DECISION PROBLEMS IN INVERTIBLE AUTOMATA

evan bergeron & klaus sutner

May 5, 2017

Abstract

We consider a variety of decision problems in groups and semigroups induced by invertible Mealy machines. Notably, we present proof that, in the Abelian case, the automorphism membership problem is decidable in these semigroups. In addition, we prove the undecidability of a Knapsack variant. Partial work toward the decidability of the IsGroup problem is discussed.

CONTENTS

INTRODUCTION

The word problem is a classic group-theoretic decision problem. Given a finitely generated group G, and a word w over the generators (and their inverses), the word problem asks "is $w = 1$ in G." The word problem is known to be undecidable in surprisingly small classes of groups - see [2] and [3] for background.

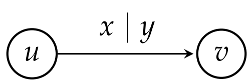
The invertible Mealy machines we consider here give rise to a class of semigroups (and sometimes groups) for which the word problem is decidable. The computability picture here is rather nuanced, however. Similarly important decision problems, among them the conjugacy problem, and the isomorphism problem are known to be undecidable - see [21] for details.

In this paper, we present proof that, for the Abelian case, automorphism membership testing is decidable in this class of semigroups.

Serre first suggested the study of subgroups of the full automorphism group $\operatorname{Aut} 2^*$ of the infinite binary tree $\mathbf{2}^*$ in [19]. This notion has been usefully applied across group theory; a classic result here is Grigorchuk's group of intermediate growth, now known to be generated by the 5 state invertible machine shown in figure 1.

BACKGROUND

An *automaton* is formally a triple (Q, Σ, δ), where Q is some finite state set, Σ is a finite alphabet of *symbols*, and δ is a transformation on $Q \times \Sigma$. Automata are typically viewed as directed graphs with vertex set Q and an edge labeled $x \mid y$ between u, v if $(u, x)\delta = (v, y)$.



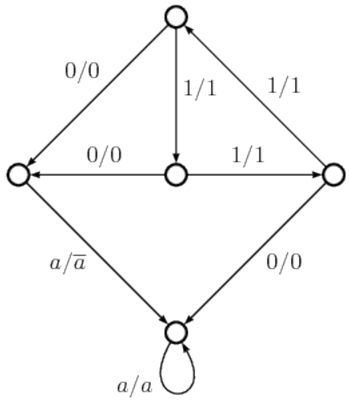


Figure 1: Grigorchuk's 5 state machine

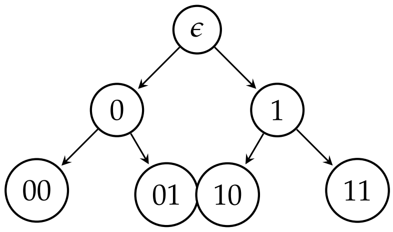


Figure 2: $\mathbf{2}^*$ interpreted as the infinite binary tree

One interprets this as if $\mathcal{A}$ is in state u and reads symbol x, then $\mathcal{A}$ transitions to state v and outputs symbol y. A computation within $\mathcal{A}$ may then start at some state q_0, and on input $\alpha_0\alpha_1 \ldots \alpha_k$, output $\beta_0\beta_1 \ldots \beta_k$, where $(q_i, \beta_i) = (q_{i-1}, \alpha_i)\delta$ for all $i = 0 \ldots k$.

As in the above case, where δ outputs exactly one character for every transition, we call the automaton $\mathcal{A}$ *alphabetic*. An automaton is called *invertible* when every state in Q has some bijection $\pi \in \mathfrak{S}_n$ on Σ such that $(u, x)\delta = (v, \pi(x))$. Here, $\mathfrak{S}_n$ denotes the symmetric group on n letters. A state in $\mathcal{A}$ is a *copy state* if π is the identity permutation and is a *toggle state* otherwise. The present paper is concerned only with invertible, alphabetic automata.

Actions on the infinite tree

We may identify the set Σ^* with an infinite, regular tree of degree $|\Sigma|$. The root is labelled with the empty string ϵ, and a vertex labelled w has the child wa for each $a \in \Sigma$. Out of convenience, we will frequently conflate a vertex with its label.

Each state $q \in Q$ acts on the corresponding tree, sending vertex w to wq. Moreover, if $\alpha\alpha' q = \beta\beta'$, then $\alpha q = \beta$, for any $\alpha, \alpha', \beta, \beta' \in \Sigma^*$. Which is to say, q's action on the tree is an adjacency-preserving map and is thus an endomorphism on the tree. Additionally, q is length-preserving, and thus preserves levels of the tree (and thus is an automorphism of the tree).

We extend the action of Q on Σ^* to words $q = q_1 \ldots q_n$ over Q^+ by

$$wq = (\ldots((wq_1)q_2) \ldots q_n).$$

This computation corresponds with running $\mathcal{A}$ starting at state q_1, then taking that output and running it through the machine starting at state q_2, and so on. We adopt the convention of applying functions from the right here. In this way, function composition corresponds naturally with string concatenation.

Thus, there is a natural homomorphism $\phi : Q^+ \to \operatorname{Aut} \mathbf{2}^*$, where $\operatorname{Aut} \mathbf{2}^*$ denotes the semigroup of automorphisms of the tree $\mathbf{2}^*$. We denote the image of ϕ by $\Sigma(\mathcal{A})$.

Semigroup theory

A *semigroup* is a set S paired with a binary operation $f : S \times S \to S$ such that S is closed under f and f is associative over S. Any set of endofunctions forms a semigroup under composition.

A semigroup is called *Abelian* when its corresponding binary operation is commutative.

For an automaton $\mathcal{A}$, we denote by $S(\mathcal{A})$ the semigroup generated by Q under composition. $\mathcal{A}$ is said to be *commutative* or *Abelian* when $S(\mathcal{A})$ is Abelian. We write $G(\mathcal{A})$ for the group generated by the elements of Q and their inverses.

One may also speak about $S(\mathcal{A})$ and $G(\mathcal{A})$ without explicit reference to an automaton $\mathcal{A}$, yielding the corresponding definition:

Definition 1. *We call a semigroup S an* automaton semigroup *if there is some automaton $\mathcal{A}$ with $S \simeq \Sigma(\mathcal{A})$. Similarly, a group G is called an* automaton group *if $G \simeq \Sigma(\mathcal{A})$ for some automaton $\mathcal{A}$.*

Wreath Recursions

Any automorphism f of Σ^* can be written in the recursive form:

$$f = (f_{\alpha_1}, f_{\alpha_2}, \ldots, f_{\alpha_n})\tau$$

where $n = |\Sigma|$ and each f_α is an automorphism of a subtree of the root. Here, τ is some permutation on Σ. In the case where $\Sigma = \{0, 1\}$, we have $f = (f_0, f_1)\sigma$ where σ denotes transposition. If $f = (f_0, f_1)\sigma$, f is said to be *odd*. If $f = (f_0, f_1)$, f is said to be *even*. That is to say, automorphisms may be classified as even or odd depending on their action on the first level of the tree.

The set of even automorphisms form a subgroup H of index 2 in $G(\mathcal{A})$.

The automorphism semigroup of Σ^* decomposes into a recursive wreath product

$$\operatorname{Aut} \Sigma^* = \operatorname{Aut} \Sigma^* \wr \tau_\Sigma$$

where τ_Σ is the tranformation semigroup on Σ. Which is to say,

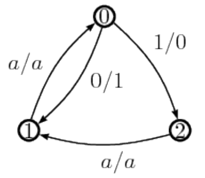
$$\operatorname{Aut} \Sigma^* = \underbrace{(\operatorname{Aut} \Sigma^* \times \ldots \times \operatorname{Aut} \Sigma^*)}_{n \text{ times}} \rtimes \tau_\Sigma.$$

Definition 2. *Define* residuation maps $\partial_a : S(\mathcal{A}) \to S(\mathcal{A})$ *that map $f = (f_0, f_1)\sigma$ to f_a, and a* parity map par *such that $\mathsf{par}(f) = \sigma$.*

When restricted to the subgroup H of even automorphisms. residuation maps are group homomorphisms

Note that a subgroup G of $\mathrm{Aut}(2^*)$ need not be closed under residuation; if it is, we call it *self-similar* or *state-closed*. In this case, the wreath characterization in the full automorphism group carriers over and we have $G \cong (G \times G) \rtimes \tau_2$.

Example 1. *The following automaton is called A_2^3, a notable member of the class of* cycle-cum-chord *automata discussed in [23].*



Write $\underline{i}$ for the automorphism corresponding to state i. Then we have the following wreath recursions: $\underline{0} = (\underline{1}, \underline{2})\sigma$, $\underline{1} = (\underline{0}, \underline{0})$, $\underline{2} = (\underline{1}, \underline{1})$. Correspondingly, we can see $\partial_0 \underline{0} = \underline{1}$ and $\partial_1 \underline{0} = \underline{2}$.

Decision problems

Automaton semigroups exhibit many interesting and nuanced computability properties. While it is an easy result that the WORD PROBLEM is solvable in such semigroups, similar group-theoretic problems such as the CONJUGACY PROBLEM and FINITENESS PROBLEM have been shown to be undecidable (see [21], and [6], respectively).

Various other semigroup theoretic decision problems have recently been considered for small classes of semigroups by Cain in [3]. We consider a subset of his distinguished properties in the automaton semigroup case here.

IsAbelian *is polynomial time*

We present a polynomial time algorithm to determine if an input automaton $\mathcal{A}$ is Abelian.

For a binary invertible automaton $\mathcal{A}$, define the *gap* of an automorphism $f \in G(\mathcal{A})$ to be $\gamma_f = (\partial_0 f)(\partial_1 f)^{-1}$.

The following result is adapted from [17].

Lemma 1. *$\mathcal{A}$ is Abelian if and only if all even automorphisms in $S(\mathcal{A})$ have gap I and odd automorphisms share the same constant gap.*

Proof. Suppose $\mathcal{A}$ is Abelian; so $fg = gf$ for all f, g in $S(\mathcal{A})$. If f and g are both odd, residuating both sides gives

$$(\partial_a f)(\partial_{\overline{a}} g) = \partial_a(fg) = (\partial_a gf) = (\partial_a g)(\partial_{\overline{a}} f)$$

which yields $\gamma_f = \gamma_g$. Here, $\overline{a}$ denotes $1 - a$. If f is even and g odd, without loss of generality, we have

$$(\partial_0 f)(\partial_0 g) = \partial_0(fg) = \partial_0(gf) = (\partial_0 g)(\partial_1 f)$$

which, with algebraic manipulation, yields $\gamma_f = I$.

Conversely, first suppose f and g are both odd. Then $fg = (\partial_0 f \partial_1 g, \partial_1 f \partial_0 g)$ and $gf = (\partial_0 g \partial_1 f, \partial_1 g \partial_0 f)$. Since $\gamma_f = \gamma_g$, these wreath recursions are the same. If f is even and g odd, $fg =$

$(\partial_0 f \partial_0 g, \partial_1 f \partial_1 g)\sigma$ and $gf = (\partial_0 g \partial_1 f, \partial_1 g \partial_r f)\sigma$, which are the same expansion, since $\gamma_f = \gamma_g$.

If f and g are both even, both directions of the claim follow by induction. $\square$

Definition 3. *For an automaton $\mathcal{A}$ with states $s_1 \ldots s_n$, the* inverse automaton *of $\mathcal{A}$, denoted $\mathcal{A}^{-1}$, has state set $t_1, \ldots t_n$ and transitions $\partial_a t_i = \partial_{\bar{a}} s_i$, with t_i a toggle state if and only if s_i is as well.*

It is easy to verify by induction that $t_i = s_i^{-1}$ for all i.

Definition 4. *For an automaton $\mathcal{A} = (Q, \Sigma, \delta)$, the* acceptor of $\mathcal{A}$ at *t, denoted $\mathcal{A}(t)$, is a partial DFA with state set Q, input alphabet $\Sigma \times \Sigma$, and transitions $s \xrightarrow{a \times b} s'$ for each transition $t \xrightarrow{a|b} t'$ in $\mathcal{A}$. Every state is accepting. The first track of the alphabet $\Sigma \times \Sigma$ may be interpreted as the input to $\mathcal{A}$, and the second, the output of $\mathcal{A}$.*

Lemma 2. *The language of the acceptor $\mathcal{A}(t)$ is*

$$\{(x_1, y_1)(x_2, y_2), \ldots, (x_n, y_n) \mid y_1 \ldots y_n = (x_1, \ldots x_n)t\}$$

Proof. By induction on the length of the input string. $\square$

Definition 5. *For an automaton $\mathcal{A} = (Q, \Sigma, \delta)$, the* product automaton *$\mathcal{A} \times \mathcal{A}$ is a machine with state set $Q \times Q$ and transition function defined by $\partial_a(s_1, s_2) = (\partial_a s_1, \partial_{as_1} s_2)$.*

We can see by induction that the behavior of each state (s_1, s_2) in the product automaton corresponds to the word $s_1 s_2 \in S(\mathcal{A})$.

Theorem 1. *There is a polynomial time algorithm to check if an automaton $\mathcal{A}$ is Abelian.*

Proof. On input automaton $\mathcal{A}$, build the inverse automaton $\mathcal{A}^{-1}$. Construct the product automaton $\mathcal{A} \times \mathcal{A}^{-1}$. Then for each toggle state t_i of $\mathcal{A}$, for the state $s_i = (\partial_1 t_i, \partial_1 t_i^{-1})$ in $\mathcal{A} \times \mathcal{A}^{-1}$, construct the acceptor DFA $(\mathcal{A} \times \mathcal{A}^{-1})(s_i)$. Verify all the constructed DFAs are equivalent. $\square$

The reader may be interested to note that this product automaton construction also provides proof that the word problem for automaton semigroups is decidable.

Automorphism MEMBERSHIP

This section considers the subsemigroup $S(\mathcal{A})$ of Aut $\mathbf{2}^*$ generated by the associated automophisms of an invertible binary transducer. We assume minimality throughout this section.

We provide proof that the automorphism membership question is decidable in the Abelian case, and discuss partial work toward the general case. Some necessary background from [16] is outlined below.

Theorem 2. *If $\mathcal{A}$ is Abelian, then $G(\mathcal{A})$ is isomorphic to either a finite Boolean group or to $\mathbb{Z}^m$ for some $m \geq 1$. In the latter case, there is an isomorphism $\phi : G(\mathcal{A}) \to \mathbb{Z}^m$ satisfying the following recursion*

$$\phi^{-1}(v) = \begin{cases} (\phi^{-1}(A \cdot v), \phi^{-1}(A \cdot v)) & \text{if } \phi^{-1} \text{ is even} \\ (\phi^{-1}(A \cdot v - r), \phi^{-1}(A \cdot v + r)) & \text{otherwise} \end{cases}$$

where $A \in GL(m, \mathbb{Q})$ and $v \in \mathbb{Q}^m$. Additionally, for all $v \in \mathbb{Z}^m$, $A \cdot v \in \mathbb{Z}^m$ or $A \cdot v \pm r \in \mathbb{Z}^m$.

We call the matrix A above the *residual matrix* of A. The vector r is referred to as the *residual vector*. Put differently, this theorem specifies that when $\mathcal{A}$ is Abelian, residuation is an affine map.

We have the following properties of the residual matrix A:

Theorem 3. *If $G(\mathcal{A}) \cong \mathbb{Z}^m$ and A is its associated residual matrix, A satisfies the following properties:*

1. *A is contracting; its spectral radius is less than 1*
2. *A is 1/2-integral, meaning that A^{-1} is a subgroup of index 2 in $\mathbb{Z}^m$. Therefore A be represented as*

$$\begin{bmatrix} \frac{a_{1,1}}{2} & a_{1,2} & \cdots & a_{1,m} \\ \frac{a_{2,1}}{2} & a_{2,2} & \cdots & a_{2,m} \\ \vdots & \vdots & \ddots & \vdots \\ \frac{a_{m,1}}{2} & a_{m,2} & \cdots & a_{m,m} \end{bmatrix}$$

   where all $a_{i,j}$ are integers.

3. *The characteristic polynomial $\chi_A(x)$ is irreducible over $\mathbb{Q}$ and has the form*

$$\chi_A(x) = x^m + \frac{1}{2} g(x)$$

   for some $g \in \mathbb{Z}[x]$ of degree at most $m - 1$. In particular, the constant term is $+ - \frac{1}{2}$.

4. *A is invertible and the characteristic polynomial $\chi_{A^{-1}}(x)$ is integral and irreducible over $\mathbb{Q}$. From property 2, Laplace expansion yields that A^{-1} is an integral matrix that is similar to the companion matrix of $\chi_{A^{-1}}(x)$ over $\mathbb{Q}$.*

The Latimer and MacDuffee proved the following theorem in [12]:

Theorem 4. *If $p(x) \in \mathbb{Z}[x]$ is monic and irreducible, the $GL(m, \mathbb{Z})$ similarity classes of integral matrices whose characteristic polynomial coincides with $p(x)$ is in one-to-one correspondance with ideal classes of the ring $\mathbb{Z}[\theta]$, where θ is any root of $p(x)$.*

Property 1 of theorem 3 provides a bound on the coefficients of $\chi_A(x)$. When combined with property 4 and the above theorem, it can be shown that, for fixed m, there exist only finitely many possibilities of A, up to $GL(m, \mathbb{Z})$ similarity.

Definition 6. *Take G to be some self-similar group. We may construct the* complete group automaton *(occasionally abbreviated as the* complete automaton*) for G, written $\mathcal{C}_G$, as follows: the automaton has G's carrier set as state set with transitions $f \xrightarrow{a|af} \partial_a f$.*

Of course in general, this invertible automaton will be infinite, but certainly $\mathcal{S}(\mathcal{C}_G)$ is a group and ismorphic to G. The more interesting case is when G may be represented in terms of a finite automaton. Toward this end, call G *finite-state* if for all $f \in G$, the number of residuals $\partial_w f$ is finite. If G is self-similar, finite-state, and finitely generated, we can construct the *group automaton* $\mathcal{A}_G$, a binary Mealy automaton, just like the complete group automaton, but with state set restricted to the collection of all residuals of the generators of G. Of course, the group generated by $\mathcal{A}_G$ is isormorphic to G. One need to be careful, however; the semigroup may be different. Pleasantly, $\mathcal{A}_G$ is minimal by construction.

The following lemma is adapted from [17].

Lemma 3. *For any admissible (A, r), the complete automaton $\mathcal{C}$ over A and r has only finitely many subautomata, each of which has finitely many states.*

Proof. Fix some state $v \in \mathbb{Z}^m$. Since residuation is an affine map, we may write every descendent w of v as a monic polynomial over A:

$$w = A^n v + \sum_{i=0}^{n-1} d_i A^i r$$

where $d \in \{-1, 0, 1\}$. This comes down to using a redundant numeration system similar to base 2, but with a symmetric digit set. Letting $\|\cdot\|$ denote both the norm over $\mathbb{Q}^m$ and the induced matrix norm, we have the bound

$$\|w\| \leq \|A^n\| \, \|v\| + \|r\| \sum_{i=0}^{n-1} \left\| A^i \right\|$$

by the triangle inequality. Taking the limit as $n \to \infty$, $\|A^n\|$ goes to 0, as $\|A^n\| = \lambda^n$, where λ is the spectral radius of A (and $\lambda < 1$ by Theorem 3). Thus, in the limit, we have a bound on $\|w\|$ that is independent of $\|v\|$.

Put differently, this says that, eventually, all descendents of v are bounded from above by some expression independent of v. Since any ball of finite radius around 0 in $\mathbb{Z}^m$ is finite, this implies that there must be finitely many strongly connected components in $\mathcal{C}$, each of finite order. □

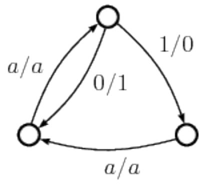
Lemma 4. *One may compute all such subautomata.*

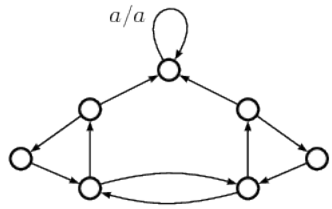
Proof. Brute force search in the finite ball around 0. The radius of the ball may be effectively computed from λ. □

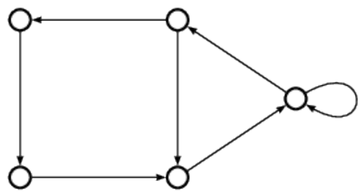
Example 2. *The following automata represent all subautomata of the complete automaton generated with residual matrix*

$$A = \begin{bmatrix} -1 & 1 \\ -\frac{1}{2} & 0 \end{bmatrix}$$

and residual vector $r = [-1, -\frac{3}{2}]^T$.







The top state of A_2^3 (seen earlier in example 1) corresponds with $e_1 \in \mathbb{Z}^2$. One can check that $A(Ae_1 - r) = e_1$, corresponding with going around the shorter loop.

Definition 7. *For a residual matrix A and residual vector $r = A \cdot e_1$, the* principal automaton *is the automaton generated from closure of e_1 under residuation defined by the pair (A, r).*

Membership *is decidable in the Abelian case*

Definition 8. *The automorphism* Membership *problem takes as input two automata, $\mathcal{A}$ and $\mathcal{B}$, a distinguished automorphism $f \in \mathcal{A}$ corresponding to some state p, and $\mathcal{B}$'s residual matrix and vector A and r, and ouputs whether $f \in S(\mathcal{B})$.*

State-closed, finite-state automorphisms admit the natural computational representation as automata. It is worth noting that our Membership problem differs from the Word Problem: the word problem considers candidates given as words over the generators of a semigroup S; here our candidates are raw automorphisms.

Returning to Membership, one thus needs to check if there is some product automaton

$$\mathcal{D} = \mathcal{B}_{p_1} \times \mathcal{B}_{p_2} \times \ldots \times \mathcal{B}_{p_n}$$

that implements f. We have no computable bound on n, so a priori this only semidecidable (this is a running theme).

Theorem 5. *Automorphism* Membership *in $S(\mathcal{B})$ for a principal Abelian automaton $\mathcal{B}$ is decidable.*

Proof. Given as input an automaton $\mathcal{A}$ and a principal Abelian automaton $\mathcal{B}$, we determine if $f = \mathcal{A}(p)$ is in the semigroup generated by $\mathcal{B}$.

It suffices to simply check if f is equivalent to any automorphism in any of the subautomata of $\mathcal{C}_{\mathcal{B}}$.

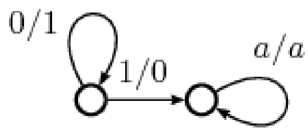


Figure 3: The adding machine

Consider the complete automaton $\mathcal{C}$ for $\mathcal{B}$. Define g to be the automorphism defined by $\mathcal{D}$. After minimization, $\mathcal{D}$ produces a subautomaton of $\mathcal{C}$ that consists of a "transient part" and a copy of $\mathcal{B}$ (there may be strongly connected components in the transient part, but they are not subautomata). Hence, there is some word w such that $\partial_w g$ is just a single state in the copy of $\mathcal{B}$.

□

Membership *is open in the general case*

The decidablity of membership is arguably the most important open problem relevant to this thesis. Its decidablity would imply the decidability of IsGroup, as one could simply check that the inverse of each generator is contained in $\mathcal{S}(\mathcal{A})$.

IsGroup

Definition 9. *The* IsGroup *decision problem takes as input an automaton $\mathcal{A}$ and answers the question "is $S(\mathcal{A}) = G(\mathcal{A})$?"*

Example 3. *There exist automata for which $S(\mathcal{A})$ is not a group; the adding machine in figure 3 is such a machine. Viewing the input string as a natural number in reverse binary, one can see that it adds one to the input.*

Proposition 1. IsGroup *is decidable in the Abelian case.*

This follows immediately from a decidable automorphism Membership problem: simply check for membership of the identity function and the inverse of each generator.

Despite a fair amount of effort, IsGroup is still open in the general case. Our work on Knapsack and Membership represent much partial work toward a solution.

Knapsack *is undecidable for automaton semigroups*

We follow a proof strategy inspired by [11].

Exponential Equations

Suppose $\mathcal{X}$ is a countably infinite set of variables.

Definition 10. *An* exponential equation *E over a semigroup S is an equation of formal products of the form*

$$s_1^{x_1} s_2^{x_2} \cdots s_l^{x_l} = t_1^{y_1} t_2^{y_2} \cdots t_l^{y_l}$$

where each x_i, y_i is in $\mathcal{X}$ and each s_i, t_i is in S. Note that we do not require the x_i's and y_i's to be distinct.

Denote by $\mathrm{Var}(E)$ the set of all variables appearing in E.

Definition 11. *For a finite set X and semigroup element s, the set of* X-solutions *of the equation E is the collection of maps*

$$S_X(E = s) = \{v : X \to \mathbb{N} \mid s_1^{v(x_1)} s_2^{v(x_2)} \cdots s_l^{v(x_l)} = s \text{ in } S\}$$

The upcoming reduction involves taking direct products of semigroups, and so we will need the following strategy to transform exponential expressions over a semigroup S to equivalent exponential expressions over a direct product containing S.

To this end, suppose we have exponential expressions E_i for $i = 1 \ldots n$, each with corresponding semigroups S_i. In each E_i, replace the base element s_i with the element

$$(\underbrace{1, \ldots, 1}_{i-1}, s_i, \underbrace{1, \ldots 1}_{n-i}) \in \prod_i S_i.$$

Then take the formal product of each new E_i.

Undecidability of Knapsack

Definition 12. *We define the* Knapsack Problem *as follows: given as input generators $g_1 \ldots g_k$ and a target semigroup element g, do there exist natural numbers $a_1 \ldots a_k$ such that*

$$g_1^{a_1} \cdots g_k^{a_K} = g$$

This decision problem is named as such due to its similarity with the *unbounded knapsack problem*, which asks, for items indexed from $i = 1$ to n, each with value v_i and weight w_i, to maximize the sum

$$\sum_{i=0}^{n} v_i x_i$$

where x_i denotes the number of times item i is used. Additionally, it is required that $\sum_{i=0}^{n} w_i x_i \leq W$ for some weight W, with each $x_i \geq 0$.

In our variant, the indexed items are group elements, each with weight g_i, and repetition a_i. Rather than seeking to maximize the formal product, the variant asks to hit a target value.

Definition 13. *The* Generalized Knapsack Problem *has as input generators $g_1 \ldots g_k, h_1, \ldots h_l$, and has as output whether there exist natural numbers $a_1 \ldots a_k, b_1, \ldots, b_l$ such that*

$$g_1^{a_1} \cdots g_k^{a_k} = h_1^{b_1} \cdots h_l^{b_l}$$

We demonstrate that the Generalized Knapsack Problem is undecidable in the class of automaton semigroups by reducing from Hilbert's tenth problem. The undecidablity of the Knapsack Problem easily follows.

Definition 14. *We define the decision problem* Hilbert *as following: "given a polynomial $P(x_1, \ldots, x_n) \in \mathbb{Z}[x_1, \ldots, x_n]$ and an integer a, do there exist values $y_i \in \mathbb{N}$ such that $P(y_1, \ldots, y_n) = a$?"*

It is well-known that there exist polynomials for which HILBERT is undecidable, see [13] for details.

One can show that the Heisenberg semigroup

$$H_3(\mathbb{N}) = \left\{ \begin{bmatrix} 1 & a & c \\ 0 & 1 & b \\ 0 & 0 & 1 \end{bmatrix} ; a, b, c \in \mathbb{N} \right\}$$

is an automaton semigroup [9]. Moreover, the class of automaton semigroups is closed under direct products, proven by Cain in [2]. We denote elements of $H_3(\mathbb{N})$ as

$$H_{x,y,z} = \begin{bmatrix} 1 & x & z \\ 0 & 1 & y \\ 0 & 0 & 1 \end{bmatrix}$$

Proposition 2. *There exist fixed constants $d, e \in \mathbb{N}$ and an exponential equation E of the form*

$$s_1^{x_1} s_2^{x_2} \cdots s_n^{x_n} = t_1^{y_1} t_2^{y_2} \cdots t_n^{y_n}$$

with $s_i, t_i \in G = H_3(\mathbb{N})^d \times \mathbb{Z}^e$ for which the GENERALIZED KNAPSACK *problem is undecidable.*

Proof. From the input polynomial, $P(x_1, \ldots, x_n)$ and target value a, we separate the positive and negative terms of P to obtain an equation of the form

$$P_+(x_1, \ldots, x_n) = P_-(x_1, \ldots, x_n, a)$$

where every coefficient in P_+ and P_- is positive. From this equation, we construct a system S of equations, where each equation has one of the following forms: $x \cdot y = z$, $x + y = z$, $x = c$ (for $c \in \mathbb{Z}$). We will have that the equation $P(x_1, \ldots, x_n) = a$ has a solution in $\mathbb{N}$ if and only if the system of equations $S_a = S \cup \{x_0 = a\}$ has a solution in $\mathbb{N}$. Let X be the set of variables that occur in S_a.

Take a natural number a (the input of the reduction). Assume that S_a contains d equations of the form $x \cdot y$, and e many equations of the form $x + y = z$ or $x = c$. We enumerate these equations as $E_1, \ldots E_{d+e}$, where the first d equations are of the form $x \cdot y = z$. Then set $G_i = H_3(\mathbb{N})$ for each $i \leq d$ and set $G_i = \mathbb{N}$ for each $i > d$. For every i, we define an element g_i and an exponential expression E_i over G_i as follows:

Case 1: $E_i = (x \cdot y = z)$. Thus we have $G_i = H_3(\mathbb{N})$. Set g_i to be the identity matrix in $H_3(\mathbb{N})$ and consider the following equation:

$$H_{1,0,0}^x = \begin{bmatrix} 1 & x & xy \\ 0 & 1 & y \\ 0 & 0 & 1 \end{bmatrix} = \begin{bmatrix} 1 & x & z \\ 0 & 1 & y \\ 0 & 0 & 1 \end{bmatrix} = H_{1,0,0}^z H_{0,0,1}^y H_{1,0,0}^x$$

A mapping $v : X \to \mathbb{N}$ is a solution if and only if $v(x)v(y) = v(z)$.

Case 2: $E_i = (x + y = z)$ and so $G_i = \mathbb{Z}$. Set $g_i = 0$ and consider the equation (written additively in the group $\mathbb{Z}$)

$$x + y - z = 0$$

Then a mapping $v : X \to \mathbb{Z}$ is a solution if and only $v(x) + v(y) = v(z)$.

Case 3: $E_i = (x = c)$. (This includes our distinguished equation $x_0 = a$). We have $G_i = \mathbb{Z}$. Then set $g_i = c$ and $E_i = x$. Then as usual, a mapping v is a solution if and only if $v(x) = c$.

Finally, define E to be the direct product of the E_i's $\prod_{i=1}^{d} E_i$ and define $g = (g_1, \ldots g_{d+e})$. Then a mapping v is a solution to the equation $E = g$ if and only if v is a solution to the system of equations S. □

Theorem 6. Generalized Knapsack *is undecidable in the class of automaton semigroups.*

Proof. For all $d, e \in \mathbb{N}$, $H_3(\mathbb{N})^d \times \mathbb{Z}^e$ is an automaton semigroup. It follows that Generalized Knapsack is undecidable for automaton semigroups.

□

A monoid with decidable Word Problem *and undecidable* IsGroup

We establish the existence of a monoid with decidable Word Problem, but undecidable IsGroup. One may take this result as an intermediate step toward the decidability of the IsGroup problem for automaton semigroups.

Preliminaries

Here we take a *Turing machine* to be a 6-tuple, $(Q, \Sigma, \Gamma, \delta, q_0, q_{accept}, q_{reject})$, where Q, Σ, Γ are all finite sets. $\delta : Q \times \Sigma \to Q \times \Gamma \times \{L, S, R\}$ is the transition function, Q is the state set, Σ is the input alphabet, $\Gamma \supseteq \Sigma$ is the tape alphabet, b is some blank symbol, with $b \in \Gamma - \Sigma$, q_{accept} is the unique accepting final state, and q_{reject} the single rejecting final state.

We further define a Turing machine *configuration* to be a triple $(u, q, v) \in \Gamma^* \times Q \times \Gamma^*$. Here, u denotes the tape contents to the left of the tapehead, q is the current state, and v begins at the tapehead and extends to the right.

A configuation C for a TM M is said to *yield* configuration C' if M can step directly from C to C'.

For a Turing machine M, take C_M to be the set of all valid configurations of M. Then define CG_M to be the graph (C_M, E), where $(u, v) \in E$ if and only if u yields v in M.

Definition 15. *Define the* canonical computation of M on w.

$$\mathsf{canon}(M, w) : \mathcal{T} \times \Sigma^* \to C_M^* \cup C_M^\omega$$

to be the function that maps input w to the sequence of configurations M takes on while computing over w. Note that $\mathsf{canon}(M, w)$ *will be a finite sequence if and only if M halts on w.*

Certainly, not every configuration in C_M will be along the sequence $\mathsf{canon}(M, w)$. Which is to say, there are unreachable configurations.

Informally, a *self-verifying Turing machine S* is one that, at every step, verifies that the current configuration lies upon the canonical computation. If S finds that this is not the case, S immediately rejects. Otherwise, the computation steps forward a single step.

In the configuration graph CG_S, there is a path extending from each valid starting configuration (ϵ, q_0, w) for $w \in \Sigma^*$. The remaining states form a countably infinite star graph with q_{reject} as the center.

Proposition 3. *There is a computable[1] function* sv *that maps Turing machines to equivalent self-verifying Turing machines.*

Speaking informally, the new Turing machine S maintains on its tape a time step t, the original input w to our original Turing machine M, and M's current configuration C. After simulating each step of M, S will re-run M on w for the first t steps. If this computation results in M sitting in configuration C, S increments t, replaces C with the new configuration C', and proceeds to the next step of M. Otherwise, S immediately rejects.

One can show that any invalid configuration will be caught after finitely many steps, as S manually verifies each and every configuration. For further reading, see [4], [5], and [20].

The submonoid in question

Define the Turing machine $M = (Q, \Sigma, \Gamma, \delta, q_0, F)$ to operate only on the blank tape; for all $s \in \Sigma$, $\delta(q, s) = (q_{reject}, b, S)$

Then take the ambient Abelian group $G_M = (C_M, \cdot)$ whose carrier set is all configurations of M. For c, c' in G_M, we have $c = c'$ if and only if c yields c'.

Proposition 4. $G_{\mathsf{sv}(M)}$ *has a decidable word problem.*

Proof. For every word w in $G_{\mathsf{sv}(M)}$, there exist nonnegative integers a, r such that $w = q_{accept}^a q_{reject}^r$. Further, we may compute a and b. Recall that $\mathsf{sv}(M)$ maintains a program counter p to the left of the input. So we may simply run M for the first p steps and then check for configuration equality.

So then given two words w_1, w_2 in $G_{\mathsf{sv}(M)}$, simply compute a_1, r_1, a_2, r_2. Then w is in $G_{\mathsf{sv}(M)}$ if and only if $a_1 = a_2$ and $r_1 = r_2$. $\square$

Proposition 5. *If s is the start configuration of the Turing machine, it is undecidable whether $\langle s \rangle$ is a group.*

Proof. It is well known that the following language is undecidable

$$\textsc{halts} = \{\langle M \rangle \mid \text{TM } M \text{ halts on } \epsilon\}$$

and so we reduce from $\textsc{halts}$. Given as input a TM M, we use an oracle for IsGroup as follows: first, compute $\mathsf{sv}(M)$, and then consider $G_{\mathsf{sv}(M)}$. Let s be the starting configuration for M on ϵ.

If M halts, then the submonoid generated by s is the trivial group. If M hangs, then $\langle s \rangle$ is the free monoid of rank one. So then $\langle s \rangle$ is a group if and only if M halts. Since $\mathsf{sv}(M)$ and M are equivalent, we are done. $\square$

[1] The reader may interested to find that sv is in fact primitive recursive.

Much partial work has been attempted toward a proof of the solvability or unsolvability of the IsGroup problem in the general case. The monoid presented here serves a sort of bound for this problem; optimistically suggesting that perhaps the class of automaton semigroups is not so big as to have an undecidable IsGroup.

There is also a rich area of smaller subclasses of automata to consider. Godin proved the decidablity of Knapsack for the class of bounded automata in [7]. It is natural to then consider the decidability of other decision problems, such as IsGroup, Isomorphism, and others.

One may also consider decision problems from a group presentation angle; all automaton semigroups are recursively presented. If we restrict our considerations to automaton semigroups whose presentations are regular, or context-free, does this affect the decidability of various decision problems? One suspects these questions are probably quite hard.

Additionally, most of the semigroup-theoretic decision properties listed in [3] remain open in the class of automaton semigroups. Notably, determining the decidability of Markov properties would be of great help to future work.

REFERENCES

[1] L. Bartholdi and P. V. Silva. Groups defined by automata. *CoRR*, abs/1012.1531, 2010.

[2] A. J. Cain. Automaton semigroups. *TCS*, 410(47–49):5022–5038, 2009.

[3] A. J. Cain and V. Maltcev. Decision problems for finitely presented and one-relation semigroups and monoids. *International Journal of Algebra and Computation*, 19(6):747–770, 2009.

[4] M. Davis. A note on universal turing machines. *Annals of Mathematics studies*, 34:167–175, 1956.

[5] M. Davis. The definition of universal turing machine. *Proceedings of the American Mathematical Society*, 8:1125–1126, 1957.

[6] Pierre Gillibert. The finiteness problem for automaton semigroups is undecidable. *CoRR*, abs/1304.2295, 2013.

[7] T. Godin. Knapsack problem for automaton groups. *HAL*, 2016.

[8] John E. Hopcroft, Rajeev Motwani, Rotwani, and Jeffrey D. Ullman. *Introduction to Automata Theory, Languages and Computability*. Addison-Wesley Longman Publishing Co., Inc., Boston, MA, USA, 2nd edition, 2000.

[9] R. Kravchenko I. Bondarenko. Finite-state self-similar actions of nilpotent groups. *Geometriae Dedicata*, 163(1):339–348, 2013.

[10] O. Kharlampovich, B. Khoussainov, and A. Miasnikov. From automatic structures to automatic groups. *ArXiv e-prints*, July 2011.

[11] D. Konig, M. Lohrey, and G. Zetzsche. Knapsack and subset sum problems in nilpotent, polycyclic, and co-context-free groups. *Contemporary Mathematics*, 2015.

[12] Claiborne G. Latimer and C. C. MacDuffee. A correspondence between classes of ideals and classes of matrices. *Annals of Mathematics*, 34(2):313–316, 1933.

[13] Yuri V. Matiyasevich. *Hilbert's Tenth Problem*. MIT Press, Cambridge, MA, USA, 1993.

[14] Y. Muntyan. *Automata Groups*. PhD thesis, Texas A&M University, May 2009.

[15] V. Nekrashevych. *Self-Similar Groups*, volume 117 of *Math. Surveys and Monographs*. AMS, 2005.

[16] V. Nekrashevych and S. Sidki. *Automorphisms of the binary tree: state-closed subgroups and dynamics of 1/2-endomorphisms*. Cambridge University Press, 2004.

[17] T. Okano. Invertible binary transducers and automorphisms of the binary tree. Honors Thesis, Carnegie Mellon University, May 2015.

[18] Jacques Sakarovitch. *Elements of Automata Theory*. Cambridge University Press, New York, NY, USA, 2009.

[19] J.-P. Serre. *Arbres, Amalgames, SL_2*. Société Mathématique de France, Paris, France, 1977.

[20] J. C. Shepherdson. Machine configuration and word problems of given degree of unsolvability. *Z. f. Math. Logik u. Grundlagen d. Mathematik*, 11:149–175, 1965.

[21] Z. Sunic and E. Ventura. The conjugacy problem in automaton groups is not solvable. *Journal of Algebra*, 364(148–154), 2012.

[22] K. Sutner. Invertible transducers, iteration and coordinates. In S. Konstantinidis, editor, *CIAA*, volume 7982 of *LNCS*, pages 306–318. Springer Berlin, 2013.

[23] K. Sutner and K. Lewi. Iterating inverse binary transducers. *Journal of Automata, Languages, and Combinators*, 17(2–4):293–313, 2012.