Automata and Automatic Groups

A Final Project for the Second Course in the Theory of Groups

Gabriel Ong

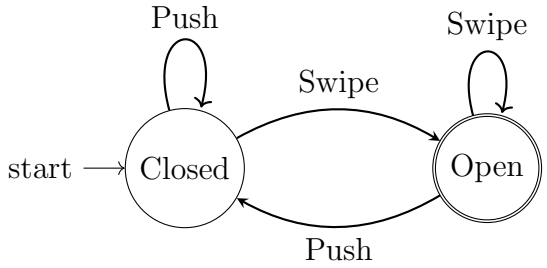
with Brady Nichols, Eitan Marcus, and Juan Atehortúa

May 2022

MATH 3602: Advanced Topics in Group Theory

1 Computing

Computers have become ubiquitous in our daily lives, yet the theory behind them is often poorly understood. Most modern day computers are in fact examples of finite state automata, simple machines whose output depends on both the machine's current state and its input. Let us consider the a turnstile in a public transportation system.



The diagram above, known as a state transition diagram, tells us how the turnstile behaves in responses to pushes and swipes. A more careful examination of the diagram tells us that the turnstile behaves as expected. A turnstile starts in the closed position and remains closed when pushed without swiping one's transit card. It is only when the transit card is swiped that the turnstile opens. In the open state, an additional swipe keeps the turnstile open but pushing and going through the turnstile returns it to its closed state for the next commuter. The same information can be represented by the following table, known as a next-state table.

	Swipe	Push
$\rightarrow$ Closed	Open	Closed
$\circ$ Open	Open	Closed

Let us now be more precise.

Definition 1.1 (Finite State Automata)**.** A finite state automata $A = (I, S, s_0, B, N)$ consists of the following: a set I, the input alphabet of symbols; a set S, the states of the automata; a designated $s_0 \in S$, the initial state of the automata; a next state function $N : S \times I \rightarrow S$ associating a next state to a given current state and input alphabet.

Returning to the example above, our set of symbols are $\{\text{Swipe}, \text{Push}\}$, our set of states are $\{\text{Closed}, \text{Open}\}$, our initial state is Closed (as indicated by the arrow), our accepting state is Open (as indicated by the double circle in the state transition diagram or the circle in the

next-state table), with our next-state function defined by the next-state table.

Denote I^* the set of strings formed from letters in I. Our finite state automata A partitions I^* into strings that lead to an accepted state and those that do not when input into the automata A in its starting state. This leads us to the following two closely related definitions.

Definition 1.2 (w Accepted)**.** Let $w \in I^*$ and A a finite state automata. w is accepted by A if and only if A goes to an accepting state when the symbols of w are input to A in sequence when A is in its initial state.

Definition 1.3 (Accepted Language)**.** The language accepted by A is the set

$$L(A) = \{w \in I^* | w \text{ is accepted by } A\}.$$

In the case of the turnstile, it is easy to see that any string in $\{\text{Swipe}, \text{Push}\}^*$ ending in Swipe will be accepted by the automata – regardless of the state the automata is in, an input of Swipe will lead to the accepted state.

We focus our discussion on deterministic finite automata, abbreviated DFA.

Definition 1.4 (Deterministic Finite Automata)**.** A finite state automata A is deterministic if and only if it has exactly one start state and no two edges leaving a state have the same label.

We can now define regular languages, which will play a large part in our latter discussions.

Definition 1.5 (Regular Language)**.** A language L is regular if and only if it is the accepted language of some deterministic automata A.

It seems intuitively true that for any language L we can construct some automata A such that $L = L(A)$. This is not true. One of the easiest ways to determine if a language is regular is if the automata will need infinite memory to determine if a word is accepted (Meier, 2008).

Lemma 1.1 (Pumping)**.** *Let $L(A)$ be the regular language accepted by the automaton A. There exists an integer $n \geq 1$ such that any $x \in L(A)$ of length greater than n can be written $x = uvw$ where v is a nonempty word, u is of length less than n, and $wv^iw \in L(A)$ for all $i \geq 0$.*

Proof. Let A be a DFA with accepted language $L(A)$. Set $n = |S|$ and $x \in L(A)$ a word of length greater than n. Since A is a DFA there is a unique path starting at s_0 and ending at an accepted state indexed by the letters of x. Since this is a path of length greater than n, Dirichlet's Box Principle indicates that there must be some state $z \in S$ visited at least twice.

Let u be the substring of x indexing the path from s_0 to the first time it intersects z, v a substring of x indexing the path between the first and last occurrences of z, and w a substring of x indexing the path between z to an accepting state. Note that v and w may be empty paths.

It follows that u is shorter than n since it cannot visit any vertex more than once and $uv^iw \in L(A)$ since this describes a path from s_0 to z, repeating a cycle from z to z i times, then a path z to an accepting state, which still ends in an accepting state. This gives us our claim. □

Let us now see this lemma in action through the following example.

Proposition 1.2 ($\{a^nb^n|n \geq 1\}$ is Not a Regular Language)**.** *The language* $\{a^nb^n|n \geq 1\}$ *is not a regular language.*

Proof. Suppose to the contrary that $\{a^nb^n|n \geq 1\}$ is a regular language and set $|S| = n$ for some automata A with language $\{a^nb^n|n \geq 1\}$. By Dirichlet's Box Principle, there must be some state attained at least twice, namely there exists a closed loop on the path indexed by the word a^{n+1} say of length k. By the pumping lemma we can repeat this loop so $a^{n+k+1}b^{n+1}$ leads to an accepting state just as $a^{n+1}b^{n+1}$ does, a contradiction of $\{a^nb^n|n \geq 1\}$ being a regular language. □

This is to say that for an automata that accepts strings of the form $ab, a^2b^2, a^3b^3, \ldots$ then then the automata must accept some set strictly containing $\{a^nb^n|n \geq 1\}$, so $\{a^nb^n|n \geq 1\}$ cannot be a regular language. Let us now define the following terms.

Definition 1.6 (Cone Type of a Word)**.** Let A be an automata and $L(A) \subseteq I^*$ its language. Suppose $x \in I^*$ its cone

$$\text{Cone}(x) = \{w \in I^* | xw \in L(A)\}.$$

Definition 1.7 (Cone Type of a Language)**.** Let A be an automata and $L(A) \subseteq I^*$ its language. The cone type of $L(A)$ is

$$\text{Cone}(L(A)) = \{\text{Cone}(w) | w \in I^*\}.$$

We can think of cone type of a word $w \in I^*$ as the set of extensions x such that wx is in the language, namely as a set of paths in the automaton from the state indicated by w to any one of the accept states. The cone type of a language is then the set of possible extensions of all words such that the extended word is accepted. Intuitively, we can think of this as the set of geodesic paths from a given state of the automata to an accept state. Returning to our example of the turnstile, the cone type of the language will be the set of all words ending in Swipe since Swipe always takes the automata to an accepted state regardless of the current state. This means that our turnstile has only one cone type. Let us now prove a theorem to conclude our discussion of automata.

Theorem 1.3 (Myhill-Nerode)**.** *Let I be a collection of symbols and $L \subseteq I^*$ a language. L is a regular language if and only if* $\mathrm{Cone}(L)$ *is finite.*

Proof. ($\Longrightarrow$) Suppose L is regular, we show its cone type is finite. If L is regular then there exists a DFA that accepts it. For each state $s \in S$ let A_s be the automata formed by making s the start state. For a word $w \in I^*$ the cone type of w is completely determined by the state s_w – the state attained after the letters of w are input sequentially into A_s. We thus have $\mathrm{Cone}(w) = L(A_{s_w})$ and have that the number of cone types for L is at most the number of states $|S|$.

($\Longleftarrow$) Suppose $\mathrm{Cone}(L)$ is finite, we show it is regular. Let $\mathrm{Cone}(L)$ be finite. We construct an automata A declaring its states to be the cone types of L. Set $s_0 = \mathrm{Cone}(\varepsilon)$ where $\varepsilon \in I^*$ is the empty word. The accepting states of A are the cones containing ε since if w is accepted then εw is as well. Fix some word $w \in I^*$ and add a directed edge corresponding to the input $x \in I$ from the state given by $\mathrm{Cone}(w)$ to $\mathrm{Cone}(wx)$. We want to show this is well-defined, that is if $\mathrm{Cone}(w) = \mathrm{Cone}(\hat{w})$ then $\mathrm{Cone}(wx) = \mathrm{Cone}(\hat{w}x)$ as

$$\mathrm{Cone}(wx) = \{w' \in I^* | xw' \in \mathrm{Cone}(w)\} = \{w' \in I^* | xw' \in \mathrm{Cone}(\hat{w})\} = \mathrm{Cone}(\hat{w}x).$$

We know that A has finitely many states since there are finitely many cone types and is deterministic by construction since there is an output for any input from I at any given state of the automata. Finally, we verify that L is the language accepted by the automata A we constructed above. Observe that $w \in L$ if and only if $\varepsilon \in \mathrm{Cone}(w)$ since an accepted word w can be extended by ε an arbitrary number of times and still be accepted. Let $w \in I^*$ be an arbitrary word. The letters of w index a sequence of states in A which by construction ends at $\mathrm{Cone}(w)$. But note that

$$w \in L \Longleftrightarrow \mathrm{Cone}(w)\text{is an accepting state} \Longleftrightarrow \varepsilon \in \mathrm{Cone}(w)$$

which is what we established above. Thus L is precisely the regular language of A. □

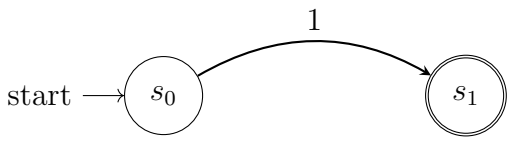
2 Automatic Groups

Let us slightly adapt our definition of a language.

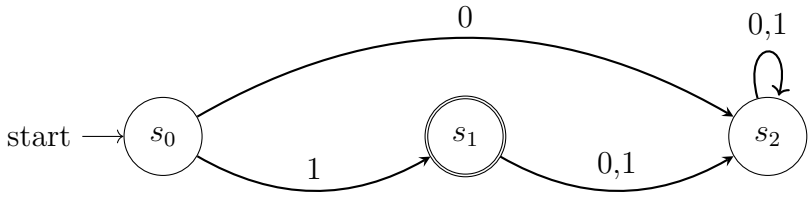
Definition 2.1 (Padded Language)**.** Let I be a set of symbols with $\$ \notin I$. L is a padded language if and only if $L \subseteq (I \cup \{\$\})^* \setminus \{\$\}$.

In particular, padded languages give us the ability to read two words in I^* of unequal length at the same rate. Suppose $u, v \in I^*$ where $u = u_1u_2 \dots u_4$ and $v = v_1v_2 \dots v_5$. We can rewrite these words in $(I \cup \{\$\})^* \setminus \{\$\}$ such that they are the same length, in particular by writing u as $u_1u_2u_3u_4\$$ and writing v as above – these are now both words of length 5. Padded languages in fact give rise to a rich theory of non-deterministic automata.

Recall that a deterministic finite automata has a well-defined next state function for every state $s \in S$ and every letter $i \in I$. This need not be the case in nondeterministic automata: for a given state there can be any number of transitions corresponding to a particular input $i \in I$. Consider the following state transition diagram:



with inputs $I = \{0, 1\}$. Let us first observe that this automata is nondeterministic, for giving an input of 0 at s_0 does not induce a transition of the automata. Moreover, we note that the accepted language of this automata is the binary string "1" since to transition from the initial state to the accepted state there can only be a 1. Though it may not be intuitively obvious, it is a remarkable fact that we can actually construct a deterministic automata with the same accepted language. Consider the following state transition diagram:



this in fact gives us a deterministic automata that accepts the binary string "1". It should be noted, however, that the nondeterministic automata in this case was much less complicated than its deterministic counterpart. Generally speaking nondeterministic autoamata are

much more powerful than deterministic automata since it is an easy way to model processes with multiple outcomes. But since computers are only capable of modeling deterministic automata, that will be the focus of our further discussions.

We are now ready to define automatic groups.

Theorem 2.1 (Automatic Groups (Farb, 1992)). *Let $G = \langle I|R\rangle$ where $I = \{i_1, i_2, \ldots, i_k\}$ generates G as a monoid, namely $I = S \cup S^{-1}$ for S a generating set of G. G is an automatic group if and only if the following conditions hold.*

- *There is a regular language $L \subseteq I^*$ with automata A such that $\pi : L \to G$ is surjective.*
- *The following padded languages are regular:*

$$\begin{aligned} L_= &= \{(u,v)|u,v \in L, \overline{u} = \overline{v}\} \\ L_{i_1} &= \{(u,v)|u,v \in L, \overline{u} = \overline{vi_1}\} \\ L_{i_2} &= \{(u,v)|u,v \in L, \overline{u} = \overline{vi_2}\} \\ &\vdots \\ L_{i_k} &= \{(u,v)|u,v \in L, \overline{u} = \overline{vi_k}\} \end{aligned}$$

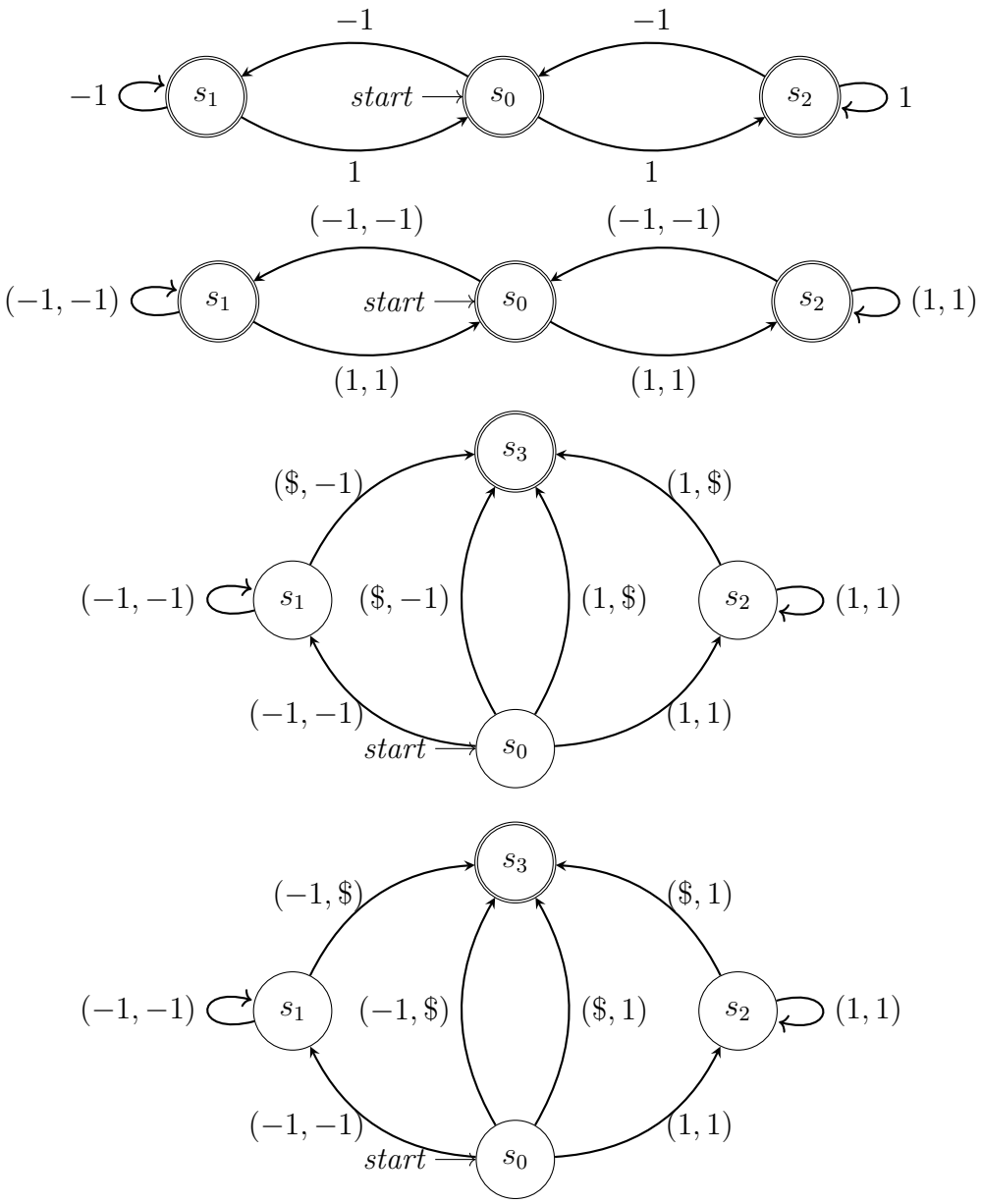
Note that $L_=$ is the language of a finite state automata $A_=$ checking if two words in L correspond to the same group element – this is in fact the equality problem. Whereas L_{i_l} is the language of an automata A_{i_l} checking if two elements of a group differ by right multiplication of the generator i_l for all $1 \leq t \leq k$.

Definition 2.2 (Automatic Structure). The family of automata $(A_=, A_{i_1}, \ldots, A_{i_k})$ is the automatic structure of G.

Let us consider some examples of automatic groups.

Example 1 (Finite Groups are Automatic). *Let $G = (\{g_1, g_2, \ldots, g_n\}, *)$. Set $I = \{g_1, g_2, \ldots, g_n\}$ and $L(A) = I^*$ the set of all words in the letters in I. Since G is a finite group any word in I^* corresponds to a letter of G under group multiplication, so we can construct our automata by setting the elements of G as the accept states.*

Example 2 ($\mathbb{Z}$ is Automatic). *We show that $\mathbb{Z} = \langle 1\rangle$ is an automatic group. We do so by constructing the word acceptor, equality checker, and comparator automata. Consider the following set of finite state machines which correspond to the word acceptor, equality checker, and comparators for the generators 1 and -1 respectively.*



We remark that an automatic group can have multiple automatic structures even for a fixed generating set (Farb, 1992). We now want to discuss an alternative definition of automatic groups. To do so, let us recall some notions from our second course in the theory of groups. Let $G = \langle I|R\rangle$ be a finitely generated group with generating set I and relator set R. Let $\Gamma(G, I)$ be the Cayley Graph of the group G with respect to generating set I. A path $u \subseteq \Gamma(G, I)$ and define a function $\gamma_u : \mathbb{N} \to \Gamma(G, I)$ such that for $u = u_1u_2 \dots u_n$ with $u_i \in S$

$$\gamma_u(t) = \begin{cases} u_1u_2 \dots u_t & t \leq n \\ u_1u_2 \dots u_n & t > n \end{cases}$$

where $\overline{u} \in G$ is the element represented by the word u in the free group represented by the

word u in the generators. Moreover, recall that we have already shown that $(\Gamma(G,I), d_I)$ is a metric space using the word metric.

Definition 2.3 (k-Fellow Traveller Property)**.** Let $u, v \subseteq \Gamma(G,S)$ be paths. u and v satisfy the k-fellow traveller property if and only if there exists $k \in \mathbb{R}$ such that $d_S(\overline{\gamma_u(t)}, \overline{\gamma_v(t)}) \leq k$ for all $t \geq 0$.

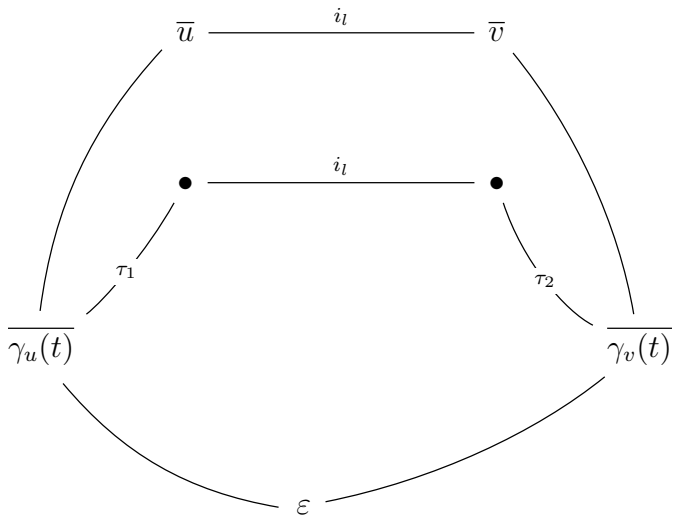
Namely two paths in a metric space satisfy the k-fellow traveller property if and only if the distance between points on two paths at any given time are bounded above by some constant $k \in \mathbb{R}$.

Let us now re-define automatic groups through the following proposition.

Proposition 2.2. *Let G be a group generated as a monoid by I. G is automatic if and only if the following hold:*

- *G has a word acceptor A with regular language $L(A) \subseteq I^*$.*
- *There exists $k \in \mathbb{R}$ such that if $u, v \in L(A)$ represent elements of G such that $d_I(u,v) = 1$ then the paths u and v satisfy the k-fellow traveller property.*

Proof. ($\Longrightarrow$) Suppose G an automatic group generated by a monoid $I = \{i_1, \ldots, i_k\}$. From the definition of an automatic group, we know G has a word acceptor so it remains to show the second property. Choose $C \in \mathbb{N}$ such that $C > \max\{|S_=|, |S_{i_1}|, \ldots, |S_{i_k}|\}$ where $S_=$ is the set of states in the equality checker and S_{i_l} the set of states in the word comparator for generator i_l. Suppose $d_I(\overline{u}, \overline{v}) = 1$ so we know that $u = vi_l$ for some $i_l \in I$. Let $s(t)$ denote the state the automata A_{i_l} is in after reading the prefixes $\gamma_u(t)$ and $\gamma_v(t)$ that may have been padded. There must exist a path from $s(t)$ to an accept state of length at most C since the longest path in any one of the automata in the automatic structure is strictly less than C. Note that this sequence of state transitions in A_{i_l} indexes a pair of paths $\overline{\gamma_u(t)}$ and $\overline{\gamma_v(t)}$ in $\Gamma(G,I)$ so that for some t fixed we have a path $\overline{\gamma_u(t)}$ through two vertices that differ by i_l to $\overline{\gamma_v(t)}$.



Note that paths τ_1, τ_2 are at most length $C-1$ since the paths to the pair of elements that differ by i_l is at most C. Thus we have $d_I(\overline{\gamma_u(t)}, \overline{\gamma_v(t)}) \leq 2(C-1)+1 = 2C-1$ giving us our k-fellow traveller property by setting $k = 2C-1$ for C chosen as above.

($\Longleftarrow$) Suppose G satisfies the conditions above. We construct an automata Diff that tracks the difference between two paths in $\Gamma(G, I)$ and use Diff to construct the each automata of the automatic structure of G in turn. We construct an the word comparator A_{i_l} generically, which easily extends to constructing the automatic structure of G. Let S_{i_l} be the set of states for the word comparator A_{i_l} and $s_{0i_l} \in S_{i_l}$ the start state of the word comparator A_{i_l}. Let $\mathcal{B}(e,k) \subseteq \Gamma(G,I)$ be the k-ball around the identity with respect to I. We construct the automata Diff by setting its states as $S_{i_l} \times S_{i_l} \times \mathcal{B}(e,k)$ with start state (s_{0i_l}, s_{0i_l}, e) and accept states $(s_{1i_l}, s_{2i_l}, i_l)$ for $s_{1i_l}, s_{2i_l} \in S_{i_l}$. We define state changes such that when the automata in the state (s_{1i_l}, s_{2i_l}, g) reads the word (x, y) in the padded alphabet the automata goes to the state $(s'_{1i_l}, s'_{2i_l}, x^{-1}gy)$ where s'_{1i_l} and s'_{2i_l} are the states the automata in the word acceptor A in states s_{1i_l} and s_{2i_l} are input with x and y respectively. Repeating this process for every $1 \leq l \leq k$ as well as the identity so we have constructed the automatic structure of G, implying it is automatic. $\square$

3 Some Examples

Let us now consider some upshots of automatic groups. Automatic groups are actually quite nice for the study of geometric group theory and possess several nice properties. For one, we can observe that the existence of the equality checker automata allows us to solve the equality problem – we can easily tell if words in I^* correspond to the same group elements. Recalling from class that the word problem is in fact equivalent to the equality problem (**Prop. 5.5 Meier, 2008**). However, we can say something stronger: the word problem solvable in $O(n^2)$ and construction of a Cayley graph in $O(n \log n)$ time (Farb, 1992). We prove the former, leaving the reader to find the proof of the latter in (Epstein, 1992).

Let us first prove the following lemma (Epstein, 1992).

Lemma 3.1 (Bounded Length Differences)**.** *Let G be an automatic group as above. There exists a constant C such that if $w \in L(A)$ is an accepted word and for $g \in G$ we have $d_I(g, \overline{w}) \leq 1$ then:*

- *g has a representative in I^* of length at most $|w| + C$ in $L(A)$*
- *if some representative of $g \in L(A)$ has length greater than $|w| + C$ then there are infinitely many representatives of $g \in L$.*

We remark that the proof utilizes many ideas present in the proof of the pumping lemma.

Proof. Let C be greater than the number of states in any one of the automata in the automatic structure, the word acceptor of G. Let $d_I(\overline{w}, g) \leq 1$ and $w' \in L$ such that $\overline{w'} = g$. One of (w, w') or (w', w) must be accepted either the equality checker or one of the comparator automata. If $|w'| > |w| + C$ then the automata undergoes more than C transitions and must thus visit some state s more than once by Dirichlet's Box Principle. Using the pumping lemma we can write $w' = xyz$ where w indexes the inputs that induce the state transitions between the first and last visit of the state s and easily shorten $w = xz$ which ends in an accepted state showing the first condition. Alternatively, one can lengthen w' arbitrarily since $xy^n z$ is accepted for all $n \in \mathbb{N}$. □

We are now ready to prove our desired theorem (Epstein, 1992).

Theorem 3.2 (Quadratic Time Word Problem)**.** *Let G be an automatic group. For any word $w \in I^*$ we can find a string in $\ell \in L$ such that $\overline{\ell} = \overline{w} = g$ for some $g \in G$ in time proportional to $|w|^2$.*

Proof. Let u be an accepted string of A_{i_t} and $i_t \in I$ a generator. We want to find an accepted string v such that represents $\overline{ux}$. Suppose our automata A_{i_t} accepts a string (u', v') where u' is a padded word representing u and v' a padded word representing $\overline{ui_k}$. We use the path traced to an accepting state in the automata to construct v.

Let X_0 be the set whose only element is s_{0i_t}, the initial state of the automata A_{i_t}. For $i > 0$ we inductively define T_i as the set of state transitions starting in X_{i-1} with label (x_i, y_i) where $y_i \in I \cup \{\$\}$ and

$$x_i = \begin{cases} u_i & i \leq |u| \\ \$ & i > |u|. \end{cases}$$

. Set X_i the set of states reached by T_i. Let $n \in \mathbb{N}$ be the smallest number such that X_n contains an accepting state of the comparator automata A_{i_t}. We can thus trace a sequence of state transitions in A_{i_t} from the start state to an accepting state. Taking the labels y_i we get a word $v' = y_1 \dots y_n$ representing $\overline{ux}$. Discarding the padding symbols of v' we yield v our representative of $\overline{ux}$.

Since there are finitely many states and state transitions, the time taken in each step is bounded above by some constant. Therefore the overall time is proportional to n, the minimum number of steps needed above. By **Lemma 3.1** n can only exceed $|u|$ by a bounded amount N as it would otherwise not be minimal. So for an accepted string u we can find a representative for $\overline{ui_t}$ in time $O(|u|)$ and that the representative's length is at most $|u|+N$. A change of coordinates show that the estimate holds for checking multiplication by i_t^{-1}. Thus we can find a representative of a word $\overline{w}$ of length at most $N|w|+n_0$ where n_0 is the length of a representative of the identity. Thus the time taken is $O(\sum_j^{|w|}(jN + n_0)) = O(|w|^2)$ which is what we wanted. □

In particular, knowing a word e representing the identity, we can solve the word problem in quadratic time by finding a representative in the accepted language L for the desired word and feeding it into the equality checker automata.

Let us now turn towards the lamplighter groups.

Definition 3.1 (Lamplighter Group)**.** The lamplighter groups L_m is defined as the semi-direct product $\mathbb{Z}_m \wr \mathbb{Z}$ with presentation

$$L_m = \langle a, t | a^m = e, [t^i a t^{-i}, t^j a t^{-j}] = e \rangle$$

for all $i, j \in \mathbb{Z}$.

Let us first look at an alternative way to view cone types. We start with a few definitions (Neumann and Shapiro, 1996).

Definition 3.2 (Outbound Paths)**.** Let $(\Gamma(G,I), d_I)$ be a metric space and $p(t)$ a path in $\Gamma(G,I)$. If $d(e,p(t))$ is strictly increasing for all t then $p(t)$ is an outbound path.

Definition 3.3 (Cone)**.** Let $g \in G$. The cone of g denoted $C'(g)$ is the set of outbound paths starting at g in the Cayley graph $\Gamma(G,I)$.

Definition 3.4 (Cone Type)**.** Let $g \in G$. The cone type of g denoted $C(g) = g^{-1}C'(g)$ is the set of outbound paths from g translated such that they start at the origin.

This notion of cone types is in fact equivalent to the one stated in **Definition 1.7** since we are looking at extensions to the word accepted by the acceptor automata. Let us state the following definition from (Neumann and Shapiro, 1996) before proving a proposition.

Definition 3.5 (Falsification by k-Fellow Traveller)**.** Let $\Gamma(G,I)$ be the Cayley graph $G = \langle I|R\rangle$ where I generates G as a monoid. $\Gamma(G,I)$ has the falsification by k-fellow traveller property if there is a constant k such that if $w \in I^*$ is not geodesic then there is $w' \in I^*$ such that $\overline{w} = \overline{w'}$, $|w'| < |w|$, and $d_I(\overline{\gamma_w(t)}, \overline{\gamma_{w'}(t)}) \leq k$ for all t.

So a Cayley graph has falsification for the k-fellow traveller property when for any representative of a group element $w \in I^*$ is not geodesic we can always find a shorter representative w' such that the two induced paths k-fellow travel in the Cayley graph. In other words if two words end up on the same group element, their paths from the identity diverge by a bounded amount. We can now show the following (Neumann and Shapiro, 1996).

Proposition 3.3. *If $\Gamma(G,I)$ has the falsification by k-fellow traveller property then $\Gamma(G,I)$ has finitely many cone types.*

Proof. Let k be the appropriate constant for the falsification by k-fellow traveler property and $g \in G$. We define a function f_g on $\mathcal{B}(e,k)$ by $f_g(h) = d_I(e,gh) - d_I(e,g)$. Namely $f_g(h)$ gives the relative distance of the vertex gh from e as compared to that of g. The falsification of k-fellow traveller property implies that if w is an outbound path from g then there is no path w' from gh to $g\overline{w}$ with $h \in \mathcal{B}(e,k)$. In this case we will have $|w'| + d_I(e,gh) < |w| + d_I(e,g)$ since $\overline{gh} = \overline{gww'^{-1}}$ must k-fellow travel. We can rewrite this inequality $|w'| < |w| - f_g(h)$. Note that $f_g(h)$ is fixed so the length of w' is bounded above. Thus the number of cone types is also bounded above by number of possible words of length $|w'|$ which is finite for a finite generating set I. □

We can now leverage the Myhill-Nerode theorem to give us the following.

Corollary 3.3.1 (Regular Language of Geodesics)**.** *Suppose $\Gamma(G, I)$ has the falsification by k-fellow traveller property then the set of geodesics in I^* is a regular language.*

Proof. From above we know that if $\Gamma(G, I)$ satisfies the falsification of k-fellow traveller property then the geodesics of $\Gamma(G, I)$ has finitely many cone types. But **Theorem 1.3** tells us that this means that the geodesics of $\Gamma(G, I)$ are a regular language. □

We will show a case of the following theorem (Cleary et al., 2006).

Theorem 3.4. *The lamplighter group L_m with respect to the wreath product generating set $\{a, t\}$ have no regular language of geodesics.*

This is to say that the set of geodesic paths for the lamplighter groups L_m do not form a regular language and thus we cannot use an automata to check if elements of $\{a, t\}^*$ are geodesic paths in L_m. To show this, we will require some previous results. In previous work, Cleary and Taback proved a theorem about geodesic paths in L_2 (Cleary and Taback, 2003).

Proof. Let g be the group element with $[1]_m$ at the $-n$th and nth position and $[0]_m$s everywhere else with the lamplighter at the origin. We know that this element is represented by two geodesic paths: $t^n a t^{-2n} a t^n$ and $t^{-n} a t^{2n} a t^{-n}$.

Suppose to the contrary that the geodesics of L_2 form a regular language. Thus there is a pumping length k for the corresponding automata. Choose $n > k$ and suppose that the first geodesic representative $t^n a t^{-2n} a t^n$ is part of the language. Then since $n > k$ we can write $g = xyz$ where $|xy| < k$. Namely we will have $x = t^i, y = t^j, z = t^{n-i-j} a t^{-2n} a t^n$ with $0 < j < k < n$. By the pumping lemma xy^2z must be automatic as well so $t^i t^{2j} t^{n-i-j} a t^{-2n} a t^n = t^{n+j} a t^{-2n} a t^n$ is geodesic as well. Note that this is a word of $4n + j + 2$ and corresponds to $[1]_m$ at the $(n+j)$th and $(j-n)$th positions and $[0]_m$ everywhere else. But this element can also be represented by the word $t^{-n+j} a t^{2n} a t^{-n}$ which is of length $4n - j + 2$ which is strictly shorter since $j > 0$. This is a contradiction. □

We can thus show the following corollary (Cleary et al., 2006).

Corollary 3.4.1 (Cone Types of L_m)**.** *The lamplighter groups L_m have infinitely many cone types with respect to the generating set $\{a, t\}$.*

Proof. **Theorem 3.4** tells us that geodesics in L_m are not a regular language. The contrapositive of **Corollary 3.3.1** then implies that the geodesics in I^* are not a regular language, and thus have infinitely many cone types. □

4 Conclusion

Our tour through automatic groups has drawn the connection between seemingly disparate fields such as abstract algebra and the theory of computation. Moreover, we have seen that automatic groups possess several nice properties such as a fast solution to the word problem and constructible Cayley graph. We have also explored the link between geodesic paths and regular languages, eventually showing that there are infinitely many cone types in the lamplighter groups L_m. Though all this is all but a mere glimpse at the power of geometric group theory and its widespread applications. Perhaps one ought go to Office Hours to learn more.[1]

[1] Of course, a nod to Clay and Margalit, ed. *Office Hours with a Geometric Group Theorist.*

References

Cleary S., Elder M., and Taback J. (2006). "Cone types and geodesic languages for lamplighter groups and Thompson's group F". *Journal of Algebra* 303.2, pp. 476–500. DOI: 10.1016/j.jalgebra.2005.11.016. URL: https://doi.org/10.1016%2Fj.jalgebra.2005.11.016.

Cleary S. and Taback J. (2003). "Metric properties of the lamplighter group as an automata group". DOI: 10.48550/ARXIV.MATH/0312331. URL: https://arxiv.org/abs/math/0312331.

Epstein D. B. A. (1992). *Word Processing in Groups*. CRC Press, Taylor & Francis Group.

Farb B. (1992). "Automatic Groups: A Guided Tour". *L'Enseignement Mathématique* 38.6. DOI: 10.5169/seals-59493.

Meier J. (2008). *Groups, Graphs and Trees: An Introduction to the Geometry of Infinite Groups*. London Mathematical Society Student Texts. Cambridge University Press. DOI: 10.1017/CBO9781139167505.

Neumann W. D. and Shapiro M. (1996). "A Short Course in Geometric Group Theory". URL: https://citeseerx.ist.psu.edu/viewdoc/download?doi=10.1.1.210.4879&rep=rep1&type=pdf.