin(3), \ldots, Bin(p)$ are primes, using (and reusing) space $\leq clog(p)$–since prime testing is in P. Keep track of the the number of primes found, increment by one (in binary) when a new prime is found. After testing $Bin(p)$, just check that the counter equals $Bin(n)$ to see whether $p = p_n$.

The second clause in the definition of uniformly $LOGSPACE$ follows from the uniformity of the proof of Theorem 2. Part (e) of Lemma 9 now gives a group with universe $Tal(\mathbb{N})$. Omitting the first component $\mathcal{C}_2$ from the sequence, we get a group with universe $Tal(\mathbb{N})$ which can then by combined with a binary copy of $\mathbb{Z}(2^\infty)$ to obtain a copy of $\mathbb{Q}$ *mod* 1 with universe $Bin(\mathbb{N})$, by Lemma 6.

For the group $\mathbb{Q}$, we proceed as in the proof of Theorem 2. That is, the universe of $\mathbb{Q}$ is the product of the universes of models for $\mathbb{Z}$ and for $\mathbb{Q}$ *mod* 1 and thus by Lemma 6 may be taken to be $Bin(\mathbb{N})$ or $Tal(\mathbb{N})$ as desired. However, for the addition, we have to add the elements from $\mathbb{Q}$ *mod* 1 as rationals and then carry the integer part over. Now in our model of $\mathbb{Q}$ *mod* 1, a finite sequence of strings $\sigma^1, \ldots, \sigma^n$ where each $\sigma^i = (e^i_0, e^i_2, \ldots, e^i_{k_i-1}) \in \mathcal{B}^{k_i}_{p_i}$ represents the p_i-adic rational $\frac{e^i_0}{p_i} + \cdots + \frac{e_{k_i-1}}{p_i^{k_i}}$. To compute the sum $\sigma_1 + \cdots + \sigma_n$ requires taking a common denominator $p_1^{k_1} \cdot p_2^{k_2} \ldots p_n^{k_n}$ and using iterated multiplication and addition to obtain the numerator and finally division to obtain the desired carry value c to be added to the integer sum. The results of [2, 11] imply that this can be done in $LOGSPACE$. □

Recall that a computable Abelian group $\mathcal{A}$ is computably categorical if any computable group $\mathcal{B}$ which is isomorphic to $\mathcal{A}$ is computably isomorphic to $\mathcal{A}$. Smith [20] characterized the computably categorical Abelian p-groups.

Theorem 4. *A computable p-group $\mathcal{G}$ is computably categorical if and only if either*

1. $\mathcal{G} \approx \oplus_{n<\omega}\mathbb{Z}(p^\infty) \oplus F$ *or*
2. $\mathcal{G} \approx \oplus_{i<n}\mathbb{Z}(p^\infty) \oplus \oplus_{i\leq\omega} Z_{p^m} \oplus F$

where F is a finite p-group and $m, n \in \mathbb{N}$.

The following is a immediate corollary of our previous results in this section.

Corollary 2. *Any computably categegorical p-group is computably isomorphic to a $LOGSPACE$ group with universe equal to $Tal(\mathbb{N})$ and to a $LOGSPACE$ group with universe equal to $Bin(\mathbb{N})$.*

The following theorem is an improvement of a result of Cenzer and Remmel [4].

Theorem 5. *Any computable Abelian torsion group $\mathcal{G} = (G, +^G, -^G, e^G)$ is computably isomorphic to a LOGSPACE group whose universe is contained in B where $B = Tal(\mathbb{N})$ or $B = Bin(\mathbb{N})$.*

Proof. We may assume that the universe G of our group is just the set $Bin(\mathbb{N})$ and also that $e^G = 0$. We first renumber the elements of G as follows. For each $k \in \omega$, let $G_k =< \{1, 2, \dots, k\} >$ be the group generated by the first k elements. Now for each g in G, let $k(g)$ be the least k such that $g \in G_k$. We then order the elements of G in the following way. We say that a precedes b if either $k(a) < k(b)$, $k(a) = k(b)$ and $|a| < |b|$, $k(a) = k(b)$ and $|a| = |b|$ and a is lexicographically less than b. Since all elements of G have finite order, this ordering has order type ω and is computable. Now list the elements of G in this order as $\{a_0, a_1, \dots\}$. The idea of this renumbering is to make $a_i +^G / -^G a_j$ occur in the list as soon as possible after a_i and a_j. We now define the computable group $\mathcal{A} = (A, +^A, -^A, 0)$ where $A = Tal(\mathbb{N})$ and, for any natural numbers m,n and p,

$$m +^A / -^A n = p \iff a_m +^G / -^G a_n = a_p.$$

It is clear that $\mathcal{A}$ is computably isomorphic to $\mathcal{G}$ via the map which takes i to a_i. Now for each k, let A_k be the subgroup of $(A, +^A)$ generated by the set $\{1, \cdots, k\}$. By the definition of $\mathcal{A}$ given above, it follows that A_k is an initial segment of A. Now suppose that $i +^A / -^A j = k$ and $i \leq j$. It is clear that $k \in A_j$. Since A_j is an initial segment of A, it follows that $\{0, 1, \cdots, k\} \subset A_j$ so that $A_k \subset A_j$.

We are now ready to define the $LOGSPACE$ group $\mathcal{B} = (B, +^B)$ which is computably isomorphic to $\mathcal{A}$ and therefore computably isomorphic to $\mathcal{G}$. For each k, let $\nu(k)$ be the total time needed to compute each of the sums and differences $a +^A / -^A b$, where a and b range over A_k. Now let $\phi(k) = 1^{2^{\nu(k)}}0^k$ and let $B = \{\phi(k) : k \in \omega\}$. Let $0^B = \phi(0)$ and define the operations $+^B$ and $-^B$ so as to make ϕ a group isomorphism.

It remains to be shown that the set B is in $LOGSPACE$ and the operations $+/-^B$ are $LOGSPACE$ functions. Given a string α, in $LOGSPACE$, we can test if it of the form $1^{2^t}0^k$. That is, given any string α, we can read the initial string of n's 1's and compute $Bin(n)$ in $LOGSPACE$ in $|\alpha|$. If $Bin(n)$ is not of the form 10^r, then α is not of the form $1^{2^t}0^k$. If $Bin(n)$ is of the proper form, then α is a string of the form $1^{2^t}0^k$ only if we do not encounter a 1 after we read the initial 0. If α is not of the form $1^{2^t}0^k$, then it is not in B and if it is of the form $1^{2^t}0^k$, then we can write 1^t on a work tape using $log(|\alpha|)$ space. Then do the following to see whether it belongs in B. First, attempt to generate a list of the elements of A_k from the set $\{1, 2, \cdots, k\}$. This requires that each possible sum $a +^A b$, for $a, b \in A_k$ be computed at most once and therefore takes time at most $c \cdot \nu(k)$ for some fixed constant c. Thus we carry out this process for time $c \cdot t$ and if the process is not finished, then $1^{2^t}0^k$ is not in B.

If the process finishes, then we perform all the operations and keep track of the total time required. Again, this will take time at most $c_1 \cdot \nu(k)$ for some fixed constant c_1. Without loss of generality, we may assume that $c_1 = c$. Thus we carry out the process for time $c \cdot t$ and if the process is not finished, then $1^{2^t}0^k$ is not in B. If the process finishes, then we compare the total time computed with t and $1^{2^t}0^k \in B$ if and only if this time exactly equals t. Now this entire procedure takes time no more than $2ct$ and hence it can be carried out in space $2ct = 2c\log(|\alpha|)$.

Given two elements $1^{2^{\nu(i)}}0^i$ and $1^{2^{\nu(j)}}0^j$ of B, compute the sum (respectively difference) $1^{2^{\nu(k)}}0^k = 1^{2^{\nu(i)}}0^i +^B / -^B 1^{\nu(j)}0^j$ as follows. Assume, without loss of generality, that $i \leq j$. First, compute $k = i +^A / -^A j$. This takes time less than $c \cdot \nu(j)$, where c is the constant discussed above. Now as above, generate a list of the set A_k from the set $\{1, 2, \cdots, k\}$, perform all of the operations $a +^A / -^A b$ where a,b range over A_k, and keep track of the total time $\nu(k)$ required to perform those operations. Since $A_k \subset A_j$, this can all be done in time $2c \cdot \nu(j)$. It follows that the addition and subtraction operations of $\mathcal{B}$ are linear time in $\nu(j)$ and $\nu(i)$ and and hence $LOGSPACE$ in $1^{2^{\nu(i)}}0^i$ and $1^{2^{\nu(j)}}0^j$. Moreover, it is easy to see that $|1^{2^{\nu(k)}}0^k| \leq |1^{2^{\nu(j)}}0^j|$. It follows that B is $LOGSPACE$ group contained in $Bin(\mathbb{N})$. Moreover, we can then apply Lemma 8 to conclude that $Tal(B)$ is in $LOGSPACE$. □

5 Torsion-Free Abelian Groups

Recall that a group G is *torsion-free* if no element of G has finite order. For any Abelian group G, the notation $n \cdot g$ denotes the sum $g + \cdots + g$ of n g's. A torsion-free Abelian group G is said to be of *rank one* if there is some $a \in G$ such that for any $g \in G$ there are integers m, n such that $m \cdot a = n \cdot g$. For example, $\mathbb{Q}$ is rank one with the unit 1 playing the role of a. It is easy to see that any torsion-free group of rank one is countable and is isomorphic to a subgroup of $\mathbb{Q}$ via the map which takes g to m/n if $m \cdot a = n \cdot g$. A torsion-free Abelian group is said to have *rank* n if it is a direct sum of n subgroups of rank one.

The *character* $\chi(g)$ of an element g of an Abelian group equals

$$\{(p, n) : p \text{ is prime and } (\exists x \in G)p^n \cdot x = g\}.$$

Let p_e denote the *e-th* prime number. Then the *character sequence* of g is the sequence $(r_0, r_1, \ldots)$ where r_i is the greatest r such that $(p_i, r) \in \chi(g)$ or $r_i = \infty$ if $(p_i, r) \in \chi(g)$ for all r. Recall that any two elements of a torsion-free Abelian group of rank one have the same character sequence modulo a finite set. Thus the character sequence of a torsion-free Abelian group G of rank one may be defined, modulo finite sets, as the character sequence of any of its elements. Similarly, the character $C = \chi(G)$ of a torsion-free Abelian group G of rank 1 may be defined, modulo finite sets, as the character of any element of G.

More generally, we say that C is a *character* if C is a set of pairs (p, r) where p is a prime number and r is a positive integer with the property that whenever

$(p_i, r) \in C$ and $n \leq r$, then $(p_i, n) \in C$. Clearly, for any torsion-free Abelian group G of rank one, $\chi(G)$ is character in this sense. Conversely, any set C with these properties is also a character of some torsion-free Abelian group of rank one.

Unless otherwise stated, all groups discussed hereafter in this section will be assumed to be torsion-free Abelian groups of rank one. It is clear that two groups with the same character are isomorphic. It is a classical result that any torsion-free Abelian group of rank one is isomorphic to a subgroup of $\mathbb{Q}$. Our goal is to consider effective versions of these two results. Mal'tsev [15] proved that any two computable torsion-free groups of rank 1 with the same character are computably isomorphic.

Our next set of result will consider characters of the form $\{(p, n) : p \in C \ \& \ n \in \mathbb{N}\}$ where C is a set of primes. In this situation, we shall simply identify the character with the set of primes C. Thus the torsion-free group of rank one with character C is the infinite sum of the groups $\mathbb{Q}_p$ for $p \in C$. For subgroups of $\mathbb{Q}$ *mod* 1, the character may be similarly defined and we can say that a torsion group G has rank one if it is a subgroup of $\oplus_{p \text{ prime}} \mathbb{Z}(p^\infty)$ and, hence, is isomorphic to a subgroup of $\mathbb{Q}$ *mod* 1. If C is set of primes, then the torsion group with character C is just the infinite sum of $Z(p^\infty)$ over $p \in C$.

Since the standard model of $\mathbb{Q}$ and the standard model of $\mathbb{Q}$ *mod* 1 are isomorphic to a $LINSPACE$ group and to a $PTIME$ group, we have the following result.

Proposition 1. *For any notion K of complexity stronger than $LINSPACE$ ($PTIME$), any nonempty set of primes C, and $B = Bin(\mathbb{N})$ or $B = Tal(\mathbb{N})$, the following are equivalent.*

(i) The character C is in K.

(ii) There is a rank one torsion-free group G with character C and universe B which is in K.

(iii) There is a torsion group H of rank one with character C and universe B which is in K.

Proof. We can take the subgroups of $\mathbb{Q}$ (or of $\mathbb{Q}$ *mod* 1) of elements $\frac{m}{n}$ with all prime factors of n in C. This can be done in K since the prime factors of n may be computed in $LINSPACE$. Then to obtain a standard universe B, just take one of the components to have universe B and apply Lemma 6 as in the proof of Theorem 3 □

To obtain a corresponding result for $LOGSPACE$ is more difficult.

Theorem 6. *Let $B = Bin(\mathbb{N})$ or $Tal(\mathbb{N})$ and C be a non-empty set of primes such that $Tal(C)$ is in $LOGSPACE$. Then*

(i) There is a torsion group $\mathcal{H}$ of rank one with character C which is in $LOGSPACE$.

(ii) There is a rank one torsion-free group $\mathcal{G}$ with character C which is in $LOGSPACE$.

Proof. $\mathcal{H}$ is the the infinite sum of the groups $Z(p^\infty)$, for prime $p \in C$. Thus we can construct $LOGSPACE$ group isomorphic to $\mathcal{H}$ in much that same way that we constructed the group $\mathbb{Q}$ mod 1 in the proof of Theorem 3. That is, let $q_0, q_1, \ldots$ enumerate the members of C in increasing order. Since C is $LOGSPACE$ in tally, we can check whether $p = q_n$ in $LOGSPACE$. That is, given $Tal(n)$ and $Tal(p)$, convert them into $Bin(n)$ and $Bin(p)$. Then we can test whether $Tal(i)$ belongs to C for $i = 2, 3, \ldots, p$ and keep track of the number of elements of C found. Here we cannot actually write out $Tal(i)$ on the tape since this would use linear space so we simulate the testing of $Tal(i)$ by a composition of two functions. We have $Bin(i)$ on a work tape which is incremented at each step. The first function converts $Bin(i)$ to $Tal(i)$ and the second function tests $i \in C$. This is the composition of a linear space function followed by a $LOGSPACE$ function so, by Corollary 1, it is $LINSPACE$ in the input $Bin(i)$ and hence takes space $\leq c\ log\ i \leq c\ log\ p$. After testing $Tal(p)$, we have $p = q_n$ if and only if the counter equals n.

Then as in the proof of Theorem 3, the group $\mathcal{C}_n$ may be taken to be a $LOGSPACE$ group with elements of the form $\langle Tal(q_n), a\rangle$ for $a \in B$. Thus the universe of $\mathcal{H}$ is a $LOGSPACE$ set of triples $\langle Tal(n), Tal(q_n), a\rangle$ and the remainder of the proof is essentially the same as the proof that the group $\mathbb{Q}$ mod 1 is computably isomorphic to a $LOGSPACE$ group over B given in Theorem 3.

For the group $\mathcal{G}$, we again proceed as in the proof of Theorem 3. That is, the universe of $\mathcal{G}$ is the product of the universes of $LOGSPACE$ models for $\mathbb{Z}$ and for $\mathcal{H}$ and thus by Lemma 6 may be taken to be $Bin(\mathbb{N})$ or $Tal(\mathbb{N})$ as desired. For the addition, we have to add the elements from $\mathcal{H}$ as rationals and then carry the integer part over. The proof that can be carried out in $LOGSPACE$ is essentially the same as the proof in Theorem 3. □

6 Feasible categoricity

For any complexity class $\mathcal{C}$, we say a structure $\mathcal{A}$ is $\mathcal{C}$-categorical over a universe $B \subseteq \{0,1\}^*$ which is in $\mathcal{C}$ if and only if for any $\mathcal{C}$ structure $\mathcal{B}$ with universe B which is isomorphic to $\mathcal{A}$, there is a $\mathcal{C}$ isomorphism f from $\mathcal{A}$ onto $\mathcal{B}$ where f^{-1} is also in $\mathcal{C}$. In [6], Cenzer and Remmel explored the question of whether there exists p-time categorical groups over the universe $Tal(\mathbb{N})$ or the universe $Bin(\mathbb{N})$. For example, Cenzer and Remmel proved the following results in [6]. First, in Theorem 4.11 (p. 126), it was shown that *any* infinite computable torsion Abelian group is isomorphic to each of a countably infinite family of $PTIME$ groups, none of which are primitive recursively isomorphic to each other. The proof uses padding, so that the elements of each group are very long strings. Second, in Theorem 4.18 (p. 150), it was shown that there is an infinite family of $PTIME$ groups each with standard universe (either $Bin(\mathbb{N})$ or $Tal(\mathbb{N})$ as desired) and isomorphic to $\mathbb{Q}$ such that no two of them are primitive

recursively isomorphic. Third, in Theorem 4.31 (p. 132), it was shown that for any $n > 0$, there is an infinite family of $PTIME$ groups each with standard universe (either $Bin(\mathbb{N})$ or $Tal(\mathbb{N})$ as desired) and isomorphic to $\oplus_{i<n}\mathbb{Z}(p^\infty)$ such that no two of them are primitive recursively isomorphic. First we shall show that we can prove similar results for $LOGSPACE$ groups.

We start by strengthening Lemma 2.8 of [6]

Lemma 10. *For any p-time set $A = \{Bin(a_0) < Bin(a_1) < \cdots\}$ and any integer $c \geq 1$, there is a set $M = M(A) = \{Bin(m_0) < Bin(m_1) < \cdots\}$ such that M is in $LOGSPACE$ and the map which takes $Bin(m_i)$ to $Bin(a_i)$ is $LOGSPACE$, but there is no primitive recursive embedding of any cofinite subset of A into M which maps at most c elements to any given element.. Furthermore, M may be taken to be a subset of $Tal(\mathbb{N})$.*

Proof. Let ϕ_e be the e-th primitive recursive function mapping $Bin(\mathbb{N})$ into $Bin(\mathbb{N})$ and, for each e, let t_e be the total time required to test all numbers $Bin(n)$ where $n \leq a_e$ for membership in A and to compute $\phi_i(Bin(a_j))$ for all $j, i \leq a_e$. Thus $t_e < t_{e+1}$ because it takes at least one step to compute anything. For each e, let $m_e = 2^{t_e}$, so that $Bin(m_e) = 10^{t_e}$ and $|Bin(m_e)| = t_e + 1$. It follows that $\phi_e(Bin(a_i)) < Bin(m_i)$ for all $i \leq e$, since by convention it takes at least k steps to compute an output of length k. Let $D = \{m_e : e < \omega\}$.

We claim that D is in $LOGSPACE$. Here is the $LOGSPACE$ algorithm for testing whether $x \in M(A)$. First check to see that $x = 2^n$ for some n by computing $Bin(n)$ in $LOGSPACE$ and checking whether it is of the proper form. Then start to test $Bin(0), Bin(1), \ldots$ for membership in A. As soon as we find that $Bin(n)$ is e-th member of A so that $Bin(n) = Bin(a_e)$, then compute in order $\phi_e(a_0), \ldots, \phi_e(a_{e-1})$ and $\phi_0(a_e), \ldots, \phi_e(a_e)$ and then return to testing whether $Bin(n+1), Bin(n+2), \ldots$ are in A. If the total number of steps reaches n exactly when the computation of some $\phi_e(a_e)$ has just been completed, then $n = t_e$ so that $x = m_e$ belongs to D. Otherwise, $x \notin D$. This argument also shows that D is in $LOGSPACE$ and the map which takes $Bin(m_e)$ to $Bin(a_e)$ is in $LOGSPACE$. Finally, we let $M = M(A) = \{m_{h_i} : i < \omega\}$ where $h_0 = 1$ and $h_{i+1} = 2^{h_i}$. It is easy to see that $M(A)$ is also in $LOGSPACE$.

Now suppose by way of contradiction that ϕ_e is an embedding of of some cofinite subset C of A into M and let $c = |A \setminus C|$ which is at most c to one. Since each primitive recursive function has infinitely many indices, we may assume that $e \geq c^2$. Then $\phi_e(Bin(a_0)), \ldots, \phi_e(Bin(a_{h_e}))$ has at least $h_e/c \geq m_e + 1$ distinct elements. This implies that at least one of them is $\geq m_e$, which contradicts the observation above that $\phi_e(Bin(a_i)) < m_e$ for all $i \leq e$ and thus establishes the result. To obtain a subset of $Tal(\mathbb{N})$, replace each m_t with $m_t^* = 2^{m_t} - 1$ in the argument, so that $Bin(m_t^*) = Tal(m_t)$. Then $M^* = \{Tal(m_t)\}$ is in LOGSPACE and the map taking $Tal(m_t)$ to $Bin(a_t)$ is also LOGSPACE computable. □

Theorem 7. *For any infinite computable Abelian torsion group $\mathcal{G}$, there is an infinite family $\mathcal{G}_i = (G_i, +_i, e_i)$ of $LOGSPACE$ groups, each recursively isomorphic to G and having universe a subset of $Tal(\mathbb{N})$, such that, for all*

$i < j$, there is no primitive recursive map from $\mathcal{G}_i$ into $\mathcal{G}_j$ which is at most c to 1 for some finite c.

Proof. Again there is no loss of generality in assuming that the universe of $\mathcal{G}$ is $Bin(\mathbb{N})$ and also that $e^G = 0$. Let $\mathcal{A}$ be the p-time group with universe $\{1^{2^{\nu(i)}}0^i : i < \omega\}$ constructed in Theorem 5. Let $B_0 = \{Tal(2^{\nu(i)}) : i < \omega\}$ and let $B_{i+1} = M(B_i)$ as in Lemma 10. Thus $B_0, B_1, \ldots$ is sequence of $LOGSPACE$ subsets of $Tal(\mathbb{N})$ so that, for any $i < j$, there is no primitive recursive map from B_i into B_j.

Let ϕ_t be the $LOGSPACE$ embedding from B_t onto B_0 for each t. Define $G_t = \{1^{2^n}0^i : Tal(n) \in B_t \ \&\ \phi_t(Tal(n)) = Tal(2^{\nu(i)})\}$. Note that $i < \nu(i)$ for all i. It is easy to see from the proof of Lemma 10 that for all i and t, if $\phi_t(Tal(n)) = Tal(2^{\nu(i)})$, then $2^{\nu(i)} < n$. Moreover it is easy to see that ϕ_t is order preserving for all t. We define the operation $+_t$ on $\mathcal{G}_t$ by defining for $a \leq b$, $1^{2^m}0^a +_t 1^{2^n}0^b = 1^{2^k}0^c$ if and only if $\phi_t(m) = 1^{2^{\nu(a)}}$, $\phi_t(n) = 1^{2^{\nu(b)}}$, and $\phi_t(k) = 1^{2^{\nu(c)}}$ and $1^{2^{\nu(a)}}0^a +_{\mathcal{A}} 1^{2^{\nu(b)}}0^b = 1^{2^{\nu(c)}}0^c$. This operation is $LOGSPACE$ since $\nu(c) \leq max\{\nu(a), \nu(b)\}$ so that $k \leq max\{m, n\}$. A similar definition can be given for subtraction. Note that we can apply Lemma 4 to transform $\mathcal{G}_i$ into a LOGSPACE group with universe a subset of $Tal(\mathbb{N})$.

Suppose now that there were a primitive recursive map ψ from $\mathcal{G}_i$ into $\mathcal{G}_j$, for some $i < j$ which is at most 2^e to 1 for some finite e. Then we could define a primitive recursive map ϕ from B_i into B_j as follows. First let $B_i = \{1^{n_1} < 1^{n_2} < \cdots\}$ and $B_j = \{1^{m_1} < 1^{m_2} < \cdots\}$. Given $1^n \in B_i$, first find k such that $1^{n_k} = 1^n$. Note that since B_t is $LOGSPACE$, we can in fact find k in polynomial time in n by simply testing $1^0, \ldots, 1^n$ for membership in B_i. Next we can find $T_{n_k,i} = \{1^{n_r}0^s : r \leq k \ \&\ 1^{n_r}0^s \in G_i\}$ also in polynomial time in n since G_i is in $LOGSPACE$ and by construction $1^m0^s \in G_i$ implies that $s < m$. Note that $T_{n_1,i} \subset T_{n_2,i} \subset \cdots \subset T_{n_k,i}$ correspond to an increasing sequence of subgroups $A_{i_1} \subset A_{i_2} \subset \cdots \subset A_{i_k}$ in the group A where we continue the notation of Theorem 5 and let A_k denote the subgroup generated by $\{1, \ldots, k\}$ in A. Similarly the sequence $T_{m_1,j} \subset T_{m_2,j} \subset \cdots T_{m_k,j}$ correspond to the same increasing sequence of subgroups $A_{i_1} \subset A_{i_2} \subset \cdots \subset A_{i_k}$ in the group A. Now for each k, we have $|A_{i_{k+1}}| \geq 2|A_{i_k}|$. It follows that for each k, $\psi(T_{n_k,i})$ cannot be contained in $T_{m_{k-e-1},j}$ for any k. Thus it must be that

$$max(\{q : 1^q0^a = \psi(1^{n_p}0^s) \ for\ some\ p \leq k\}) \geq m_{k-e-1}.$$

But then the one-to-one map $\phi(1^{n_k}) = 1^{m_{k-e-1}}$ from $B_i \setminus \{1^{n_0}, \ldots, 1^{n_e-1}\}$ onto B_j could be defined primitive recursively by letting $\phi(1^{n_{k+e}})$ be equal to 1^{m_r} where r is the least h such that there is an s and q such that $1^{m_q}0^s \in \psi(T_{n_{k+e},i})$, $h \leq q$, and $1^{m_h} \notin \{\phi(1^{n_{a+e}}) : a < k\}$. Hence there can be no such ψ and therefore there is no primitive recursive map from $\mathcal{G}_i$ into $\mathcal{G}_j$ which is at most c to 1 for some finite c. □

Theorem 8. *For any prime p, for any n with $0 < n \leq \omega$, and for $B = Bin(\mathbb{N})$ or $Tal(\mathbb{N})$, there is an infinite family of $LOGSPACE$ groups $\mathcal{A}_i$ each with*

universe B and isomorphic to $\oplus_{i<n}\mathbb{Z}(p^\infty)$ and such that there is no primitive recursive structure preserving embedding of $\mathcal{A}_i$ into $\mathcal{A}_j$ for any $i < j$.

Proof. We first give the proof for $n = 1$. Let $B_0 = Tal(\mathbb{N})$ and $B_{i+1} = M(B_i)$ as defined in Lemma 10. Thus $B_i = \{Tal(m_{0,i}) < Tal(m_{1,i}) < \cdots\}$ is a sequence of $LOGSPACE$ sets such that there is no primitive recursive embedding of B_i into B_j for any $i < j$. Recall that the $m_{t,i}$'s are defined to be sequence of 1's whose lengths are increasing powers of 2. We let $n_{t,i}$ be the integer such that $Bin(n_{t,i}) = m_{t,i}$ so that the differences $n_{t+1,i} - n_{t,i}$ are increasing in t.

We build our p-time group $\mathcal{A}_i$ to have universe $B_p(\mathbb{N})$. An element $b = e_1e_2 \dots e_r \in B_p(\mathbb{N})$ will now be broken into blocks

$$b_0 = e_1 \dots e_{n_{0,i}},\ b_1 = e_{n_{0,i}+1} \dots e_{n_{1,i}}, \dots,\ b_k = e_{n_{k-1,i}+1} \dots e_r,$$

where k is the least j such that $r < n_{j,i}$ and we let $m_{-1,i} = 0$ for completeness. In the standard model of $\mathbb{Z}(p^\infty)$, the block b_s represents the rational $\sum_{t=n_{s-1,i}+1}^{n_{s,i}} e_t p^{-t}$. Thus in particular $b_s = (10 \dots 0)$ represents $p^{-n_{s_1,i}-1}$ and $b_s = (010 \dots 0)$ represents $p^{-n_{s-1,i}-2}$. Then b represents the sum of the rationals represented by the b_s's. The addition operation includes a carry to the left which can cross from a block to the previous block.

For $\mathcal{A}_i$, we shall use a non-standard model of $\mathbb{Z}(p^\infty)$ where the block is read in reverse order, so that now $b_s = (10 \dots 0)$ represents $p^{-n_{s,i}}$, $b_s = (010 \dots 0)$ represents $p^{-n_{s,i}+1}$, and in general b_s represents the sum

$$\sum_{t=n_{s-1,i}+1}^{n_{s,i}} e_t p^{-n_{s,i}+(t-n_{s-1,i}-1)}.$$

The addition operation $+^{A_i}$ is defined so that within blocks we add like integers in $B_p(\mathbb{N})$, with carry to the right. When a carry would go beyond the right end of the $(k+1)$-th block, it is added to the left end of the k-th block. A carry in the 0^{th} block is simply dropped, as in $\mathbb{Z}(p^\infty)$. Note that by our construction in the Lemma 10, $|Bin(n_{t+1})| = |m_{t+1}| \geq 2^{|m_t|} = 2^{|Bin(n_t)|}$ so that we can write down all the positions of the left ends of the smaller blocks in $LOGSPACE$ in $|Bin(b)|$. It easily follows that we can carry out this addition in $LOGSPACE$

We can now use Lemmas 4 and 5 to show $\mathcal{A}_i$ is $LOGSPACE$ isomorphic to isomorphic to a $LOGSPACE$ structure with universe $Bin(\mathbb{N})$ and to $LOGSPACE$ structure with universe $Tal(\mathbb{N})$.

We can show that there is no primitive recursive structure preserving embedding of $\mathcal{A}_i$ into $\mathcal{A}_j$ as follows. Recall that for each $i < j$, $B_j \subset B_i$. Let $r_k = n_{k,i}$, let $s_k = n_{k,j}$ and let $s_k = r_{t(k)}$. It follows from the construction of Lemma 10 that $t(k+1) > t(k) + 1$, that is, there is always at least one element of B_i between any two elements of B_j. Now observe that in $\mathcal{A}_i$, the elements of order p^{r_k} are precisely the elements of length $r_{k-1}+1$ and the elements of order $p^{r_{k+1}+1}$ are precisely the elements of length r_k. Similarly for $\mathcal{A}_j$, the elements of order p^{s_k} are precisely the elements of length $p^{s_{k-1}+1}$ and the elements of order $p^{s_{k+1}+1}$ are precisely the elements of length s_k. Thus in particular the element $0^{s_k}1$ has order $p^{s_{k+1}}$ in $\mathcal{A}_j$ and has order $p^{r_{t(k)+1}}$ in $\mathcal{A}_i$.

Suppose that ϕ were a primitive recursive structure preserving embedding of $\mathcal{A}_j$ into $\mathcal{A}_i$. Then we could primitive recursively compute s_{k+1} from s_k by the following procedure. First compute $\phi(0^{s_k}1)$, which by the isomorphism must have order $p^{r_{t(k)+1}}$ in $\mathcal{A}_j$. Therefore the length of $\phi(0^{r_k}1)$ must be $s_{k+1} - (r_{t(k)+1} - (r_{t(k)} + 1)) = s_k + s_{k+1} - r_{t(k)+1} + 1$. Now since $r_{t(k+1)} - r_{t(k+1)-1} > r_{t(k)+1} - r_{t(k)}$, it follows that $s_k + s_{k+1} - r_{t(k)+1} + 1 > r_{t(k+1)-1} > r_{t(k)+1}$. Hence we can compute $s_{t(k)+1}$ from $\phi(0^{s_k}1)$ by examining all strings of the form 1^l where $l \leq |\phi(0^{s_k}1)|$ for membership in B_i. Since B_i is a $LOGSPACE$ subset of $\{1\}^*$, this can be done in polynomial time in the length of $\phi(0^{s_k}1)$. It follows that we could primitive recursively compute s_{k+1} from s_k. But then we could compute s_k from k by primitive recursion which would contradict the fact that there is no primitive recursive embedding from $Tal(\mathbb{N})$ into B_j.

Now if $n > 1$, let $\mathcal{A}'_i = \oplus_{k<n}\mathcal{A}_i$. It is clear that $\mathcal{A}'_i$ is a $LOGSPACE$ group which is isomorphic to $\oplus_{k<n}\mathbb{Z}(p^\infty)$ and we may be assume that it has universe B by Lemma 9. Recall that the elements of $\mathcal{A}'_i$ are finite sequences of elements of $\mathcal{A}_i$. Furthermore, if an element of $\mathcal{A}'_i$ has order p then at least one of its coordinates has order p in $\mathcal{A}_i$. Let a have order p in $\mathcal{A}_i$, so that, for any $x \neq 0$ in $\mathcal{A}_i$, we have $cp^m x = a$ for some m and for some $c \in \{1, 2, \ldots, p-1\}$.

Now suppose that ϕ is a primitive recursive embedding of $\mathcal{A}'_i$ into $\mathcal{A}'_j$. For any $x \in A_i$, let (x) denote the n-tuple with all coordinates equal to x. Then let $\phi((a)) = (b_1, \ldots, b_n)$. Then at least one of these coordinates, say b_t must have order p in $\mathcal{A}_j$. Now define the primitive recursive homomorphism from $\mathcal{A}_i$ into $\mathcal{A}_j$ by letting $\psi(x)$ be the t-th projection of $\phi((x))$, so that $\psi(a) = b_t$. We show that ψ is one-to-one as follows. Suppose that $\psi(x) = 0$ and let $cp^m x = a$. Then $\psi(a) = cp^m\psi(x) = 0$ so that we must conclude that $x = 0$. This shows that ψ is a primitive recursive embedding of $\mathcal{A}_i$ into $\mathcal{A}_j$, contradicting the case $n = 1$ above and proving the result. □

Theorem 9. *For any infinite computable set C of prime numbers, there is a countably infinite family of $LOGSPACE$ groups each isomorphic to $\oplus_{p\in C}\mathbb{Z}(p^\infty)$ and having universe a subset of $Tal(\mathbb{N})$, such that no two of these are primitive recursively isomorphic. These may be taken to have standard universe $Bin(\mathbb{N})$ or $Tal(\mathbb{N})$, as desired.*

Proof. The first part of this is a corollary of Theorem 7. For the second part, just leave aside one element, p_i of C and take a direct sum with a copy of $Z(p_i^\infty)$ having standard universe. Then the resulting groups will all have standard universe by Lemma 6 and the groups will still not be primitive recursively isomorphic. □

Next we consider the additive group of the rationals $\mathbb{Q}$. Recall $\mathbb{Q}$ mod 1 be the additive group of rationals in $[0, 1)$ where the operation is addition modulo 1 is a torsion group.

Theorem 10. *Let B be either $Bin(\mathbb{N})$ or $Tal(\mathbb{N})$. Then there is an infinite family of $LOGSPACE$ groups $\mathcal{H}_i$ each with universe B and isomorphic to $\mathbb{Q}$ and such that there is no primitive recursive embedding of $\mathcal{H}_i$ into $\mathcal{H}_j$ for any $i < j$.*

Proof. First we take a specific representation of $\mathbb{Q} \mod 1$ with universe is $Tal(\mathbb{N})$ given in Theorem 3. For any $i < j$ where i and j are relatively prime, let $g(i/j)$ be the element represented by i/j in $\mathbb{Q} \mod 1$. Now by Theorem 7, there exists an infinite family of p-time groups $\mathcal{G}_i = (G_i, +_i, 0_i)$ each with universe contained in $Tal(\mathbb{N})$ where $\mathcal{G}_0 = \mathcal{G}$ and for all $i < j$, there is no primitive recursive map from $\mathcal{G}_i$ into $\mathcal{G}_j$. Moreover the construction of Theorem 7 also gives us a $LOGSPACE$ isomorphism ψ_i from $\mathcal{G}_i$ onto $\mathcal{G}_0$ for all $i > 0$. Now let $\mathbb{Z}$ be a standard group of the integers with universe equal to B. In what follows, we will simply write n for $n \cdot 1^{\mathbb{Z}}$ and $-n$ for $-n \cdot 1^{\mathbb{Z}}$ for any natural number n. Then for each i, we let $\mathcal{H}'_i$ consist of all pairs $\langle a, n\rangle$ such that $a \in G_i$, where the group operation is defined by

$$\langle a, n\rangle +_i \langle b, m\rangle = \begin{cases} \langle a +^{H_i} b, n +^{\mathbb{Z}} m\rangle & \text{if } g^{-1}(\psi_i(a)) + g^{-1}(\psi_i(b)) < 1 \\ \langle a +^{H_i} b, n +^{\mathbb{Z}} m +^{\mathbb{Z}} 1^{\mathbb{Z}}\rangle, & \text{if } g^{-1}(\psi_i(a) + g^{-1}(\psi(b)) \geq 1. \end{cases}$$

It is easy to see that our definitions ensure that each $\mathcal{H}'_i$ is a $LOGSPACE$ group which is isomorphic to $\mathbb{Q}$. Since $\mathcal{H}'_i$ has universe equal to $G_i \times B$, which is $LOGSPACE$ isomorphic to B by Lemma 6, it follows that $\mathcal{H}'_i$ is $LOGSPACE$ isomorphic to a $LOGSPACE$ group $\mathcal{H}_i$ with universe B.

Now suppose that there were a primitive recursive isomorphism from $\mathcal{H}_i$ onto $\mathcal{H}_j$ for some $i < j$. Then clearly there would be a primitive recursive isomorphism θ from $\mathcal{H}'_i$ onto $\mathcal{H}'_j$. This induces an isomorphism f from $\mathbb{Q}$ to $\mathbb{Q}$ which must map a given rational r to ar, where $a = \pm p/q$ is the image of 1 under the isomorphism and p is relatively prime to q. Now consider the corresponding map $h : \mathbb{Q} \mod 1 \to \mathbb{Q} \mod 1$ defined by $h(r) = g(r) \mod 1$. If $p < q$, then the map h is clearly one-to-one. If $q < p$ and $p = (c-1)q + s$ for some c and some s with $0 \leq s < q$, then h has kernel $\{0, q/p, 2q/p, \ldots, cq/p\}$ and is c to 1. With this in mind, consider the primitive recursive map ϕ from G_i into G_j defined by

$$\phi(a) = \pi_0(\theta(\langle a, 0\rangle)).$$

Then by the discussion above ϕ would have to be a d to 1 map for some finite d, contradicting the choice of $\mathcal{G}_i$ and $\mathcal{G}_j$ by Theorem 7. Thus there can be no such primitive recursive isomorphism θ and hence the groups $\mathcal{H}_i$ satisfy the requirements of the theorem. □

In [8], Cenzer and Remmel examined extra hypotheses under which groups of low time complexity should have isomorphisms between them of low time complexity. It was suggested in [8] that suitable conditions on divisibility might imply that two $PTIME$ copies of $\mathbb{Z}(p^\infty)$ are $EXPTIME$ isomorphic, for example. In the remainder of this article, we will present some results of this type.

Let us say that a torsion group G whose universe is contained in $Bin_k(\mathbb{N})$ for some k has *linear size order* if there exists a constant $c \geq 1$ such that for all $a \in G$, $|Bin_k(o(a))| \leq c|a|$ and $|a| \leq c|Bin_k(o(a))|$. For example, in the

standard model of $\mathbb{Z}(p^\infty)$ with elements taken from $\{0, 1, \ldots, p-1\}^*$, the order of an element $a = (e_0, e_1, \ldots, e_{n-1})$ with $e_{n-1} \neq 0$ is precisely p^n, so that $|Bin_p(o(a))| = |a|$.

Theorem 11. *Let $\mathcal{G}$ and $\mathcal{H}$ be two LINSPACE groups isomorphic to $Z(p^\infty)$ and each having linear size order. Then there is a LINSPACE isomorphism between $\mathcal{G}$ and $\mathcal{H}$.*

Proof. It suffices to show that each group is linear space isomorphic to the standard model of $Z(p^\infty)$, so we may assume that $\mathcal{G}$ is the standard model. The mapping from $\mathcal{G}$ to $\mathcal{H}$ is defined as follows. Recursively define a sequence of elements $a_0, \ldots, a_{n-1} \in \mathcal{H}$ so that a_0 is the the shortest and lexicographically least element of $\mathcal{H}$ which has order p and, for each n, a_{n+1} is the shortest and lexicographically least element a of $\mathcal{H}$ such that $p \cdot a = a_n$. Then $(c_0, \ldots, c_{n-1})$ is mapped to

$$c_0 \cdot a_0 + c_1 \cdot a_1 + \cdots + c_{n-1} \cdot a_{n-1} = (c_0 p^{n-1} + c_1 p^{n-2} + \cdots + c_{n-2} p + c_{n-1}) \cdot a_{n-1}.$$

It is easy to see that this is a group isomorphism from $\mathcal{G}$ onto $\mathcal{H}$. The function may be computed in linear space as follows. Given input $(e_0, \ldots, e_{n-1})$, which has order p^n, compute the desired element a_{n-1} as follows. Since $o(a_n) = p^n$, we know by assumption that $|a_{n-1}| \leq cn \leq c|(e_0, \ldots, e_{n-1})|$. Thus we can use linear space to guess a value $a = a_{n-1}$ of length $\leq cn$. Now to ensure that a_{n-1} is the shortest and lexicographically least a such that $pa = pa_{n-1}$, use linear space to compute px for all elements x of length $\leq cn$ and, if $px = pa$, check to see that a is either shorter or lexicographically less than x. If this works, we say that a_{n-1} is *optimal*, let $a_{n-2} = pa_{n-1}$ and again check that a_{n-2} is optimal among all x with $px = pa_{n-2}$. Continue this process, so that at stage i, we have stored a_{n-1} and also a_{n-i} as well as i and we have checked the optimality of $a_{n-1}, \ldots, a_{n-i}$. If at any stage a_{n-i} is not optimal, then we return to the beginning and take a new value for $a = a_{n-1}$. Otherwise we continue until we have verified a correct choice of a_{n-1}. Then $(c_0 p^{n-1} + c_1 p^{n-2} + \cdots + c_{n-2} p + c_{n-1}) \cdot a_{n-1}$ may be computed in linear space from a_{n-1} and $(c_0, \ldots, c_{n-1})$.

For the inverse mapping, suppose we are given some $b \in \mathcal{H}$. Since $o(b) \leq c|b|$, we may compute the order $o(b) = p^n$ in linear space by repeatedly adding b to itself, keeping i on one tape and $i \cdot b$ on a second tape, so that $o(b)$ is the value of i when $i \cdot b = 0$. Once we have p^n, we may compute the element a_{n-1} as above and then compute the integer $m = c_0 p^{n-1} + \cdots + c_{n-1}$ such that $b = m \cdot a_{n-1}$ by repeatedly adding a_{n-1} to itself until we reach b. Then the desired coefficients $c_0, \ldots, c_{n-1}$ may be read directly from the p-ary form of m. □

Theorem 12. *Let $\mathcal{G}$ and $\mathcal{H}$ be two LINSPACE groups isomorphic to $\mathbb{Q}$ mod 1 and each having linear size order. Then there is a LINSPACE isomorphism between $\mathcal{G}$ and $\mathcal{H}$.*

Proof. Let the standard linear space model $\mathcal{G}$ of $\mathbb{Q}$ *mod* 1 be the set of quotients $\frac{n}{d}$ of relatively prime positive integers $n \leq d$ together with 0. As in the proof of

Theorem 11, we will show that any other model $\mathcal{H}$ is $LINSPACE$ isomorphic to the standard model $\mathcal{G}$.

Given input $\frac{n}{d} \in \mathcal{G}$, factor d into a product of prime powers $p_1^{e_1} p_2^{e_2} \cdots p_k^{e_k}$– this can be done in linear space and the result can also be stored in linear space. Then we can compute $m_1, m_2, \ldots, m_k$ such that

$$\frac{n}{d} = m_1/p_1^{e_1} + \cdots + m_k/p_k^{e_k}$$

in linear space by guessing the sequence $m_1, \ldots, m_k$ and then checking the sum. Now for $i = 1, \ldots, k$, proceed as in the proof of Theorem 11 to compute the element $a_i = a_{i,e_k-1}$ and then $b_i = m_i \cdot a_i$. Then $\frac{n}{d}$ is mapped to the sum $b_1 + \cdots + b_k$, where we can reuse the space needed by just saving $b_1 + \cdots + b_i$ while computing b_{i+1}.

For the inverse, first observe that the element a_i above corresponding to $p_i^{e_i}$ is mapped to $1/p_i^{e_i}$. Now suppose we are given some $b \in \mathcal{H}$. First compute the order $o(b)$ and factor it into prime powers $p_1^{e_1} p_2^{e_2} \cdots p_k^{e_k}$. Then compute the elements $a_1, \ldots, a_k$ which map to $1/p_i^{e_i}$ as above. Since $o(a_i) = p_i$, we see that

$$o(a_1) + o(a_2) + \cdots + o(a_k) \leq c|p_1^{e_1}| + \cdots + |p_k^{e_k}| \leq c|o(b)|,$$

so that the sequence $a_1, \ldots, a_k$ may be stored. Then we simply guess coefficients $m_1, \ldots, m_k$ such that $b = m_1 a_1 + \cdots m_k a_k$ and check until they are correct. Finally, map b to $\frac{m_1}{p_1^{e_1}} + \cdots \frac{m_k}{p_k^{e_k}}$. □

These two results have versions for $PSPACE$, $SUPERSPACE$, $EXSPACE$ and $EXPSPACE$ with similar proofs.

References

[1] M. Agrawal, N. Kayhal and N. Saxena, *PRIMES is in P*, Annals of Mathematics 160 (2004), 781-793.

[2] [BCH86] P. W. Beame, S. A. Cook and H. J. Hoover, *Log depth circuits for division and related problems*, SIAM J. Computing 15 (1986), 994-1003.

[3] [CCHM] W. Calvert, D. Cenzer, V. Harizanov and A. Morozov, Δ_2^0 *categoricity of equivalence structures*, Ann. Pure Appl. Logic, to appear.

[4] [CR91] D. Cenzer and J. B. Remmel, *Polynomial-time versus recursive models*, Ann. Pure Appl. Logic 54 (1991), 17-58.

[5] [CR92] D. Cenzer and J.B. Remmel, *Polynomial-time Abelian groups*, Ann. Pure Appl. Logic 56 (1992), 313-363.

[6] [CR95a] D. Cenzer and J.B. Remmel, *Feasibly categorical Abelian groups*, in Feasible Mathematics II, ed. P. Clote and J. Remmel, Prog. in Comp. Science and Appl. Logic 13, Birkh'auser (1995), 91-154.

[7] [CR95b] D. Cenzer and J.B. Remmel, *Feasibly categorical models*, in Logic and Computational Complexity, ed. D. Leivant, Springer Lecture Notes in Computer Science 960 (1995), 300-312.

[8] [CR98a] D. Cenzer and J.B. Remmel, *Complexity and categoricity*, Information and Computation 140 (1998), 2-25.

[9] [CR98b] D. Cenzer and J.B. Remmel, *Complexity-theoretic model theory and algebra*, in Handbook of Recursive Mathematics (Vol. I), ed. Y. Ershov, S.S. Goncharov, A. Nerode and J.B. Remmel, Elsevier Studies in Logic and Found. Math. 138 (1998), 381-513.

[10] D. Cenzer and Z. Uddin, Space complexity of structures, Proc. CIE 2006 (Computability in Europe, Swansea, June 2006), eds. A. Beckmann, U. Berger, B. Loewe and J. Tucker, Springer Lecture Notes in Computer Science, Vol. 3988 (2006), 55-64.

[11] [CDL] A. Chiu, G. Davida and B. Litow, *Division in logspace-uniform* NC^1, Theor. Inform. Appl. 35 (2001), 259-275.

[12] [G] S. Grigorieff *Every Recursive Linear Ordering Has a Copy in DTIME(n)*, J. Symbolic Logic **55** (1990) 260-276.

[13] [HU69] J.E. Hopcroft and J.D. Ullman, *Formal Languages and their Relation to Automata*, Addison-Wesley (1969)

[14] [Knuth98] *The Art of Computer Programming, Volume 2: Seminumerical Algorithms*, Addison-Wesley, 1998.

[15] [Mal62] A.I. Mal'tsev, *On recursive abelian groups*, Soviet Math.-Doklady 3 (1962), 1431-1434.

[16] [NR89] A. Nerode and J.B. Remmel, *Complexity-theoretic algebra II: Boolean algebras*, Ann. Pure Appl. Logic 44 (1989), 71-79.

[17] [NR90a] A. Nerode and J.B. Remmel, *Polynomial time equivalence types*, in Logic and Computation, ed. W. Sieg, Contemp. Math. 106 (1990), 221-249.

[18] [NR90b] A. Nerode and J.B. Remmel, *Polynomially isolated sets*, in Recursion Theory Week (Oberwolfach 1989), ed. K. Ambos-Spies, G.H. Muller and G.E. Sacks, Springer Lecture Notes in Math. 1432 (1990), 323-362.

[19] [Papa95] C. H. Papadimitriou, *Computational Complexity*, Addison-Wesley (1995).

[20] [Smith] R. Smith, *Two theorems on autostability in p-groups*, Logic Year 1979-80 (Storrs, CT), Lecture Notes in Math. 859, Springer-Verlag, Berlin (1981), 302-311