Efficient Average-Case Algorithms for the Modular Group*

Jin-Yi Cai
SUNY Buffalo
cai@cs.buffalo.edu

Wolfgang H. Fuchs
Cornell University
fuchs@math.cornell.edu

Dexter Kozen
Cornell University
kozen@cs.cornell.edu

Zicheng Liu
Princeton University
zl@cs.princeton.edu

Abstract

The modular group occupies a central position in many branches of mathematical sciences. In this paper we give average polynomial-time algorithms for the unbounded and bounded membership problems for finitely generated subgroups of the modular group. The latter result affirms a conjecture of Gurevich [5].

1 Introduction

1.1 The Modular Group

The modular group Γ is a remarkable mathematical object. It has several equivalent characterizations:

(i) $SL_2(\mathbb{Z})/\pm I$, the quotient of the group $SL_2(\mathbb{Z})$ of 2×2 integer matrices with determinant 1 modulo its central subgroup $\{\pm I\}$;

(ii) the group of complex fractional linear transformations

$$z \quad \mapsto \quad \frac{az+b}{cz+d}$$

with integer coefficients satisfying $ad - bc = 1$;

(iii) the free product of cyclic groups of order 2 and 3; *i.e.*, the group presented by generators R, S and relations $R^2 \equiv S^3 \equiv 1$;

(iv) the group of automorphisms of a certain regular tesselation of the hyperbolic plane (Figure 1);

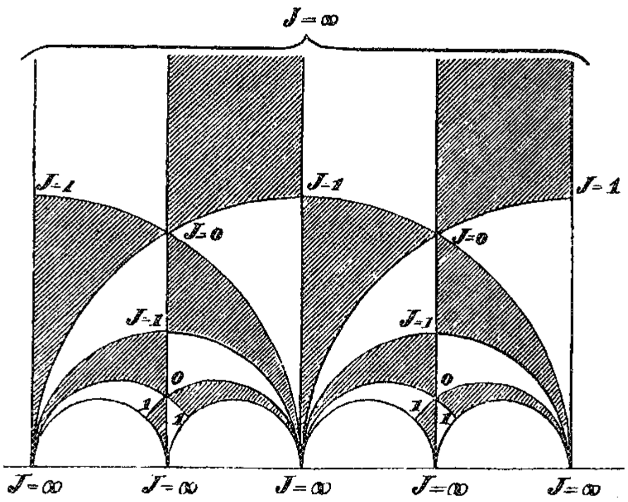

Figure 1: A tesselation of the hyperbolic plane[1]

(v) the group of sense-preserving automorphisms of the undirected cubic plane tree (Figure 2).

The modular group is intimately connected with the theory of elliptic curves, modular functions and modular forms, hyperbolic geometry, and number theory [1].

For instance, it is known that elliptic curves can be uniformly parametrized by the Weierstrass $\wp$ function. This function is invariant under the action of a group of transformations of the plane isomorphic to $\mathbb{Z} \times \mathbb{Z}$. This action gives rise to a discrete Euclidean tesselation of the plane. In contrast, a *hyperbolic uniformization* is a uniform parametrization of the elliptic curve by functions that are invariant under the

*Proc. 35th IEEE Symp. Foundations of Computer Science, Nov. 1994, to appear.

[1]Reproduced from Klein (1879) [9].

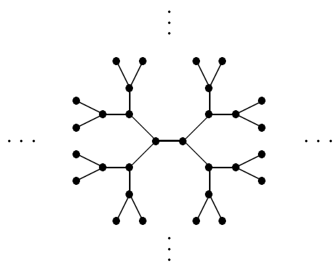

Figure 2: The undirected cubic plane tree

modular group Γ or some subgroup of it. Here the so-called *congruence subgroups* of Γ play a dominant role. The *Taniyama-Weil conjecture* states that all elliptic curves with rational coefficients admit such a uniformization by functions invariant under some congruence subgroup of Γ. It is known that a counterexample to Fermat's Last Theorem would invalidate this conjecture. While some difficulties remain, it appears that Andrew Wiles has made a significant advance towards resolving this conjecture.

The modular group is also deeply connected with many algorithmic issues. For instance, the ordinary Euclidean integer gcd algorithm can be understood in terms of a basis reduction algorithm on 2×2 integer matrices, where the reducing operations are elements of the modular group in the form (i) above. This connection allows us to apply a result of Yao and Knuth [17] concerning the integer gcd algorithm in our analysis.

Some algorithms of Schönhage [14, 15] can be best understood in light of the modular group.

A recent paper by Yap [18] is concerned with the modular group and its connection with lattice basis reduction algorithms. The basis reduction algorithms of Lenstra, Lenstra and Lovász [10] have had considerable impact on algorithm design and analysis, ranging from integer programming to polynomial factorization.

Finally, we note that the modular group has found applications in computational learning theory [3].

1.2 Subgroup Membership

In this paper we consider four natural decision problems for the modular group Γ:

The Unbounded Subgroup Membership Problem Given a finite subset $\mathcal{S} \subseteq \Gamma$ and an element $x \in \Gamma$, is x contained in the subgroup of Γ generated by $\mathcal{S}$?

The Bounded Subgroup Membership Problem Given a finite subset $\mathcal{S} \subseteq \Gamma$, an element $x \in \Gamma$, and $n \geq 0$ in unary, can x be expressed as a product of at most n elements of $\mathcal{S}$ and their inverses (repetitions allowed)?

The Unbounded Submonoid Membership Problem Given a finite subset $\mathcal{S} \subseteq \Gamma$ and an element $x \in \Gamma$, is x contained in the submonoid of Γ generated by $\mathcal{S}$?

The Bounded Submonoid Membership Problem Given a finite subset $\mathcal{S} \subseteq \Gamma$, an element $x \in \Gamma$, and $n \geq 0$ in unary, can x be expressed as a product of at most n elements of $\mathcal{S}$ (repetitions allowed)?

The only difference between the subgroup and submonoid membership problems is that in the subgroup membership problems, inverses are allowed. The subgroup membership problems reduce to the submonoid membership problems by simply including the inverses in the set $\mathcal{S}$.

We assume that these problems are presented in the form (i) of §1.1; that is, as 2×2 integer matrices with entries written in binary.

1.3 Average-Case Complexity

The study of *NP*-hard problems that are hard on average was initiated by Levin [11] and generated considerable subsequent interest [2, 6, 5, 8, 16].

Suppose the inputs to an algorithm occur randomly according to a distribution with the property that the probability that the input size is n is either zero or at least n^{-k} for some fixed k. Such a distribution is called *regular*. (For definiteness, Gurevich [5] takes the probability of the event $|x| = n$ to be proportional to $n^{-1}(\log n)^{-2}$, but any regular distribution will do.)

A deterministic algorithm runs in *polynomial time on average* if there exists an $\epsilon > 0$ such that

$$\sum_x \frac{T(x)^\epsilon}{|x|} \mathbf{Pr}(x) \quad < \quad \infty \ ,$$

where $T(x)$ is the running time of the algorithm on input x. For regular distrubutions, it suffices to show that there exists an $\epsilon > 0$ such that for all n,

$$\sum_{|x|=n} T(x)^\epsilon \cdot \mathbf{Pr}_n(x) \quad \leq \quad n^{O(1)} \ ,$$

where $\mathbf{Pr}_n(x)$ denotes the conditional probability that x occurs given that the size of the input instance is n [6, 5].

Gurevich [5] applied this notion to several algebraic problems. In particular, he showed that certain matrix decomposition problems involving the modular group are hard on average.

Gurevich defined the bounded subgroup membership problem stated in §1.2 and conjectured that it was polynomial time on average.

1.4 Main Results

In this paper we show:

Theorem 1.1 *The bounded and unbounded membership problems for finitely generated subgroups and submonoids of the modular group can be solved in polynomial time on average.*

This affirms Gurevich's conjecture.

We do not know whether the subgroup membership problems are NP-hard. However, the semigroup membership problems are quite easily shown to be NP-hard by a straightforward encoding of the subset sum problem.

1.5 Overview

Our approach is to convert x and every element in $\mathcal{S}$ to the representation (iii) of §1.1 (*i.e.*, words in $\{R, S\}^*$ reduced modulo the identities $R^2 \equiv S^3 \equiv 1$), and work in that representation.

This will be of little use if the representation (iii) is too long or if it is hard to compute from the representation (i). It turns out that it is easy to compute, but may be exponentially long in the worst case. However, it is short on average.

Our analysis makes use of an intermediate representation (2.3), which is similar to (iii), but for which a polynomial bound on the average length is known. The lengths of minimal representations in (iii) and (2.3) are mutually proportional.

Our analysis proceeds in two steps:

(i) In §4, we give deterministic polynomial-time algorithms in representation (iii) for the bounded and unbounded membership problems. These algorithms reduce the problems to a certain automata-theoretic reachability problem.

(ii) In §5 we show that the process of converting an input instance from representation (i) to representation (iii) and then executing the algorithm of §4 on the resulting data gives an average-case polynomial-time algorithm. This part of the argument relies on an estimate of Yao and Knuth [17].

The same techniques also handle other related groups such as $SL_2(\mathbb{Z})$ or the congruence subgroups of Γ. We do not treat these cases in this paper.

2 Representations of Γ

To understand this work, one must first understand the relationships among the different representations (i)–(v) of Γ described in §1.1. See [1, 13, 12, 4] for details.

In the representation (i), elements of Γ are represented as 2×2 matrices with integer entries. The group Γ is generated by the matrices

$$\begin{gathered} T = \begin{pmatrix} 1 & 1 \\ 0 & 1 \end{pmatrix} \quad R = \begin{pmatrix} 0 & -1 \\ 1 & 0 \end{pmatrix} \\ S = TR = \begin{pmatrix} 1 & -1 \\ 1 & 0 \end{pmatrix} \end{gathered} \tag{2.1}$$

Any two of these three matrices generate Γ.

These matrices correspond to the fractional linear transformations

$$T : z \mapsto z + 1 \quad R : z \mapsto -\frac{1}{z} \quad S : z \mapsto 1 - \frac{1}{z} \tag{2.2}$$

on $\mathbb{C}$, respectively. The matrices (2.1) represent the transformations (2.2) in homogeneous coordinates, viewing them as linear transformations on the projective complex line. This gives the relationship between the representations (i) and (ii).

Note that R is of order 2 and S is of order 3 (recall we are working modulo $\pm I$). In fact Γ is the free product of the cyclic groups generated by R and S. This gives the relationship with representation (iii).

To see the relationship with (iv), observe that the transformations (2.2) preserve the upper half plane $\mathbf{H}$. $\mathbf{H}$ can be regarded as a model of hyperbolic geometry, where geodesic lines are semicircles or lines perpendicular to the real axis. Under the appropriate metric, Γ is a group of isometries of $\mathbf{H}$. The region

$$\{z \in \mathbb{C} \mid -\frac{1}{2} < \Re z < \frac{1}{2},\ |z| > 1\}$$

is a fundamental region for the action of Γ, and its orbit gives a tesselation of $\mathbf{H}$. This region corresponds to the union of the two uppermost central regions, one shaded and one not, shown in Figure 1. Several works by M. C. Escher are based on this universe.

To understand the connection to (v), we observe that the infinite undirected cubic plane tree shown in Figure 2 is embedded in Figure 1 by considering the segment of the circle of radius 1 centered at 0 from $e^{2\pi i/3}$ to $e^{\pi i/3}$ as a directed edge E, then taking the orbit of this edge under the action of the group. Every element of Γ is uniquely identified with a directed edge produced in this way.

With this identification, observe that R reverses the direction of E, T corresponds to a left turn out of E, and $S = TR$ rotates about the vertex at the head of E. In any product $X_1 \cdots X_n \in \{T, R, S\}^*$ applied in order from right to left, the destination of E can be calculated by reading the string $X_1 \cdots X_n$ from left to right and interpreting T as "turn left", R as "reverse direction", and S as "rotate clockwise about the vertex before you". We can also define $U = ST$ ("turn right").

The group Γ has the following presentation in terms of T (turn left), U (turn right), and R (reverse):

$$\begin{array}{c} TRU \;=\; URT \;=\; R \\ TRT \;=\; U \qquad URU \;=\; T \\ R^2 \;=\; 1 \end{array} \tag{2.3}$$

The equations (2.3) can be applied as term rewriting rules to reduce any string in $\{R, T, U\}^*$ to normal form $(R+\epsilon)(T+U)^*(R+\epsilon)$. Every element of Γ can be expressed uniquely as a product of this form, and the length of any expression of this form is within two of minimal among all expressions in $\{R, T, U, R^{-1}, T^{-1}, U^{-1}\}^*$ denoting the same group element. This is a consequence of the fact that shortest paths in the graph of Figure 2 are unique. A similar statement holds for the presentation (iii); in this case, normal forms are strings in $\{R, S\}^*$ with no occurrence of two consecutive R's or three consecutive S's.

The presentations (2.3) and (iii) are interderivable using the facts $T = SR$, $U = SSR$, $S = TR$. Moreover, these relations show that for any group element, the lengths of the minimal representations in $\{R, S\}^*$ and $\{R, T, U\}^*$ differ by at most a factor of three.

In terms of representation (i), the left and right turns are

$$T = \begin{pmatrix} 1 & 1 \\ 0 & 1 \end{pmatrix} \quad U = \begin{pmatrix} 1 & 0 \\ 1 & 1 \end{pmatrix},$$

respectively. Note that elementary row and column operations on 2×2 matrices (adding a row or column to the other) are effected by multiplying on the left or right by T or U. In this interpretation, the significance of the normal form $(R+\epsilon)(T+U)^*(R+\epsilon)$ is that for any matrix, we can multiply by R on the left or right if necessary to make all entries nonnegative, and then there is a unique sequence of column operations to bring the matrix to I while keeping entries nonnegative. The same is true for row operations. This gives us an effective method for converting between the representations (i) and (2.3).

2.1 Integer GCD

The matrices T and U have the following significance regarding integer gcd. Let $s(m, n)$ be the number of steps in the following *subtractive Euclidean algorithm* for finding the gcd of m and n: replace the larger number by the difference of the two numbers until both are equal. Note that $s(m, n)$ is one less than the sum of all partial quotients in the continued fraction representation of m/n, $1 \le m \le n$. For example,

$$\frac{7}{16} = \frac{1}{2 + \frac{1}{3+\frac{1}{2}}}$$

and $s(7, 16) = (2 + 3 + 2) - 1 = 6$.

The matrices T and U correspond to the basic operations of the subtractive gcd algorithm in the sense that if m and n are relatively prime and appear in the top row of a matrix $A \in \Gamma$, then T^{-1} and U^{-1} applied on the right hand side effect the column operations corresponding to the steps of the subtractive gcd algorithm. It follows that the length of the unique expression in $\{T, U\}^*$ equivalent to A is exactly $s(m, n)$.

3 Length of Representations

Gurevich showed that the size of any element $A \in \Gamma$ in representation (2.3) is polynomial in the size of A in representation (i) on average [5, Lemma 4.2]. Our observation that minimal-length representations in (2.3) and (iii) are mutually proportional implies that the size of A in representation (iii) is also polynomial in the size of A in representation (i) on average. This result, together with the polynomial time algorithm of the next section, do not immediately imply an average polynomial-time complexity of the membership problems, since the number of input matrices is not fixed.

Gurevich's argument is based on the following estimate of Yao and Knuth:

Lemma 3.1 (Yao and Knuth [17])

$$\sum_{m \le n} s(m, n) = \frac{6}{\pi^2} n (\log n)^2 + O(n \log n (\log \log n)^2) .$$

It follows immediately that for fixed n, the average value of $s(m,n)$, where m is chosen uniformly at random among all positive integers less than and relatively prime to n, is at most

$$O(\frac{n(\log n)^2}{\varphi(n)}) \quad \leq \quad O((\log n)^2 \log\log n) \ , \quad (3.4)$$

where $\varphi(n)$ is the Euler totient function. The inequality (3.4) follows from the estimate $\varphi(n) = \Omega(n/\log\log n)$ [7, Theorem 328].

Except for I, T, and U, if $A \in \Gamma$ has nonnegative entries and maximum entry n, and if m is the other entry in the same row as n, then $1 \leq m < n$, $(m,n) = 1$, and the rest of A is uniquely determined by the constraint on the determinant of A. Since there are four ways to choose the position of the maximal entry n in A, such matrices are in four-to-one correspondence with the pairs m, n such that $1 \leq m < n$ and $(m,n) = 1$. It follows that the length of the unique expression in $\{T,U\}^*$ corresponding to $A \in \Gamma$ is also polynomial on average.

4 Deterministic Algorithms

In this section we give deterministic polynomial-time algorithms for the unbounded and bounded membership problems when the input is given in representation (iii) of §1.1, *i.e.* in terms of generators R, S and relations $R^2 \equiv S^3 \equiv 1$.

Consider the term rewriting system over strings in $\{R,S\}^*$ consisting of reduction rules $R^2 \to \epsilon$, $S^3 \to \epsilon$. We write $x \to y$ if the string x reduces to the string y in zero or more steps. A string is said to be *reduced* or in *normal form* if no reduction rule applies. This system has nonoverlapping redexes (the *redexes* are R^2 and S^3), thus it follows from term rewriting theory that normal forms are unique, and $x \equiv y$ iff x and y have a common normal form.

Suppose now we are given a set $\mathcal{S}$ of reduced strings in $\{R,S\}^*$, a reduced string $x \in \{R,S\}^*$, and (for the bounded membership problem) an integer n in unary. Let $\mathcal{S}^*$ denote the submonoid of $\{R,S\}^*$ generated by $\mathcal{S}$. The *unbounded membership problem* is to determine whether there exists a string $y \in \mathcal{S}^*$ such that $y \to x$. For the *bounded membership problem*, we require in addition that $y \in \mathcal{S}^m$ for some $m \leq n$. We will give an algorithm that runs in time polynomial in n and the sum of the lengths of x and the elements of $\mathcal{S}$.

Note that this formulation of the problem asks for membership of x in a finitely generated submonoid of Γ. If we wish to determine membership in a finitely generated subgroup, we can simply include the inverses of elements of $\mathcal{S}$.

In a fixed reduction sequence $x \to y$, we say that an occurrence of a letter a in y *comes from* an occurrence of a in x if $x = uav$ and $y = zaw$, where the mentioned occurrences of a in x and y are as shown, and the appropriately chosen subsequences of the reduction sequence give $u \to z$ and $v \to w$. For a fixed reduction sequence $x \to y$, every letter of y comes from a unique letter of x. The remaining letters of x must eventually become part of a redex and disappear.

For any set H of strings, we denote by $H/\!\equiv$ the set of strings $\equiv$-equivalent to some string in H. Thus $\mathcal{S}^*/\!\equiv$ denotes the set of strings representing elements of the submonoid of Γ generated by $\mathcal{S}$. This notation is slightly nonstandard but convenient for our purposes. Our task is to find an efficient membership test for $\mathcal{S}^*/\!\equiv$ for the unbounded membership problem and $\bigcup_{m \leq n} \mathcal{S}^m/\!\equiv$ for the bounded membership problem.

4.1 An Automata-Theoretic Characterization

Let M be the finite automaton with states

$$Q \quad = \quad \{u \mid u \text{ is a suffix of some } x \in \mathcal{S}\} \ ,$$

start and final state ϵ (the null string), and transitions

$$\begin{array}{ll} au \xrightarrow{a} u \ , & a \in \{R,S\} \ , \\ u \xrightarrow{\epsilon} v \ , & u^{-1}v \in \mathcal{S}^*/\!\equiv \ . \end{array}$$

We will show below that for any reduced x, $x \in \mathcal{S}^*/\!\equiv$ iff x is accepted by M. Note that M has linearly many states and the ϵ edges are transitive. Once we construct the automaton for a given set of generators $\mathcal{S}$, we can test membership in $\mathcal{S}^*/\!\equiv$ of any string efficiently by reducing to normal form and then testing whether the resulting string is accepted by M. This will give us an efficient algorithm for the unbounded membership problem.

For the bounded membership problem, we will need a slightly stronger formulation. Define

$$A(x) \quad = \quad \{n \mid x \in \mathcal{S}^n/\!\equiv\}$$

for any string x. Note $x \in \mathcal{S}^*/\!\equiv$ iff $A(x) \neq \emptyset$. Label each ϵ-transition $u \xrightarrow{\epsilon} v$ in M with the nonempty set $A(u^{-1}v)$. Let $+$ denote setwise addition:

$$X + Y \quad = \quad \{m + n \mid m \in X, \ n \in Y\} \ .$$

For any computation path $\sigma : u \xrightarrow{x} v$ in the automaton M, let $A(\sigma)$ denote the sum of the sets labeling the ϵ-transitions along the path σ. More formally,

$$\begin{aligned} A(\sigma) &= \{0\}\ , \quad \text{if } \sigma \text{ is of length } 0 \\ A(\sigma \cdot (au \xrightarrow{a} u)) &= A(\sigma) \\ A(\sigma \cdot (u \xrightarrow{\epsilon} v)) &= A(\sigma) + A(u^{-1}v)\ . \end{aligned}$$

Theorem 4.1 *For any reduced x and $n \geq 0$, $x \in \mathcal{S}^n/\equiv$ if and only if there is an accepting computation path $\sigma : \epsilon \xrightarrow{x} \epsilon$ with $n \in A(\sigma)$. In other words, for any reduced x,*

$$A(x) \quad = \bigcup_{\sigma:\epsilon \xrightarrow{x} \epsilon} A(\sigma)\ .$$

Proof. ($\longleftarrow$) We show by induction on the length of σ that if $\sigma : \epsilon \xrightarrow{x} u$ and $n \in A(\sigma)$ then $xu \in \mathcal{S}^n/\equiv$. The result follows by taking $u = \epsilon$. If σ is of length zero, then $A(\sigma) = \{0\}$ and $x = \epsilon \in \mathcal{S}^0$. If $\sigma = \tau \cdot (au \xrightarrow{a} u)$, then $x = ya$, $\tau : \epsilon \xrightarrow{y} au$, and $n \in A(\tau) = A(\sigma)$. By the induction hypothesis, $xu = (ya)u = y(au) \in \mathcal{S}^n/\equiv$. Finally, if $\sigma = \tau \cdot (v \xrightarrow{\epsilon} u)$ where $\tau : \epsilon \xrightarrow{x} v$, then $n = k + m$ for some $k \in A(\tau)$ and $m \in A(v^{-1}u)$. Then $v^{-1}u \in \mathcal{S}^m/\equiv$, and by the induction hypothesis, $xv \in \mathcal{S}^k/\equiv$. Thus $xu \equiv xvv^{-1}u \in \mathcal{S}^n/\equiv$.

($\longrightarrow$) If $x \in \mathcal{S}^0/\equiv$, then $x = \epsilon$ since x is reduced. In this case take σ to be the null path $\epsilon \xrightarrow{x} \epsilon$ and we are done. Otherwise, we show by induction that if $r \in \mathcal{S}^{n-1}$, $st \in \mathcal{S}$, and $rs \to y$ where y is reduced, then there is a computation path $\sigma : \epsilon \xrightarrow{y} t$ with $n \in A(\sigma)$. The result then follows by taking $t = \epsilon$.

For $n = 1$, we have $r = \epsilon$. Then $y = s$ since s is reduced, and there is a computation path $\tau : \epsilon \xrightarrow{\epsilon} st$ of length one with $1 \in A(\tau) = A(st)$. Combining this with $|s|$ transitions of the form $au \xrightarrow{a} u$, we obtain a computation path $\sigma : \epsilon \xrightarrow{s} t$ with $1 \in A(\sigma)$.

Now suppose $n \geq 2$. If $s = \epsilon$, we have $y \equiv r \in \mathcal{S}^{n-2}\mathcal{S}$ and $t \in \mathcal{S}$. Then $1 \in A(t)$ and by the induction hypothesis, we have a computation path $\tau : \epsilon \xrightarrow{y} \epsilon$ with $n-1 \in A(\tau)$. Combining this with the transition $\epsilon \xrightarrow{\epsilon} t$, we obtain a path $\sigma : \epsilon \xrightarrow{y} t$ with $n \in A(\sigma)$.

If $s \neq \epsilon$ and the last symbol of y comes from the last symbol of s in the reduction $rs \to y$, then $s = ua$, $y = va$, and $ru \to v$ for some u, v. By the induction hypothesis, we have a computation path $\tau : \epsilon \xrightarrow{v} at$ with $n \in A(\tau)$. Combining this with the transition $at \xrightarrow{a} t$, we obtain a computation path $\sigma : \epsilon \xrightarrow{y} t$ with $n \in A(\sigma)$.

Finally, if the last symbol of y does not come from the last symbol of s, then the last symbol of y cannot come from any symbol of s, since s is reduced. Thus we can write $r = upqv$ where $u \in \mathcal{S}^k$, $v \in \mathcal{S}^m$, $pq \in \mathcal{S}$, the last symbol of y comes from the last symbol of p, and $qvs \equiv \epsilon$. Then $up \equiv upqvs = rs \equiv y$. Since y is reduced, $up \to y$. By the induction hypothesis, we have a computation path $\tau : \epsilon \xrightarrow{y} q$ with $k+1 \in A(\tau)$. Moreover, since $qvs \equiv \epsilon$, we have $q^{-1}t \equiv vst \in \mathcal{S}^{m+1}$, thus $m+1 \in A(q^{-1}t)$. Combining τ with the transition $q \xrightarrow{\epsilon} t$, we obtain a computation path $\sigma : \epsilon \xrightarrow{y} t$ with $n = k + m + 2 \in A(\sigma)$. $\square$

Corollary 4.2 *For any reduced x, $x \in \mathcal{S}^*/\equiv$ if and only if M accepts x.*

4.2 Construction of M

We have reduced the problem of determining membership in $\mathcal{S}^*/\equiv$ of arbitrary strings x to the problem of determining membership in $\mathcal{S}^*/\equiv$ of $u^{-1}v$ for $u, v \in Q$. We now give an efficient algorithm for this problem.

Let N be the set of normal forms of strings $u^{-1}v$ for $u, v \in Q$. Note $\mathcal{S} \subseteq N$ and N is finite. Let $B(x)$, $x \in N$, be the smallest family of sets closed under the following rules:

(i) $0 \in B(\epsilon)$

(ii) $1 \in B(x)$, $x \in \mathcal{S}$

(iii) $B(x) + B(y) \subseteq B(z)$, where $z \equiv xy$.

If x is not reduced but $x \to y \in N$, we define $B(x) = B(y)$.

We show below that $A(x) = B(x)$ for $x \in N$. This gives a simple inductive method for determining the ϵ-transitions of M: mark ϵ and all $x \in \mathcal{S}$ as required by rules (i) and (ii), then mark $z \in N$ whenever $x, y \in N$ are marked and $xy \to z$. Then $u \xrightarrow{\epsilon} v$ iff the normal form of $u^{-1}v$ is marked.

Lemma 4.3 *If $u \in Q$, $pq \in \mathcal{S}$, $r \in \mathcal{S}^n$, and $urp \equiv \epsilon$, then $n + 1 \in B(u^{-1}q)$.*

Proof. If $n = 0$, then $u^{-1}q \equiv pq$, and the conclusion follows from rule (ii).

If $n \geq 1$ and $u = \epsilon$, then we can write $r = vs$ with $v \in \mathcal{S}$, $s \in \mathcal{S}^{n-1}$, and $vsp \to \epsilon$. Then $1 \in B(u^{-1}v)$, and by the induction hypothesis, $n \in B(v^{-1}q)$, therefore $n + 1 \in B(u^{-1}q)$ by rule (iii).

Similarly, if $p = \epsilon$, then we can write $r = sv$ with $s \in \mathcal{S}^{n-1}$, $v \in \mathcal{S}$, and $usv \to \epsilon$. Then $1 \in B(\epsilon^{-1}v)$, and by the induction hypothesis, $n \in B(u^{-1}\epsilon)$, therefore $n + 1 \in B(u^{-1}q)$ by rule (iii).

Assume now that $n \geq 1$ and both u and p are nonnull. The proof proceeds by induction on the length of the reduction sequence $urp \to \epsilon$.

If urp can be expressed as the concatenation of two nonnull strings, each of which reduces to ϵ, then the first of these cannot be a substring of u and the second cannot be a substring of p, since u and p are reduced. Thus we can write $r = stxy$ where $tx \in \mathcal{S}$, $s \in \mathcal{S}^k$, $y \in \mathcal{S}^m$, $m + k + 1 = n$, $ust \equiv xyq \equiv \epsilon$. By the induction hypothesis, $k + 1 \in B(u^{-1}x)$ and $m + 1 \in B(x^{-1}q)$. By rule (iii), $n + 1 = m + k + 2 \in B(u^{-1}q)$.

If urp has no such decomposition, then in the reduction $urp \to \epsilon$, if the last reduction rule applied is $RR \to \epsilon$, the first R must come from the leftmost symbol of u and the second must come from the rightmost symbol of p, otherwise we would have a decomposition as in the previous case. Thus $u = Rx$, $p = yR$, and $xry \to \epsilon$. By the induction hypothesis, we have $n + 1 \in B(x^{-1}Rq) = B(u^{-1}q)$.

If the last reduction rule applied is $SSS \to \epsilon$, then again the first S must come from the leftmost symbol of u and the third must come from the rightmost symbol of p.

If the second S comes from u, then we have $u = SSx$ and $p = yS$, where $xry \to \epsilon$. By the induction hypothesis we have $n + 1 \in B(x^{-1}Sq) = B(u^{-1}q)$.

If the second S comes from p, then we have $u = Sx$ and $p = ySS$, where $xry \to \epsilon$. By the induction hypothesis we have $n + 1 \in B(x^{-1}SSq) = B(u^{-1}q)$.

Finally, if the second S comes from r, then we have $u = Sx$, $r = yzSws$, and $p = tS$, where $zSw \in \mathcal{S}$, $y \in \mathcal{S}^k$, $s \in \mathcal{S}^m$, and $xyz \equiv wst \equiv \epsilon$. By the induction hypothesis we have $k + 1 \in B(x^{-1}Sw)$ and $m + 1 \in B(w^{-1}Sq)$, therefore by rule (iii) we have $n + 1 = m + k + 2 \in B(x^{-1}SSq) = B(u^{-1}q)$. □

Theorem 4.4 *$A(x) = B(x)$ for $x \in N$.*

Proof. We argue first that the sets $A(x)$ satisfy all the rules (i)–(iii) for $x \in N$, thus $B(x) \subseteq A(x)$. The rule (i) just says $\epsilon \in \mathcal{S}^0$, (ii) just says that $x \in \mathcal{S}^1$ for $x \in \mathcal{S}$, and (iii) says that if x in $\mathcal{S}^m$ and $y \in \mathcal{S}^n$, then $xy \in \mathcal{S}^{m+n}$.

For the reverse inclusion, we show by induction on n that for all $u, v \in Q$, if $n \in A(u^{-1}v)$ then $n \in B(u^{-1}v)$. If $n = 0$, then $u^{-1}v \equiv \epsilon$, and $0 \in B(u^{-1}v)$ by rule (i). If $n = 1$, then $u^{-1}v \equiv x \in \mathcal{S}$, and $1 \in B(u^{-1}v)$ by rule (ii).

Assume now that $n \geq 2$. Let $u^{-1}v \equiv r \in \mathcal{S}^n$. Then $ur \equiv v$, and since v is reduced, we have $ur \to v$.

We proceed by induction on the length of v. If $v = \epsilon$, then writing $r = st$ with $s \in \mathcal{S}^{n-1}$ and $t \in \mathcal{S}$, we have $ust \to \epsilon$, so $n \in B(u^{-1}v)$ by Lemma 4.3.

Suppose now that v is nonnull. If the first letter of v comes from u in the reduction $ur \to v$, then it must come from the first letter of u, since u is reduced. Thus $u = ay$, $v = aw$, and $yr \to w$. By the induction hypothesis, $n \in B(y^{-1}w) = B(u^{-1}v)$.

If the first letter of v comes from r, then we can write $r = styz$ where $s \in \mathcal{S}^k$, $z \in \mathcal{S}^m$, $ty \in \mathcal{S}$, and the first letter of v comes from the first letter of y. Then $ust \to \epsilon$ and $yz \to v$. By Lemma 4.3, $k+1 \in B(u^{-1}y)$, and by the induction hypothesis, $m \in B(y^{-1}v)$. By rule (iii), $n = m + k + 1 \in B(u^{-1}v)$. □

4.3 Unbounded Membership

Once we have constructed the automaton M for a given set of generators $\mathcal{S}$, we can solve the unbounded membership problem for a given string efficiently by reducing to normal form and then testing whether the resulting string is accepted by M. Corollary 4.2 asserts the correctness of this procedure.

4.4 Bounded Membership

One approach to solving the bounded membership problem is to observe that the closure rules (i)–(iii) are essentially equivalent to the following context-free grammar over a single-letter alphabet $\{a\}$ and nonterminals A_x, $x \in N$:

$$\begin{array}{lcll} A_\epsilon & \to & \epsilon & \\ A_x & \to & a\ , & x \in \mathcal{S} \\ A_z & \to & A_x A_y\ , & xy \equiv z\ . \end{array}$$

Then for $x \in N$, $A(x)$ is the set of lengths of strings in $\{a\}^*$ generated from the nonterminal A_x. By Parikh's Theorem, this is a regular set, and we can determine membership in $A(x)$ efficiently using known algorithms for context-free language recognition.

However, for the purpose of deciding whether there exists an accepting computation path $\sigma : \epsilon \xrightarrow{x} \epsilon$ with $m \in A(\sigma)$ and $m \leq n$, we do not need to know the entire set $A(u^{-1}v)$ but only its smallest element. Indeed, if $A(u^{-1}v)$ is nonempty but its smallest element is greater than n, then we might as well delete the edge $u \xrightarrow{\epsilon} v$, since it cannot contribute to such an accepting computation path.

Let r be the number of relations $x \equiv yz$ that hold among elements of N. Here is an $O(nr)$ algorithm for determining all the minimum elements of $A(x)$ for $x \in N$. For each $x \in N$ we have an integer variable m_x that holds a current estimate of $\min A(x)$. We initialize m_x to $n+1$, which we regard as ∞. We assume that for each $x \in N$ we have a list L_x of all relations

$z \equiv xy$ or $z \equiv yx$ that hold among the elements of N with x on the right hand side. The combined length of all the lists L_x is at most $2r$.

Now $\min A(\epsilon) = 0$ and $\min A(x) = 1$ for $x \in \mathcal{S}$, so we set $m_\epsilon := 0$ and $m_x := 1$ for $x \in \mathcal{S}$ and put ϵ and all $x \in \mathcal{S}$ in a bag for further processing. We then repeat the following procedure until the bag becomes empty. Take the next x out of the bag and scan through the list L_x. For each relation $z \equiv xy$ or $z \equiv yx$ on the list, check whether $m_z > m_x + m_y$. If so, set $m_z := m_x + m_y$ and put z in the bag.

Each x taken out of the bag takes $O(|L_x|)$ time to process, and a particular x can enter the bag at most n times, since m_x is decremented each time. This gives $O(nr)$ in all.

Once we have computed the minimum element of $A(u^{-1}v)$ for each pair $u, v \in Q$, we can weight the ϵ-transition $u \xrightarrow{\epsilon} v$ with this quantity and weight the other transitions $au \xrightarrow{a} u$ zero. Then to compute the minimum element of $A(x)$ for a given reduced x, we can use a variant of Dijkstra's shortest path algorithm to find a minimum-weight computation path $\epsilon \xrightarrow{x} \epsilon$ and check that its weight is at most n. The correctness of this method is given by Theorem 4.1. This solves the bounded membership problem.

5 Average Case Algorithms

In this section we prove Theorem 1.1, which states that the bounded and unbounded subgroup and submonoid membership problems are polynomial-time on average.

For a positive integer m, we take the *size* of m to be $\log m$, the base 2 logarithm of m. For a sequence $\overline{m}$ of positive integers, we take the *size* of $\overline{m}$, denoted $\|\overline{m}\|$, to be the sum of the sizes of its components.

An instance of the unbounded subgroup or submonoid membership problem of §1.2 is a sequence $\mathcal{S}$ of 2×2 integer matrices with determinant one and entries written in binary. An instance of the bounded subgroup or submonoid membership problem is a pair $(\mathcal{S}, n)$ where $\mathcal{S}$ is as above and n is a positive integer. For our analysis, we will measure the size of such instances as follows. For a matrix with entries a, b, c, d, we take $\mu(A) = \max\{|a|, |b|, |c|, |d|\}$, where $|a|$ denotes the absolute value of a. Let $\mu(\mathcal{S})$ be the sequence $(\mu(A) \mid A \in \mathcal{S})$. We define the *size* of an instance $\mathcal{S}$ of the unbounded membership problem to be $\|\mathcal{S}\| = \|\mu(\mathcal{S})\|$, and the size of an instance $(\mathcal{S}, n)$ of the bounded membership problem to be $\|(\mathcal{S}, n)\| = \|\mathcal{S}\| + n$.

Let $\sigma(\mathcal{S})$ denote the sum of the lengths of the R, S representations of the matrices in $\mathcal{S}$, as described in §2.

Lemma 5.1 *Let $\overline{m} = (m_1, \ldots, m_k)$. For $d \geq 1$, the quantity $\sum_{i=1}^k (\log m_i)^d$ is maximized subject to the constraints $1 \leq m_i$, $1 \leq i \leq k$, and $\prod_{i=1}^k m_i = n$ at the extremes $m_i = n$ and $m_j = 1$, $j \neq i$.*

Proof. Taking $a_i = \log m_i / \log n$, the problem is equivalent to maximizing $\sum_{i=1}^k a_i^d$ subject to the constraints $0 \leq a_i$, $1 \leq i \leq k$, and $\sum_{i=1}^k a_i = 1$. This occurs at the extremes, since the function is convex and symmetric. □

Proof of Theorem 1.1. We treat the unbounded membership problems first. As remarked in §1.3, we need only show that there exists an $\epsilon > 0$ such that

$$\sum_{\|\mathcal{S}\|=n} T(\mathcal{S})^\epsilon \cdot \mathbf{Pr}_n(\mathcal{S}) \quad = \quad n^{O(1)} , \qquad (5.5)$$

where $T(\mathcal{S})$ is the running time of the algorithm on input $\mathcal{S}$ and $\mathbf{Pr}_n(\mathcal{S})$ denotes the conditional probability that $\mathcal{S}$ occurs given that the size of the input instance is n.

By results of §4, we have $T(\mathcal{S}) = \sigma(\mathcal{S})^c$ for some constant c. Since all instances of size n are equally likely, $Prn\mathcal{S} = |\{\mathcal{S} \mid \|\mathcal{S}\| = n\}|^{-1}$ for $\mathcal{S}$ of size n, where $|X|$ denotes the cardinality of the set X. Taking $\epsilon = 1/c$, (5.5) becomes

$$\frac{\sum_{\|\mathcal{S}\|=n} \sigma(\mathcal{S})}{|\{\mathcal{S} \mid \|\mathcal{S}\| = n\}|} \quad = \quad n^{O(1)} . \qquad (5.6)$$

We now establish (5.6). For $\overline{\ell} = (\ell_1, \ldots, \ell_k)$ and $\overline{m} = (m_1, \ldots, m_k)$, $\overline{\ell} \leq \overline{m}$ means that $\ell_i \leq m_i$, $1 \leq i \leq k$, and $(\overline{\ell}, \overline{m}) = 1$ means that ℓ_i and m_i are relatively prime, $1 \leq i \leq k$. The numerator of (5.6) is

$$\sum_{\|\mathcal{S}\|=n} \sigma(\mathcal{S}) \quad = \quad \sum_{k=1}^{n} \sum_{\substack{\overline{m} \in \mathbb{N}^k \\ \|\overline{m}\| = n}} \sum_{\mu(\mathcal{S})=\overline{m}} \sigma(\mathcal{S}) \quad (5.7)$$

and for $\overline{m} \in \mathbb{N}^k$,

$$\begin{aligned} & \sum_{\mu(\mathcal{S})=\overline{m}} \sigma(\mathcal{S}) \\ \leq \quad & 12k \sum_{\substack{\overline{\ell} \leq \overline{m} \\ (\overline{\ell}, \overline{m}) = 1}} \sum_{i=1}^{k} s(\ell_i, m_i) \\ = \quad & O(n \sum_{\substack{\overline{\ell} \leq \overline{m} \\ (\overline{\ell}, \overline{m}) = 1}} \sum_{i=1}^{k} s(\ell_i, m_i)) . \qquad (5.8) \end{aligned}$$

The coefficient $12k$ reflects the number of ways of choosing the positions of the largest elements of the matrices in $\mathcal{S}$ and the factor bounding the lengths of the R, S and R, T, U representations as discussed in §2. The vectors $\overline{\ell}$ represent the possible entries in the same row as the largest entry of each matrix in $\mathcal{S}$. As discussed in §2.1, once that row is given, the rest of the matrix is uniquely determined, and the length of the R, T, U representation of the i^{th} matrix in $\mathcal{S}$ is $s(\ell_i, m_i)$.

Changing the order of summation in (5.8), we have

$$\begin{aligned}
&\sum_{i=1}^{k} \sum_{\substack{\overline{\ell} \leq \overline{m} \\ (\overline{\ell}, \overline{m}) = 1}} s(\ell_i, m_i) \\
&= \sum_{i=1}^{k} \Big(\prod_{\substack{j=1 \\ j \neq i}}^{k} \varphi(m_j) \Big) \Big(\sum_{\substack{\ell_i \leq m_i \\ (\ell_i, m_i) = 1}} s(\ell_i, m_i) \Big) \\
&= O\Big(\sum_{i=1}^{k} m_i (\log m_i)^2 \prod_{\substack{j=1 \\ j \neq i}}^{k} \varphi(m_j) \Big) \qquad (5.9) \\
&= O\Big(\Big(\prod_{j=1}^{k} \varphi(m_j) \Big) \sum_{i=1}^{k} \frac{m_i}{\varphi(m_i)} (\log m_i)^2 \Big) \\
&= O\Big(\Big(\prod_{j=1}^{k} \varphi(m_j) \Big) \sum_{i=1}^{k} (\log m_i)^3 \Big) . \qquad (5.10)
\end{aligned}$$

Step (5.9) uses Lemma 3.1 and step (5.10) uses the estimate $\varphi(m) \geq \Omega(m/\log\log m)$ [7, Theorem 328]. Thus (5.7) is bounded by

$$\begin{aligned}
&O\Big(n \sum_{k=1}^{n} \sum_{\substack{\overline{m} \in \mathbb{N}^k \\ \|\overline{m}\| = n}} \Big(\prod_{j=1}^{k} \varphi(m_j) \Big) \sum_{i=1}^{k} (\log m_i)^3 \Big) \\
&\leq O\Big(n^4 \sum_{k=1}^{n} \sum_{\substack{\overline{m} \in \mathbb{N}^k \\ \|\overline{m}\| = n}} \prod_{j=1}^{k} \varphi(m_j) \Big) . \qquad (5.11)
\end{aligned}$$

The inequality (5.11) follows from Lemma 5.1.

The denominator of (5.6) is

$$\begin{aligned}
&|\{\mathcal{S} \mid \|\mathcal{S}\| = n\}| \\
&= \sum_{k=1}^{n} \sum_{\substack{\overline{m} \in \mathbb{N}^k \\ \|\overline{m}\| = n}} \sum_{\mu(\mathcal{S}) = \overline{m}} 1 \\
&= \sum_{k=1}^{n} \sum_{\substack{\overline{m} \in \mathbb{N}^k \\ \|\overline{m}\| = n}} \sum_{\substack{\overline{\ell} \leq \overline{m} \\ (\overline{\ell}, \overline{m}) = 1}} 4k \\
&\geq \Omega\Big(\sum_{k=1}^{n} \sum_{\substack{\overline{m} \in \mathbb{N}^k \\ \|\overline{m}\| = n}} \prod_{j=1}^{k} \varphi(m_j) \Big) . \qquad (5.12)
\end{aligned}$$

Dividing the upper bound (5.11) for the numerator of (5.6) by the lower bound (5.12) for the denominator of (5.6), we obtain the polynomial bound $O(n^4)$ for the quotient.

Thus the condition (5.5) is fulfilled, and the algorithm is polynomial time on average.

For the bounded membership problems, as above we need to show for each n that

$$\frac{\sum_{\|\mathcal{S}\| + m = n} \sigma(\mathcal{S}) + m}{|\{(\mathcal{S}, m) \mid \|\mathcal{S}\| + m = n\}|} = n^{O(1)} .$$

But the left hand side is bounded by

$$\begin{aligned}
&\frac{\sum_{\|\mathcal{S}\| \leq n} \sigma(\mathcal{S}) + \sum_{\|\mathcal{S}\| \leq n} n}{|\{\mathcal{S} \mid \|\mathcal{S}\| \leq n\}|} \\
&\leq \sum_{m=1}^{n} \frac{\sum_{\|\mathcal{S}\| = m} \sigma(\mathcal{S})}{|\{\mathcal{S} \mid \|\mathcal{S}\| = m\}|} + n ,
\end{aligned}$$

which by (5.6) is polynomial in n. □

Acknowledgements

We thank Michael Ben-Or, Marshall Cohen, Joachim von zur Gathen, David Henderson, Russell Impagliazzo, Donald Knuth, Richard Lipton, Gabriele Meyer, Paul Pedersen, Adi Shamir, Avi Wigderson, Andrew Yao, and Richard Zippel for their help. We gratefully acknowledge the support of the National Science Foundation under grants CCR-9057486 and CCR-9317320, BRICS (Basic Research in Computer Science), a Centre of the Danish National Research Foundation, the U.S. Army Research Office through ACSyAM, a branch of the Mathematical Sciences Institute of Cornell University under contract DAAL03-91-C-0027, and the Alfred P. Sloan Foundation.

References

[1] T. M. Apostol, *Modular Functions and Dirichlet Series in Number Theory*, Springer-Verlag, 1976.

[2] S. Ben-David, B. Chor, O. Goldreich, and M. Luby, *On the theory of average case complexity*, in 21st Symp. Theory of Computing, ACM, 1989, pp. 204–216.

[3] N. Bshouty, T. Hancock, and L. Hellerstein, *Learning arithmetic read-once formulas*, in 24th Symp. Theory of Computing, 1992, pp. 370–381.

[4] H. S. M. Coxeter and W. O. J. Moser, *Generators and Relations for Discrete Groups*, Springer-Verlag, 4th ed., 1984.

[5] Y. Gurevich, *Matrix decomposition problem is complete for the average case*, in 31st Symp. Foundations of Computer Science, IEEE, 1990, pp. 802–811. SIAM J. Comput., to appear.

[6] ———, *Average case complexity*, J. Comput. Syst. Sci., 42 (1991), pp. 346–398.

[7] G. H. Hardy and E. M. Wright, *An Introduction to the Theory of Numbers*, Oxford University, 5th ed., 1979.

[8] R. Impagliazzo and L. Levin, *No better ways to generate hard NP instances than picking uniformly at random*, in 31st Symp. Foundations of Comput. Sci., IEEE, 1990, pp. 812–821.

[9] F. Klein, *Über die Transformation der elliptischen Functionen und die Auflösung der Gleichungen fünften Grades*, Math. Ann., 14 (1879), pp. 111–172.

[10] A. K. Lenstra, H. W. Lenstra, and L. Lovász, *Factoring polynomials with rational coefficients*, Math. Ann., 261 (1982), pp. 515–534.

[11] L. Levin, *Average case complete problems*, SIAM J. Comput., 15 (1986), pp. 285–286.

[12] W. Magnus, A. Karrass, and D. Solitar, *Combinatorial Group Theory*, Interscience, 1966.

[13] M. Newman, *Integral Matrices*, Academic Press, 1972.

[14] A. Schönhage, *Schnelle berchnung von kettenbruchentwicklungen*, Acta Informatica, 1 (1971), pp. 139–144.

[15] ———, *Fast reduction and composition of binary quadratic forms.* Preprint, 1992.

[16] R. Venkatesan and L. Levin, *Random instances of a graph coloring problem are hard*, in 20th Symp. Theory of Computing, ACM, 1988, pp. 217–222.

[17] A. C. Yao and D. E. Knuth, *Analysis of the subtractive algorithm for greatest common divisors*, Proc. Nat. Acad. Sci. USA, 72 (1975), pp. 4720–4722.

[18] C. Yap, *Fast unimodular reductions: planar integer lattices*, in 33rd Symp. Foundations of Comput. Sci, IEEE, 1992, pp. 437–446.