Matrix Transformation is Complete for the Average Case*

Andreas Blass† and Yuri Gurevich‡

Abstract

In the theory of worst case complexity, NP completeness is used to establish that, for all practical purposes, the given NP problem is not decidable in polynomial time. In the theory of average case complexity, average case completeness is supposed to play the role of NP completeness. However, the average case reduction theory is still at an early stage, and only a few average case complete problems are known. We present the first algebraic problem complete for the average case under a natural probability distribution. The problem is this: Given a unimodular matrix X of integers, a set S of linear transformations of such unimodular matrices and a natural number n, decide if there is a product of $\leq n$ (not necessarily different) members of S that takes X to the identity matrix.

Contents

*SIAM J. on Computing

†Partially supported by NSF grants DMR 88-01988 and DMS-9204276. Address: Mathematics Department, University of Michigan, Ann Arbor, MI 48109-1003, USA; ablass@umich.edu

‡Partially supported by NSF grants CCR 89-04728, CCR 92-04742 and ONR grant N00014-91-J-11861. Address: Electrical Engineering and Computer Science Department, University of Michigan, Ann Arbor, MI 48109-2122, USA; gurevich@umich.edu

1 Introduction

The theory of NP completeness is very useful. It allows one to establish that certain NP problems are NP complete and therefore, for all practical purposes, not decidable in polynomial time (PTime). One way around the NP completeness phenomenon is to consider the given NP problem together with an appropriate probability distribution and seek a decision algorithm that runs quickly on average. This works very well for some problems (see [GS] for example), but some other randomized decision problems appear too difficult even on average. It would be very useful to generalize the theory of NP completeness to be able to establish that certain randomized decision problems are, for all practical purposes, not decidable quickly on average. This is the motivation for the theory of average case completeness.

Before we plunge into this theory, let us review briefly the NP completeness theory. The idea is that PTime algorithms are considered easy. In particular, PTime decidable NP problems are considered easy. One says that an NP problem Π_1 reduces to an NP problem Π_2 if there is a PTime algorithm R from instances of Π_1 to instances of Π_2 that takes positive instances to positive instances and negative instances to negative instances. Such an R is a many-one PTime reduction from Π_1 to Π_2. A PTime decision algorithm A for Π_2 gives rise to the PTime decision algorithm $R \circ A$ for Π_1. A decision problem is hard for NP (via many-one PTime reductions) if every NP problem reduces to it (by means of a many-one PTime reduction). A decision problem is complete for NP if it belongs to NP and is hard for NP. Most known natural NP problems are either PTime decidable or NP complete.

The theory of average case completeness was pioneered by Leonid Levin in [Le]. Levin replaced NP with the class RNP of NP problems with so-called PTime computable probability distributions [Le, Gu1]. He generalized PTime computability to computability in time polynomial on average (APtime computability) and defined many-one PTime reductions of RNP problems. Then he established that a bounded version of the known tiling problem together with a natural probability distribution is complete for RNP via many-one PTime reductions. (That is, the randomized tiling problem belongs to RNP and every problem in RNP reduces to it via many-one PTime

reductions.) Another RNP problem, implicitly present in [Le], is Bounded Halting, a bounded version of the standard halting problem together with natural probability distribution; this problem is explicitly defined and proved complete in [Gu1].

Some progress has been achieved in the meantime. In particular, the restriction to PTime computable distributions was liberalized [BCGL]; Levin's complete problems remained complete (we return to the issue of the liberalized RNP later in this introduction). The reduction theory has been revised [Gu1, BCGL, VL, BG1, BG2]. In particular, deterministic reductions, shown insufficient in [Gu1], have been replaced in [VL] by randomizing reductions; in other words reduction algorithms have been allowed to flip coins. This tuning up of the reduction theory continues in this paper. In Sections 2–5 of this paper, we describe the current state of the reduction theory in full detail and in particular define a clean notion of randomized many-one reductions of randomized decision problems.

A number of additional natural RNP complete problem have been found [Gu1, VL, Gu2, VR] but that number is still very small. Moreover, none of the known complete problems, however natural they are, arose in applications. All of them were designed specially for the purpose of finding additional average-case complete problems. Why have not more problems been found? It is possible that the reduction theory has to be tuned up further. It is certainly true that establishing average-case completeness is much more difficult than establishing worst-case completeness; the range of an average-case reduction of a problem Π_1 to a problem Π_2 cannot comprise only very, very special instances of Π_2. What should be done? Consider the theory of undecidability or NP completeness. In either case, a rich collection of complete problems (complete for recursive enumerability via recursive reductions or NP complete, respectively) has been accumulated which are convenient for reductions to other problems. We need, it seems, to accumulate a rich collection of various average-case complete problems with the hope that these problems will be useful for further reductions. In this connection, Leonid Levin challenged the second author (who started his career an algebraist) to find an average-case complete problem of algebraic character. Such a problem was found in [Gu2]; this paper is a full version of the extended abstract [Gu2].

The matrix decomposition problem involves linear transformations of unimodular matrices. The *modular group* is the multiplicative group $\mathrm{SL}_2(Z)$ of two-by-two integer matrices of determinant 1 (*unimodular matrices*). The notion of linear transformation of $\mathrm{SL}_2(Z)$ does not seem to make sense because $\mathrm{SL}_2(Z)$ is not closed under addition, but this difficulty is not serious. Define a *linear transformation* of $\mathrm{SL}_2(Z)$ to be a function T from $\mathrm{SL}_2(Z)$ to $\mathrm{SL}_2(Z)$ such that $T(\sum X_i) = \sum T(X_i)$ whenever all the X_i and $\sum X_i$ are unimodular matrices. We show in Section 9 that a linear transformation T of $\mathrm{SL}_2(Z)$ uniquely extends to a linear transformation of all two-by-two integer (or even complex) matrices; this gives rise to the standard representation of T by a four-by-four integer matrix. Moreover we will describe a simple (certainly PTime) test to

determine when a given four-by-four integer matrix represents a linear transformation of $\mathrm{SL}_2(Z)$. Identify linear transformations with the four-by-four integer matrices representing them.

Now we are ready to define our randomized decision problem. An instance of Matrix Transformation comprises three components: a unimodular matrix X, a finite set $\mathcal{S}$ of linear transformation of unimodular matrices and a natural number n. The corresponding question is whether there exists a linear transformation $T \in \mathcal{S}^n$ such that $T(X)$ is the unit (or identity) matrix. Here $\mathcal{S}^n$ comprises products $T_1 \cdots T_m$ where $m \leq n$ and each $T_i \in \mathcal{S}$.

Define the size of an integer matrix (whether it is two-by-two or four-by-four) to be the length of the binary representation of the maximal absolute value of the entries. The size of an instance $(X, \mathcal{S}, n)$ is n plus the size of X plus the sum of the sizes of all members of $\mathcal{S}$.

The probability distribution on the instances is rather natural. The three components of a random instance $(X, \mathcal{S}, n)$ are chosen independently. The integer component n is chosen with respect to the default probability function $\dfrac{1}{n(n+1)}$. (The choice of the default distribution does not matter much [Gu1].) To choose the unimodular component X, first choose a positive integer k with respect to the default distribution and then choose X randomly (with respect to the uniform probability distribution) among all unimodular matrices of size k. An auxiliary probability distribution on linear transformations is defined similarly; linear transformations of the same size have the same probability and the probability to have size k equals the default probability of k. Finally, the probability of $\mathcal{S}$ is proportional to the product of the probabilities of the members of $\mathcal{S}$. This completes the definition of Matrix Transformation.

We reduce Bounded Halting to Matrix Transformation and this way prove Matrix Transformation is complete for RNP via randomized many-one reductions. It remains complete under various restrictions on the cardinality of $\mathcal{S}$ and/or the number n; see Section 10 in this connection.

Actually, we prove only that Matrix Transformation is hard for RNP. It is obvious that (the unrandomized version of) Matrix Transformation is NP. Checking that the probability distribution is PTime is routine and we ignore it. We mentioned already that the definition of RNP has been liberalized in [BCGL] by allowing more general distributions called *samplable*. Impagliazzo and Levin proved that every NP search problem with samplable distribution reduces via many-one PTime computable reductions to an NP search problem with PTime computable distribution [IL] (see also [BG2]). In an appropriate sense a search problem with a PTime computable distribution reduces to a decision problems with PTime computable distribution [BCGL]. Thus, Matrix Transformation is hard for the class of NP search problems with samplable distributions. Our reduction of Bounded Halting to Matrix Transformation can

be easily modified to obtain a many-one randomized reduction of the search version of Bounded Halting to the search version of Matrix Transformation. Thus, the search version of Matrix Transformation is complete for the class of NP search problems with samplable distributions. The question remains whether Matrix Transformation (or Bounded Halting) is complete for the class of NP decision problems with samplable distributions. In the usual NP theory, decision problems are easily reducible to search problems. The situation is different in the average-case theory. We intend to consider these issues in [BG3].

A simpler version of Matrix Transformation is obtained by making $\mathcal{S}$ just a set of unimodular matrices and asking whether another unimodular matrix X can be represented as a product of at most n matrices from $\mathcal{S}$. This bounded version of the classical membership problem [Mi, p. 511] for $\mathrm{SL}_2(Z)$ is NP complete; see Section 11 in this connection. However, we don't think that the naturally randomized version of the bounded membership (Section 11) problem is complete for the average case. Indeed, there are indications that it is solvable in time polynomial on average. However, Venkatesan and Rajagopalan proved that the same problem for higher-dimension matrices is complete for the average case [VR].

Since we deal almost exclusively with randomized decision problems, the term "decision problem" will usually mean "randomized decision problem"; similarly, the term "search problem" will usually mean "randomized search problem".

Acknowledgment We thank Suzanne Zeitman for allowing us to publish her proof that Integer Sum is NP complete. We thank Abraham Sharell, the team of Belanger and Wang, and the referees for pointing out various flaws in the previous versions of this paper.

2 Domains

As a general framework for the study of average case complexity, we use domains [Gu2, BG1, BG2].

Definition 2.1 A *domain* X consists of:

- An underlying set, the *universe* of X, often called X as well, comprising strings in some alphabet Σ_X;
- A size function, assigning to each $x \in X$ a positive integer $|x| = |x|_X$ called the *size* of x; and
- A probability distribution $\mathbf{P}_X$ on X.

We require elements of the domain to be strings in order to use the usual computation model based on the Turing machine. In the rest of the paper, an algorithm

is a Turing machine. Traditional concepts of (worst-case) complexity are defined by means of the size function $|x|$. Concepts of average-case complexity are defined by averaging with respect to the probability distribution $\mathbf{P}_X$. As was pointed out by Levin [Le] and discussed in some detail in [BCGL, Gu1], the most obvious definition of the concept "polynomial time on average" has inappropriate consequences, and some care is needed to obtain a suitable definition. We use the following definition due to Levin [Le], as modified in [BG1] to allow ∞ as a value.

Definition 2.2 Let T be a function from a domain X to the interval $[0, \infty]$ of the real line augmented with ∞. T is *linear on average* if $T(x)/|x|$ has finite expectation,

$$\mathop{\mathbb{E}}_{x} \frac{1}{|x|} T(x) = \sum_{x} \mathbf{P}_X(x) \frac{1}{|x|} T(x) < \infty,$$

and T is *polynomial on average*, abbreviated AP, if it is bounded by a polynomial of a function that is linear on average. In other words, T is AP if, for some $\varepsilon > 0$,

$$\mathop{\mathbb{E}}_{x} \frac{1}{|x|} (Tx)^{\varepsilon} = \sum_{x} \mathbf{P}_X(x) \frac{1}{|x|} (Tx)^{\varepsilon} < \infty \ .$$

We use the convention that $0 \cdot \infty = 0$; thus, an AP function can take the value ∞ but only at points of probability 0.

Lemma 2.3 ([Gu1])

- *If $E(|T(x)| \mid |x| = l)$ is bounded by a polynomial of l, then T is AP.*
- *On any domain, the collection of AP functions is closed under addition and multiplication.*

Definition 2.4 A (deterministic) algorithm, taking elements of a domain X as inputs, is *polynomial time on average* or *AP time* if its running time on input x is an AP function of x.

We consider the running time to be ∞ if the algorithm fails to terminate, so an AP time algorithm must terminate on all inputs of nonzero probability. In general, we take the point of view that instances of zero probability do not matter. By following that line consistently, which we try to do, one often has the luxury of throwing elements of zero probability out and supposing, when convenient, that no element of the domain X in question has zero probability or throwing elements of zero probability in and supposing, when convenient, that every string over Σ_X is an element of X. However, we do not go so far as to identify two domains if one of them is obtained from the other by eliminating some elements of zero probability.

In the case of domains with finitely many elements, it would be natural to call a domain uniform if all elements have the same probability. This definition makes no

sense in the case of infinite domains, which is the only case of interest to us. Another natural way to define uniform domains requires a default probability distribution on positive integers; it is customary to assign the probability $\frac{1}{n(n+1)}$ to a positive integer n. The choice of the default probability distribution does not matter much; see [Gu1] in this connection.

Definition 2.5 PI is the domain of positive integers such that $|n| = n$ and the probability of any n is $1/[n(n+1)]$. The probability of a number n in PI is called the *default* probability of n.

Definition 2.6 A domain is *uniform* if it has a finite number of elements of any given size, all elements of a given size have the same probability, and $\mathbf{P}\{x : |x| = n\}$ equals the default probability of n.

Here are two examples of uniform domains.

Definition 2.7 BS is the uniform domain of non-empty binary strings where the size of a string is its length. FRACTION is the uniform domain of fractions a/b where a, b are relatively prime positive integers, $a \leq b$ and the size of a/b is the length $\lceil log_2(b+1) \rceil$ of the (shortest) binary notation for b.

Definition 2.8 A domain Y with universe V is a *subdomain* of a domain X if V is a subset of (the universe of) X and $\mathbf{P}_X(V) > 0$ and $|x|_Y = |x|_X$, $\mathbf{P}_Y(x) = \mathbf{P}_X(x)/\mathbf{P}_X(V)$ for every $x \in V$. Y is also called the *restriction* $X|V$ of X to V.

Definition 2.9 The direct product $X \times Y$ of domains X and Y is the domain of pairs (x, y), where $x \in X$ and $y \in Y$, such that $|(x, y)| = |x| + |y|$ and $\mathbf{P}(x, y) = \mathbf{P}_X(x) \times \mathbf{P}_Y(y)$.

The direct product construction allows us to define powers $X^2, X^3, \ldots$ of a given domain X. Sometimes it is more natural to deal with subsets rather than sequences of elements of a given domain.

Definition 2.10 For each positive integer σ, $\mathrm{Set}_\sigma(X)$ is the domain of σ-element subsets S of a given domain X such that the size $|S| = \sum_{x \in S} |x|$ and the probability $\mathbf{P}(S)$ is proportional to the product $\prod_{x \in S} \mathbf{P}_X(x)$, i.e., $\mathbf{P}(S) \propto \prod_{x \in S} \mathbf{P}_X(x)$.

There are various natural domains D of all finite nonempty subsets of a given domain X such that each $Set_\sigma(X)$ is a subdomain of D.

Definition 2.11 *Set*(X) *is the domain of nonempty subsets of* X *such that* $|S| = \sum_{x \in S} |x|$ *and* $\mathbf{P}(S) \propto \prod_{x \in S} \mathbf{P}_X(x)$.

We need to check that $\sum_S \mathbf{P}(S)$ converges. For each positive integer n, we have

$$1 = \left(\sum_{x\in X} \mathbf{P}(x)\right)^n = \sum_{x_1,\ldots,x_n\in X} \mathbf{P}(x_1)\cdot\cdots\cdot\mathbf{P}(x_n).$$

In the sum on the right, each $\mathbf{P}(S)$ occurs $n!$ times (and there are some additional terms with repeated x's). So $\sum_{S:\,|S|=n} \mathbf{P}(S) \le \frac{1}{n!}$.

Another possibility is to define a probability distribution on the collection of nonempty finite subsets of X which corresponds to the following experiment: first choose a positive integer n with respect to the default probability and then choose an n-element subset with respect to $Set_n(X)$; call this alternative domain $\mathrm{Set}'(X)$.

In many respects, AP functions differ from polynomially bounded functions. Here is one illustration.

Proposition 2.12 *For every countable family $\{f_1, f_2, \ldots\}$ of AP functions on BS there exists an AP function F on BS that is not majorized by any f_i.*

Proof By induction on i, select elements x_i of BS such that:

1. $|x_i| > |x_j|$ for all $j < i$, and
2. $2^{|x_i|} > f_i(x_i)$.

Such elements exist because each f_i is AP. Define

$$F(x) = \begin{cases} 2^{|x|} & \text{if } \exists i(x = x_i) \\ 0 & \text{otherwise.} \end{cases}$$

No f_i majorizes F because $F(x_i) > f_i(x_i)$. It remains to check that F is AP. We have

$$\sum_x \frac{1}{|x|} F(x)\mathbf{P}(x) \approx \sum_i \frac{1}{|x_i|} 2^{|x_i|} \frac{1}{|x_i|^2} 2^{-|x_i|} = \sum_i \frac{1}{|x_i|^3}$$

and the last sum is finite because all elements x_i are of different lengths. □

3 Domain reductions

In this section, we define and discuss many-one reductions, both deterministic and randomizing, between domains. These are the only sort of reductions used in this paper.

Definition 3.1 A function f from domain X to domain Y satisfies the *domination condition* if $\mathbf{P}_X[f^{-1}(fx)]/\mathbf{P}_Y(fx)$ is AP on X.

Corollary 3.2 *An injective function f from domain X to domain Y satisfies the domination condition if and only if $\mathbf{P}_X(x)/\mathbf{P}_Y(fx)$ is AP on X.*

Theorem 3.3 ([BG1]) *Let f be an arbitrary function from a domain X to a domain Y. The following statements are equivalent:*

- *For every AP function T on Y, the composition $T \circ f$ is AP on X, and*
- *$|f(x)|_Y$ is AP on X and f satisfies the domination condition.*

This theorem, with T regarded as the running time of an algorithm on Y, suggests

Definition 3.4 A *deterministic reduction* from a domain X to a domain Y is an AP time computable function f from X to Y such that $|f(x)|_Y$ is AP on X and f satisfies the domination condition.

Such a reduction and an AP time algorithm on Y yield an AP time algorithm on X.

Corollary 3.5 *Deterministic reductions of domains compose. Therefore the relation of deterministic reducibility of domains is transitive.*

Recall also the two domains $\mathrm{Set}(X)$ and $\mathrm{Set}'(X)$ of finite nonempty subsets S of X defined above.

Lemma 3.6 *The identity function reduces Set(X) to Set$'(X)$ and Set$'(X)$ to Set(X).*

Proof is obvious. □

Lemma 3.7 *Let a function f reduce a domain X to a domain Y and suppose that f is one-to-one. Then, for each positive integer σ, the function*

$$F\{x_1, \ldots, x_\sigma\} = \{f(x_1), \ldots, f(x_\sigma)\}$$

reduces Set$_\sigma(X)$ to Set$_\sigma(Y)$.

Proof Let σ be an arbitrary positive integer, $A = \mathrm{Set}_\sigma(X)$ and $B = \mathrm{Set}_\sigma(Y)$. It suffices to prove that F satisfies the domination condition. Since F is one-to-one, it suffices to prove that $\dfrac{\mathbf{P}_A(S)}{\mathbf{P}_B(F(S))}$ is AP on $\mathrm{Set}_\sigma(X)$.

Since f satisfies the domination condition, the ratio $\rho(x) = \mathbf{P}_X(x)/\mathbf{P}_Y(fx)$ is AP on X. Ignoring constant factors, we have

$$\frac{\mathbf{P}_A(S)}{\mathbf{P}_B(FS)} \approx \frac{\prod_{x\in S} \mathbf{P}_X(x)}{\prod_{x\in S} \mathbf{P}_Y(fx)} = \prod_{x\in S} \rho(x).$$

It remains to prove that the last product is AP on A.

Let δ witness that ρ is AP so that $\sum_{x\in X} \rho(x)^\delta \frac{1}{|x|}\mathbf{P}(x) < \infty$. It follows that $\sum_{x\in X} \rho(x)^{\delta/\sigma} \frac{1}{|x|^{1/\sigma}}\mathbf{P}(x) < \infty$. For, the terms where $\rho(x)\frac{1}{|x|} \leq 1$ remain ≤ 1 and

therefore sum up to at most 1, and the other terms become smaller. The number $\varepsilon = \delta/\sigma$ witnesses that $\prod_{x\in S}\rho(x)$ is AP on $\mathrm{Set}_\sigma(X)$.

$$\sum_S \left(\prod_{x\in S}\rho(x)\right)^\varepsilon \frac{1}{|S|}\prod_{x\in S}\mathbf{P}(x) \le \sum_S \prod_{x\in S}\frac{\rho(x)^\varepsilon \mathbf{P}(x)}{|x|^{1/\sigma}} \le \left[\sum_{z\in X}\frac{\rho(z)^\varepsilon \mathbf{P}(z)}{|z|^{1/\sigma}}\right]^\sigma < \infty.$$

The first of the three inequalities holds because the geometric mean $\prod_{x\in S}|x|^{1/\sigma}$ is bounded by the arithmetical mean of the same numbers $|x|$ which is bound by the size $|S|$. To prove the second inequality, notice that every summand on the left side is obtained when you multiply the σ copies of the infinite series. Concerning the third inequality, we have already checked that the infinite series in square brackets converges. It remains to apply Lemma 2.3. □

Lemma 3.8 *Suppose that domains X and Y have the same universe and the same probability distribution. The identity function deterministically reduces X to Y if and only if the size function of Y is AP on X.*

Proof Use Theorem 3.3. □

Proviso 3.9 *Restrict attention to domains X such that the size function of X is AP on the domain X' obtained from X by redefining the size of a string as its length.*

Corollary 3.10 *A function f from a domain X to a domain Y reduces X to Y if and only if f is AP time computable and satisfies the domination condition.*

Proof Let Y' be the domain obtained from Y by redefining the size as length. Since the length $\ell(f(x))$ of $f(x)$ is bounded by the time needed to compute $f(x)$, it is AP on X. Thus f reduces X to Y'. Now use Lemma 3.8 and the transitivity of the deterministic reducibility relation. □

Reductions are used in the usual way to define the notion of a complete problem in a complexity class, i.e., a problem in the class to which all problems in the class are reducible. Unfortunately, deterministic reductions are too weak to yield a good notion of completeness; see [Gu1] where it is shown that complete problems in this sense must (under the reasonable assumption that nondeterministic exponential time differs from deterministic exponential time) have too special probability distributions (non-flat, in the terminology of [Gu1]). Therefore, we use the larger class, suggested in [VL], of randomizing reductions, i.e., we allow the computation of the reducing function to flip coins. In order to introduce randomizing reductions, we need some auxiliary notions.

Terminology and Notation A set S of binary strings satisfies the *prefix condition* if no string in S is a prefix of a different string in S. If Δ is a subset of the cartesian product $U \times V$ of sets U and V then, for each $x \in U$, $\Delta(x) = \{y : (x, y) \in \Delta\}$.

The notion of dilation was introduced in [Gu1] and used in [BG1, BG2]. The idea is to combine the probability distribution on instances and the probability distribution of coin flips into one probability distribution. In the following definition, think of Δ as the set of pairs (x, s) where x is an input to a randomizing algorithm and s is a sequence of coin flips just sufficient to make that algorithm, with input x, produce an output.

Definition 3.11 (cf. [Gu1, BG1]) A *dilation* of a domain X is a domain Δ such that

- The universe of Δ is a subset of $X \times \mathrm{BS}$ such that, for every $x \in X$ of non-zero probability, $\Delta(x)$ is non-empty and satisfies the prefix condition,
- The size function $|(x, s)| = |x|$, and
- The probability distribution $\mathbf{P}(x, s) = \mathbf{P}(x)\frac{2^{-|s|}}{\sum_{t \in \Delta(x)} 2^{-|t|}}$.

Definition 3.12 Let Δ be a dilation of a domain X. Then

$$\mathrm{Density}_\Delta(x) = \sum_{s \in \Delta(x)} 2^{-|s|}, \text{ and } \mathrm{Rarity}_\Delta(x) = \frac{1}{\mathrm{Density}_\Delta(x)}.$$

Further, Δ is *nonrare* if the rarity function $\mathrm{Rarity}_\Delta(x)$ is AP on X. Δ is *almost total* if $\mathrm{Rarity}_\Delta(x) = 1$ for every x of nonzero probability; in terms of coin flips, that means that, if we repeatedly flip a fair coin to produce a string of 0's and 1's, then, with probability 1, we shall eventually obtain a string in $\Delta(x)$. Finally, Δ is *trivial* if, for every $x \in X$ of non-zero probability, $\Delta(x)$ contains the empty string (and therefore contains no other string).

Theorem 3.3 generalizes to randomizing reductions.

Theorem 3.13 ([BG2]) *Suppose that Γ is a nonrare dilation of a domain X and f is a function from Γ to Y. Then the following statements are equivalent:*

- *$|f(x)|_Y$ is AP on Γ and f satisfies the domination condition,*
- *For every nonrare dilation Δ of Y and every AP function T on Δ, the composition $T(f(x, s), t)$ is AP on the dilation $\Gamma * \Delta$ of A that comprises pairs (x, st) such that $(x, s) \in \Gamma$ and $(f(x, s), t) \in \Delta$.*

Definition 3.14 A *(randomizing) reduction* of a domain X to a domain Y consists of a nonrare dilation Γ of X and a deterministic reduction f of Γ to Y.

Randomizing reductions of domains compose in the following sense. If Γ and Δ are non-rare dilations of X and Y respectively and and if $f : \Gamma \longrightarrow Y$ and $g : \Delta \longrightarrow Z$ are randomizing reductions of X to Y and of Y to Z, then there is a composite reduction $g \circ f : \Gamma * \Delta \longrightarrow Z$ of X to Z. Here $\Gamma * \Delta$ is as in Theorem 3.13 and $g \circ f$ is defined by $(g \circ f)(x, st) = g(f(x, s), t)$ whenever $(x, s) \in \Gamma$ and $(f(x, s), t) \in \Delta$. (Although this composition is not the ordinary composition of functions, it does yield a category of domains and random functions.)

Corollary 3.15 *The relation of (randomized) reducibility is transitive.*

Definition 3.16 Let Σ be an alphabet. A *randomizing algorithm* on Σ^* is an algorithm A on $\Sigma^* \times \mathrm{BS}$, but the two input strings, a string x over Σ and a binary string s, play different roles. The string x is viewed as the input, and the string s is viewed as a sequence of coin flips. It is supposed that A does not flip a coin unless the computation requires another random bit.

Definition 3.17 A dilation Δ of a domain X is *(AP time) certifiable* if there exists a randomizing algorithm A on Σ_X^* such that:

- For every $x \in X$ of nonzero probability and every binary string s, A outputs YES on input (x, s) if and only if $(x, s) \in \Delta$, and
- The computation time of A is AP on Δ.

The need for certifiable reductions arises when one deals with decision problems. We shall consider algorithms which, given an input x of non-zero probability, produce a correct output on any random string $s \in \Delta(x)$ but may produce an incorrect output on $s \notin \Delta(x)$. When the correctness of the output cannot be verified efficiently, the certifiability of Δ will be needed to justify believing the output.

Definition 3.18 A reduction (Γ, f) of a domain X to a domain Y is *certifiable* if Γ is certifiable.

Lemma 3.19 *Certifiable reductions of domains compose.*

Proof Chase the definitions and apply Theorem 3.3. □

As an example of how much easier it may be to deal with randomizing reductions, consider the problem of reducing BS to FRACTION. In order to avoid trivial solutions, like constant functions, let us require that different elements of BS are taken to different elements of FRACTION. The problem is easily solved with the help of randomization.

Lemma 3.20 *There exists a randomized reduction (Γ, f) from BS to FRACTION such that $f(x_1, s_1) \neq f(x_2, s_2)$ whenever $x_1 \neq x_2$.*

Proof $\Gamma(x)$ comprises all binary strings s of length $|x|$ such that the numbers (represented by binary strings) s and $1x$ are relatively prime. Since the chance that a random s is relatively prime to $1x$ is sufficiently large [HW], Γ is nonrare. $f(x,s) = \dfrac{s}{1x}$. □

4 Search problems

Definition 4.1 A *(randomized) search problem* $\mathrm{SP}(X,W)$ is given by a domain X (of *instances*) and a PTime computable relation $W(x,w)$ (the *witness relation*) between elements of X and arbitrary strings in a fixed alphabet. The problem is: Given an instance x with $W(x) \neq \emptyset$, find an element of $W(x)$ (a *witness* for x).

Definition 4.2 $\mathrm{SP}(X,W)$ is *AP time solvable* if there exist a nonrare dilation Γ of $X|\{x : W(x) \neq \emptyset\}$ and an AP time algorithm M on Γ that, given any $(x,s) \in \Gamma$ with $\mathbf{P}_X(x) > 0$, finds a witness for x. Such a pair (Γ, M) is called an *AP time solution* for $SP(X,W)$. A solution (Γ, M) is *almost total* if Γ is so.

This notion of AP solvability may seem weaker than it is.

Theorem 4.3 ([BG2]) *Every AP time solvable search problem $SP(X,W)$ has an almost total solution.*

Definition 4.4 A *(randomizing) reduction* of $\mathrm{SP}(X,U)$ to $\mathrm{SP}(Y,V)$ consists of

Dilation: A nonrare dilation Γ of X,

Instance transformer: A deterministic reduction f of Γ to Y, and

Witness transformer: A PTime computable function $g((x,s),v)$ such that if $s \in \Gamma(x)$, and $v \in V(f(x,s))$ then $g((x,s),v) \in U(x)$.

Theorem 4.5 ([BG2]) *The reducibility relation on search problems is transitive, and a problem $SP(X,U)$ is solvable in AP time if it is reducible to some problem $SP(Y,V)$ which is solvable in AP time.*

The notion of reduction allows us to define complete problems in the usual way. $\mathrm{SP}(X,W)$ is *complete* for a class $\mathcal{C}$ of search problems if it is in $\mathcal{C}$ and every problem in $\mathcal{C}$ reduces to it.

5 Decision problems

Definition 5.1 A *(randomized) decision problem* $\mathrm{DP}(X, P)$ is given by a domain X of *instances* and a subset P of X. Instances in P are called *positive*, and instances in $X - P$ are called *negative*. The problem is: Given an instance $x \in X$, decide whether x is positive or negative.

Definition 5.2 $\mathrm{DP}(X, P)$ is *AP time solvable* (or *AP time decidable*) if there exist a non-rare certifiable dilation Γ of X and an AP time algorithm M on Γ which, given any element $(x, s) \in \Gamma$ with $\mathbf{P}_X(x) > 0$, decides whether x is positive or negative. The pair (Γ, M) is an *AP time solution* for $\mathrm{DP}(X, P)$. A solution (Γ, M) for $\mathrm{DP}(X, P)$ is *almost total* if Γ is so.

Notice that certifiability is required, as we cannot check whether the output of M (yes or no) is correct. Contrast this with search problems where the assumed computability of the witness relation lets us check whether the output (an alleged witness) is indeed a witness and certifiability is therefore not required.

Again, the notion of AP time solution may seem weaker than it is.

Theorem 5.3 *If DP(X, P) is AP time solvable then it has an almost total AP time solution.*

Proof Let (Γ, M) be an AP time solution for a decision problem $\mathrm{DP}(X, P)$ and let A be a certifying algorithm for Γ. For each $s \in \Gamma(x)$, let s' be the computation of A on (x, s) and s'' be the computation of M on (x, s). Define

$$W = \{(x, (s, s', s'')) : (x, s) \in \Gamma\}.$$

Obviously, the relation W is PTime computable. (The intended algorithm for computing W uses A and M; we need not check $(x, s) \in \Gamma$ because this follows from $A(x, s)$ providing s'.) The dilation Γ and a combination of the algorithms A and M give an AP time solution for the search problem $\mathrm{SP}(X, W)$. By Theorem 4.3, this search problem has an almost total AP time solution (Δ, N). For each $(x, t) \in \Delta$ with $\mathbf{P}_X(x) > 0$, N outputs a triple $(s, s', s'') \in W$. By the definition of W, $s = s(t) \in \Gamma(x)$ and therefore $s(t)''$ is a computation of M deciding whether x is positive or negative.

The dilation Δ is certifiable. Given an instance x of non-zero probability and an arbitrary string t, the desired certifying algorithm A' runs N on (x, t). If a prefix t_0 of t belongs to $\Delta(x)$, N will produce an output on (x, t_0) and A' will output "yes" in the case $t_0 = t$ or "no" in the case t_0 is a proper prefix of t. Suppose that no prefix of t belongs to $\Delta(x)$. Since Δ is almost total, both $t0$ and $t1$ are prefixes of strings in Δ. The computation of N on (x, t) will stop without producing any output, waiting for another random bit; in such a case A' will output "no".

Let N' be the modification of N that, given $(x,t) \in \Delta$ with $\mathbf{P}_X(x) > 0$, outputs only the result (yes or no) of the computation $s(t)''$. The pair (Δ, N') constitutes an almost total AP time solution for $\mathrm{DP}(X,P)$. □

Notice the role of the certifying algorithm A. The certifiability of dilation was unnecessary in the case of search problems, but in the case of decision problems it plays an important role.

Definition 5.4 A *(randomizing) many-one reduction* of $\mathrm{DP}(X,P)$ to $\mathrm{DP}(Y,Q)$ comprises:

- A non-rare certifiable dilation Γ of X, and
- A deterministic reduction f from Γ to Y (the *instance transformer*) satisfying the following *correctness property*: For all $(x,s) \in \Gamma$,

$$f(x,s) \in Q \iff x \in P.$$

Theorem 5.5 *The many-one reducibility relation on decision problems is transitive, and a problem DP(X,P) is AP time decidable if it reduces to some problem DP(Y,Q) that is AP time decidable.*

Proof Use the fact, established in Section 3 that randomizing domain reductions compose. □

Definition 5.6 A many-one reduction (Γ, f) of $\mathrm{DP}(X,P)$ to $\mathrm{DP}(Y,Q)$ is *deterministic* if Γ is trivial. In this case, the reduction (Γ, f) is specified only by the instance transformer f; Γ may be identified with X.

Lemma 5.7 *The identity function deterministically reduces any decision problem* Π *to the decision problem* Π' *obtained from* Π *by redefining the size of a string as its length.*

Proof Use Corollary 3.10. □

A decision problem is *hard* for a class $\mathcal{C}$ of decision problems if every problem in $\mathcal{C}$ reduces to it. A decision problem is *complete* for $\mathcal{C}$ if it belongs to $\mathcal{C}$ and is hard for $\mathcal{C}$.

RNP is the class of decision problems with PTime computable probability distributions. PTime computable distributions are defined in [Le] and analyzed in [Gu1].

6 Positive matrices

We now turn from the general theory of randomizing algorithms and reductions to the specific problem, Matrix Transformation, whose completeness for RNP we shall prove in Section 10. We begin with information about unimodular matrices.

Call a unimodular matrix (i.e. an element of $\mathrm{SL}_2(Z)$, a two-by-two matrix with determinant 1) *positive* if it has no negative entries. Positive matrices form a monoid $PM = \mathrm{SL}_2(N)$. In this section, a column is a column of two relatively prime non-negative integers; for notational simplicity, we view a positive matrix as the pair of its columns. If u is a column, let u_1 be the upper and u_2 the lower components of u. Partially order columns componentwise: $u \leq v$ if $u_1 \leq v_1$ and $u_2 \leq v_2$, and $u < v$ if $u \leq v$ and either $u_1 < v_1$ or $u_2 < v_2$. Define $\max(X)$ to be the maximal entry of a positive matrix X. In this section, $A_0 = \begin{pmatrix} 1 & 0 \\ 1 & 1 \end{pmatrix}$ and $B_0 = \begin{pmatrix} 1 & 1 \\ 0 & 1 \end{pmatrix}$.

Lemma 6.1 *1.* $(u, v) \times A_0 = (u + v, v)$, *and* $(u, v) \times B_0 = (u, u + v)$.

2. If A_0 is a right divisor of a positive matrix (u, v) in PM then $u > v$, and if B_0 is a right divisor of (u, v) in PM then $u < v$.

3. If the maximal entry m of a matrix (u, v) appears in two or more places then $m = 1$.

Proof (1) is obvious, and (2) follows from (1).

(3) If m occurs twice in the same row or the same column, then it divides the determinant 1 and therefore $m = 1$. So assume this doesn't happen. If $v_1 = u_2 = m$ then the determinant cannot be positive. If $u_1 = v_2 = m$ then $1 = u_1v_2 - v_1u_2 \geq m^2 - (m-1)^2 = 2m - 1$ and therefore $m = 1$. □

The second statement of Lemma 6.1 implies that the monoid generated by the matrices A_0 and B_0 is free. This fact is noticed in [Ei, Chapter VI, Section 12]. The following theorem should be known too, but we don't have an appropriate reference.

Theorem 6.2 *The monoid PM is isomorphic to the monoid BS of binary strings. The two indecomposable non-unit elements are the matrices A_0 and B_0.*

Proof Since A_0 and B_0 generate a free monoid, it suffices to prove that every non-unit positive matrix (u, v) is a product of matrices A_0 and B_0. Define *weight*$(u) = u_1 + u_2$ and *weight*$(u, v) = \mathrm{weight}(u) + \mathrm{weight}(v)$. The proof is an induction on $s = \mathrm{weight}(u, v)$. Since the entries of the main diagonal are not zero, $s \geq 2$.

The case $s \leq 3$ is easy: A_0 and B_0 are the only non-unit matrices of weight ≤ 3. Suppose that $s > 3$. Then $m = \max(u, v) > 1$. Exploiting the symmetry, we may suppose that m appears in u. If $u_1 = m$ then $1/m = (u_1v_2 - v_1u_2)/m > v_2 - u_2$ and

therefore $u_2 \geq v_2$. Similarly, if $u_2 = m$ then $u_1 \geq v_1$. Thus, the column $u - v$ has nonnegative entries. The determinant of $(u-v, v)$ equals 1 and therefore $(u-v, v)$ is an element of $\mathrm{SL}_2(N)$. By the induction hypothesis, $(u-v, v)$ is a product of matrices A_0 and B_0. By Lemma 6.1(1), $(u, v) = (u - v, v) \times A_0$. □

Corollary 6.3 *If a positive matrix (u, v) is not the unit matrix then one of the two columns is greater than the other.*

Proof The fact has been established in the proof of Theorem 6.2. □

Call the greater column of a non-unit positive matrix *major*; in the case of the unit matrix, call either column *major*. The other column of the matrix will be called *minor*.

Lemma 6.4 *The major column and one bit indicating whether it is the first or the second column uniquely define the minor column.*

Proof Without loss of generality, the given matrix (u, v) is not the unit matrix. It follows that both components of the major column are positive. By virtue of symmetry, suppose that u is the major column. We show that the minor column v is the least column such that $u_1v_2 - u_2v_1 = 1$. Let w be any column such that $u_1w_2 - u_2w_1 = 1$. Then $u_1(w_2 - v_2) = u_2(w_1 - v_1) = u_1u_2k$ for some k because u_1 and u_2 are relatively prime; hence $w_1 = v_1 + ku_1$ and $w_2 = v_2 + ku_2$. If $k < 0$ then either w_1 or w_2 is negative. Hence $k \geq 0$ and therefore $w_1 \geq v_1$, $w_2 \geq v_2$. □

Lemma 6.5 *There exists a PTime algorithm that, given a column u, computes the minor column of the unique positive unimodular matrix with the first and major column u.*

Proof Use the extended Euclid's algorithm [Kn1]. □

Remark 1 Instead of columns, we could use rows above in this section. This would cause some insignificant changes in Lemma 6.1 (for example, the first statement would say that $A_0 \times (u, v) = (u, u + v)$, and $B_0 \times (u, v) = (u + v, v)$ where u is the upper row of the given matrix and v is the lower row), but Corollary 6.3 and Lemma 6.4 remain true.

Let $\ell(n)$ be the length of the binary notation for n.

Definition 6.6 We define a domain structure on the monoid PM. It is the uniform domain with the size function $|X| = \ell(\max(X))$. Thus, PM is the monoid and domain of positive matrices.

Strictly speaking, the elements of a domain should be strings. For this purpose, we may regard a matrix as a list of its entries in binary notation. Then Proviso 3.9 is satisfied.

Lemma 6.7 *The relative probability* $\mathbf{P}_{PM}[X \mid |X| = l] = \Theta(2^{-2l})$.

Proof Let $g(l) \approx f(l)$ mean that $g(l) = \Theta(f(l))$, i.e., that there exist positive constants c, c' and l_0 such that $cf(l) \leq g(l) \leq c'f(l)$ for all $l \geq l_0$ [Kn2]. It suffices to prove that the number $N(l)$ of positive matrices of size l is $\Theta(2^{2l})$. Recall that $\phi(m)$ is the number of positive integers $n \leq m$ that are prime to m, and that $\Phi(m) = \phi(1) + \ldots + \phi(m) = 3m^2/\pi^2 + O(m \cdot \log m)$ [HW, Theorem 330]. Thus,

$$N(l) \approx \sum_{\ell(m)=l} \phi(m) \approx \Phi(2^l - 1) - \Phi(2^{l-1}) = \Theta(2^{2l}). \qquad \Box$$

By Theorem 6.2, PM is isomorphic to BS as a monoid. There are exactly two isomorphisms of PM onto BS. One of them takes A_0 to 0 and B_0 to 1 while the other one takes A_0 to 1 and B_0 to 0. Let I be the isomorphism that takes A_0 to 0, and let J be the corresponding isomorphism I^{-1} from binary strings to PM. It is easy to check by induction on the length of the given string x that if x starts with a zero (resp. one) then the lower (resp. upper) row of $J(x)$ is major (see "transposed" Lemma 6.1 in Remark 1). Notice that the size of a matrix X may be quite different from the length of the corresponding string $I(X)$. It is easy to see that the isomorphism I is not computable in polynomial time: A matrix $A_0^n = \begin{pmatrix} 1 & 0 \\ n & 1 \end{pmatrix}$ is of size $\ell(n)$ whereas the string $0^n = I(A_0^n)$ is of length n.

Lemma 6.8 *I is AP time computable.*

Proof The following recursive algorithm computes $I(X)$. If X is the unit matrix then $I(X)$ is the empty string. Suppose that $X = (u, v)$ differs from the unit matrix. If u is the major column, $w = u - v$ and $z = I(w, v)$ then $I(X) = z0$, and if v is the major column, $w = v - u$ and $z = I(u, w)$ then $I(X) = z1$. The computation time of that algorithm is essentially proportional to $|I(X)|$, which is AP by Lemma 6.12. $\Box$

The isomorphism J is PTime computable but $\mathbf{P}_{\mathrm{PM}}$ does not dominate $\mathbf{P}_{\mathrm{BS}}$ with respect to J and thus J fails to reduce BS to PM.

Theorem 6.9 $\mathbf{P}_{PM}$ *does not dominate* $\mathbf{P}_{BS}$ *with respect to* J.

Proof By contradiction, suppose that the function $g(x) = \mathbf{P}_{\mathrm{BS}}[x]/\mathbf{P}_{\mathrm{PM}}[Jx]$ is polynomial on average with respect to $\mathbf{P}_{\mathrm{BS}}$ and fix $\varepsilon > 0$ to witness that fact. Thus,

$$\infty > \sum_n \sum_{x:\, |x|=n} \frac{1}{n} (g(x))^{\varepsilon} \frac{1}{n(n+1)} 2^{-n},$$

so, as a function of n, the conditional expectation of $(g(x))^\varepsilon$ for strings of length n, $\sum_{x:\,|x|=n}(g(x))^\varepsilon 2^{-n}$ is $o(n^3)$. We obtain the desired contradiction by proving that this expectation is not bounded by any polynomial of n.

Let $l = |Jx|$. By Lemma 6.7,

$$g(x) = \Theta(l^2 \cdot 2^{2l} \cdot n^{-2} \cdot 2^{-n}).$$

Let $s(x)$ be the sum of the entries of the major row of Jx. Clearly, $s(x) = \Theta(2^l)$. Hence it suffices to prove that the expectation $E_n = \sum_x[(s(x)^2/2^n)^\varepsilon \cdot 2^{-n}]$ of the function $[s(x)^2/2^n]^\varepsilon$ is not bounded by a polynomial of n. (The factor n^{-2} in $g(x)$ won't matter as it is the reciprocal of a polynomial, and l^2 won't matter as it is ≥ 1.) We may restrict attention to $\varepsilon < 1/2$. Let y range over strings of length $n-1$.

Lemma 6.10 *There exists some $\alpha > 1$ such that every*

$$A(y) = (1/2)[s(y0)^{2\varepsilon} + s(y1)^{2\varepsilon}]/s(y)^{2\varepsilon} \geq \alpha 2^\varepsilon$$

.

Proof Let $a > b$ be the two entries of the major row of $J(y)$, and $\gamma = b/a$. Then

$$A(y) = (1/2)[(2a+b)^{2\varepsilon} + (a+2b)^{2\varepsilon}]/(a+b)^{2\varepsilon} = (1/2)[(2+\gamma)^{2\varepsilon} + (1+2\gamma)^{2\varepsilon}]/(1+\gamma)^{2\varepsilon}.$$

Let $\delta = 1/(1+\gamma)$. Then $A(y) = (1/2)[(1+\delta)^{2\varepsilon} + (2-\delta)^{2\varepsilon}]$.

Consider the function $f(t) = t^{2\varepsilon}$ of a real variable t. The graph of f is concave because $f''(t) = 2\varepsilon(2\varepsilon-1)t^{2\varepsilon-2} < 0$. Since $0 < \delta < 1$, the chord C between the points $(1+\delta, f(1+\delta))$ and $(2-\delta, f(2-\delta))$ lies strictly above the chord C_0 between the points $(1, f(1))$ and $(2, f(2))$. Further, the center of the interval $[1+\delta, 2-\delta]$ coincides with the center 1.5 of the interval $[1,2]$, and therefore the center $(1.5, A(y))$ of C lies directly above the center $(1.5, [1+2^{2\varepsilon}]/2)$ of C_0. The arithmetical mean $[1+2^{2\varepsilon}]/2$ of numbers 1 and $2^{2\varepsilon}$ exceeds their geometrical mean 2^ε. Thus, $A(y) > [1+2^{2\varepsilon}]/2 > 2^\varepsilon$. The desired $\alpha = (1/2)[1+2^{2\varepsilon}]/2^\varepsilon$. □

We continue to prove Theorem 6.9. Let $A(y)$ and α be as in Lemma 6.10, and let x range over strings of length n and y over strings of length $n-1$.

$$E_n = \sum_x[(s(x)^2/2^n)^\varepsilon \cdot 2^{-n}] = 2^{-n\varepsilon} \cdot \sum_x[s(x)^{2\varepsilon} \cdot 2^{-n}] \geq 2^{-n\varepsilon} \cdot \sum_y[s(y)^{2\varepsilon} \cdot A(y) \cdot 2^{-(n-1)}].$$

By Lemma 6.10,

$$E_n \geq 2^{-n\varepsilon} \cdot \sum_y[s(y)^{2\varepsilon} \cdot \alpha 2^\varepsilon \cdot 2^{-(n-1)}] = \alpha \sum_y[(s(y)^2/2^{n-1})^\varepsilon \cdot 2^{-(n-1)}] = \alpha E_{n-1}.$$

It follows that E_n is $\Omega(\alpha^n)$ and therefore is not bounded by a polynomial of n. □

Recall the notion of a (simple) continued fraction [HW]. Here is an example:

$$\frac{81}{17} = 4+\frac{13}{17} = 4+\frac{1}{\left(\frac{17}{13}\right)} = 4+\frac{1}{1+\frac{4}{13}} = 4+\frac{1}{1+\frac{1}{\left(\frac{13}{4}\right)}} = 4+\frac{1}{1+\frac{1}{3+\frac{1}{4}}} = [4,1,3,4].$$

Similarly, every positive rational number r can be uniquely represented by a continued fraction $[a_l, \ldots, a_0]$ where a_l is a non-negative integer, and each a_i with $0 < i < l$ is a positive integer, and a_0 is an integer ≥ 2 unless $l = 0$; the integers $a_i \geq 0$ are called partial quotients.

Lemma 6.11 *Suppose that x is a nonempty binary string and let $m \leq n$ be the two entries of the major row of $J(x)$. Then $|x|$ equals the sum $s(n,m)$ of the partial quotients in the continued fraction for n/m.*

Proof If $|x| = 1$ then $m = n = 1$ and $s(n,m) = 1 = |x|$. Suppose that $|x| > 1$. By virtue of symmetry, we may suppose that $x = y0$; the other case is similar. Let (i,j) be the major row of $J(y)$. By Lemma 6.1, the major row (n,m) of $J(x)$ is $(i+j,j)$. It suffices to prove that if $i \leq j$ then $s(n,m) = s(j,i)+1$, and if $i \geq j$ then $s(n,m) = s(i,j)+1$. Since $J(y)$ is not the unit matrix, neither i nor j is zero. In any case, $\frac{n}{m} = \frac{i+j}{i} = 1+\frac{j}{i}$.

If $i \leq j$ and $\frac{j}{i} = [a_l, \ldots, a_0]$ then $\frac{n}{m} = [a_l+1, \ldots, a_0]$, so $s(n,m) = s(j,i)+1$.

If $j < i$ and $\frac{i}{j} = [a_l, \ldots, a_0]$ then $\frac{n}{m} = 1+\frac{1}{\left(\frac{i}{j}\right)} = [1, a_l, \ldots, a_0]$,

so $s(n,m) = s(i,j)+1$. □

Lemma 6.12 *$|I(X)|$ is AP on PM.*

Proof Let $s(n,m)$ be as in Lemma 6.11. We use the following strong result of Yao and Knuth [YK]: $\sum_{m=1}^{m=n} s(n,m) = (6n/\pi^2)(\ln n)^2 + O(n(\log n)(\log\log n)^2) = \Theta(n(\log n)^2)$. Let X be a matrix of size $l > 0$, and let $a(X) < b(X)$ form the major row of the matrix X. Then $\sum_{X:\, b(X)=n} s(b(X), a(X)) = \Theta(n(\log n)^2)$. By Lemma 6.11 $\sum_{b(X)=n} |I(X)| = \Theta(n(\log n)^2)$ and therefore $\sum_{X:\,|X|=l} |I(X)| = \Omega(2^l \cdot l^2 2^l)$. Now use Lemma 6.7 to check that the expectation of $|I(X)|$ with respect to the conditional probability $\mathbf{P}_{\rm PM}[X \mid |X| = l]$ is bounded by a polynomial of l. It follows (see Lemma 2.3) that $I(X)$ is AP on PM. □

7 Positive Matrix Correspondence Problem

Definition 7.1 Let T be a nondeterministic Turing machine with binary input alphabet. The *bounded halting problem* $\mathrm{BH}(T)$ is the randomized decision problem with domain $\mathrm{BS} \times \mathrm{PI}$ such that an instance (x, n) is positive if and only if T has a halting computation of length $\leq n$ on x. Call an instance (x, n) of $\mathrm{BH}(T)$ *robust* if either T has a halting computation of length $\leq n$ on x or else T has no halting computation on x at all. $\mathrm{RBH}(T)$ is the restriction of $\mathrm{BH}(T)$ to robust instances.

Definition 7.2 WBS is the domain of binary strings where $|x|$ is the length of x and $\mathbf{P}(x) = \mathbf{P}_{\mathrm{PM}}(Jx)$. Let T be a nondeterministic Turing machine with binary input alphabet. The *weird halting problem* $\mathrm{WH}(T)$ and its robust version $\mathrm{RWH}(T)$ are similar to $\mathrm{BH}(T)$ and $\mathrm{RBH}(T)$ except the domain is $\mathrm{WBS} \times \mathrm{PI}$ rather than $\mathrm{BS} \times \mathrm{PI}$.

Lemma 7.3 *For a certain U, RWH(U) is hard for RNP.*

Proof Some $\mathrm{RBH}(T)$ is RNP complete, by Corollary 1 of Theorem 1 in [Gu1]. (Actually, a slightly different version of bounded halting problems was considered in [Gu1]. It was supposed there that $n > |x|$ and $\mathbf{P}(x, n) \propto n^{-3} 2^{|x|}$. However the same proof works. Also, the identity function deterministically reduces that older version of every $\mathrm{RBH}(T)$ to the new one.) Thus it suffices to reduce an arbitrary $\mathrm{RBH}(T)$ to an appropriate $\mathrm{RWH}(U)$. We will do just that.

One may be tempted to take $U = T$ and to use the identity mapping as a reduction. By Theorem 6.9, the identity function fails to do the job.

For every binary string s, let $N(s)$ be the integer with binary representation $1s$. If $N(s) = k$, let $S(k) = s$. Given a binary string y, the desired U computes $x = S(\max(J(y)))$, turns itself into T and then runs on input x. We construct a reduction (Γ, f) from $\mathrm{RBH}(T)$ to $\mathrm{RWH}(U)$. Here Γ is a dilation of $\mathrm{BS} \times \mathrm{PI}$, and f is a function from Γ to $\mathrm{WBS} \times \mathrm{PI}$.

Define $\Gamma(x, n)$ to comprise binary strings s of length $\geq |x| - 1$ such that $N(s)$ is prime to and less than $N(x)$. It is obvious that the dilation Γ is certifiable. By Theorem 328 in [HW], the number $\phi(k)$ of positive integers prime to and less than k is $\Omega\left(\frac{k}{\log\log k}\right)$. Notice that, if j is relatively prime to k, then so is $k - j$. Thus, half of the integers counted by $\phi(k)$ are $\geq \frac{k}{2}$, and so the cardinality of $\Gamma(x, n)$ is $\geq \frac{1}{2}\phi(N(x))$. It follows that Γ is not rare:

$$\mathrm{Density}_{\Gamma}(x) = \sum_{s \in \Gamma(x)} 2^{-|s|} \geq \sum_{s \in \Gamma(x)} 2^{-|x|} = \Omega\left(\frac{1}{\log\log N(x)}\right) = \Omega\left(\frac{1}{\log|x|}\right).$$

By Lemma 6.4, for each $(x, n, s) \in \Gamma$, there exists a unique positive unimodular matrix $M(x, s)$ with first and major column $(N(x), N(s))$. View M as a function on Γ; we check that it reduces Γ to PM. By Lemma 6.5, the function M is PTime

computable. To check the domination property, use Lemma 6.7 and the fact that M is injective.

The desired f is given by $f((x,n),s) = (y, t(x,s)+n)$ where $y = I(M(x,s))$ and $t(x,s)$ is the time that U needs to convert y into x. First, we check that f takes robust instances of BH(T) to robust instances of WH(U) and has the correctness property. By the definition of U, it halts on y if and only if T halts on x. Moreover, if T halts within $\leq n$ steps on x then U halts within $t(x,s)+n$ steps on y. Now suppose that U halts on y. Then T halts on x. Since (x,n) is robust, T halts within n steps on x. Hence U halts within $t(x,s)+n$ steps on y.

Next, we check that f is AP time computable on Γ. We have to prove that y and $t(x,s)$ are computable in AP time relative to Γ. In the case of y this follows from the definition $y = I(M(x,s))$, the fact that M reduces Γ to PM, Lemmas 6.8 and 6.12 and Theorem 3.3. Now consider $t(x,s)$. It is the time that U needs to compute $x = S(\max(J(y)))$ from y, which is bounded by a polynomial of $|y|$ because J, max and S are all PTime computable. So, by Lemma 2.3, $t(x,s)$ is AP on Γ. In addition, it is computable in AP time relative to Γ, because it can be computed by first computing y (which we already saw takes AP time) and then computing x from y while running a clock (which takes time essentially $t(x,s)$).

Finally, we check that f has the domination property. Notice that f is one-to-one, so we can use Corollary 3.2. In addition, the probability functions for WBS $\times$ PI and its restriction to robust instances differ by a constant factor, so we can compute with the former instead of the latter. We have

$$\frac{\mathbf{P}_\Gamma((x,n),s)}{\mathbf{P}_{\mathrm{WBS}\times\mathrm{PI}}(y, t(x,s)+n))} = \frac{\mathbf{P}_\Gamma((x,n),s)}{\mathbf{P}_{\mathrm{WBS}}(y)} \times (t(x,s)+n)(t(x,s)+n+1).$$

The fraction on the right is AP on Γ because $\mathbf{P}_{\mathrm{WBS}}(y) = \mathbf{P}_{\mathrm{PM}}(M(x,s))$ and M has the domination property. Now use Lemma 2.3 and the fact that $t(x,s)$ is AP on Γ. □

The direct product PM $\times$ PM is a domain and monoid of positive matrix pairs; the multiplication of matrix pairs is componentwise: $(X_1,Y_1)\times(X_2,Y_2) = (X_1X_2, Y_1Y_2)$. If S is a set of positive matrix pairs, let S^n comprise products $P_1 \times \ldots \times P_m$ where $m \leq n$ and each $P_i \in S$. In the following definition, PMC stands for Positive Matrix Correspondence.

Definition 7.4 For each positive integer σ, PMC(σ) is the decision problem with domain

$$\mathrm{PM} \times \mathrm{Set}_\sigma(\mathrm{PM}\times\mathrm{PM}) \times \mathrm{PI}$$

where an instance (A,S,n) is positive if and only if there exists a pair (X,Y) in S^n such that $AX = Y$. An instance (A,S,n) of PMC(σ) is *robust* if either $AX = Y$ for some pair (X,Y) in S^n or else the whole submonoid of PM $\times$ PM generated by S

has no pair (X, Y) with $AX = Y$. RPMC(σ) is the restriction of PMC(σ) to robust instances (A, S, n).

Theorem 7.5 *Some RPMC(σ) is hard for RNP.*

Remark 2 Replacing PM with BS in the definition of RPMC(σ) gives a variant RPCP(σ) of the Post Correspondence Problem; this variant has been proved RNP complete for not too small σ in [Gu1]. (Actually, RPCP(σ) is slightly different from the version in [Gu1] but the difference is immaterial. For readers with Section 5.1 of [Gu1] before them, we indicate the changes needed in the completeness proof to cover our variant. Remove the clause L0 from the definition of $L(w)$ to obtain a reduced set $L'(w)$ of pairs. Instead of the instance $(L(w), p(m))$, use the instance $(ws_0, L'(w), p(m))$.) If we ignore probabilities and deal with decision problems only then the isomorphism J of Section 3 gives rise to a polynomial time reduction of RPCP(σ) to PMC(σ). Unfortunately, this reduction fails to have the domination property and it is difficult to alter in any way: The correctness property of the reduction is too closely related to fact that J is an isomorphism. Theorem 7.5 is not proved by a reduction from RPCP(σ), but the proof of completeness of RPCP(σ) is used in an essential way.

Proof of Theorem 7.5. Fix a Turing machine U witnessing Lemma 7.3. We will reduce RWH(U) to RPMC(σ) for appropriate σ. The variant RPCP(σ) of the Post Correspondence Problem was defined in a remark above. According [Gu1, Section 5] (with changes indicated in the remark), there exists a PTime reduction

$$F(x, n) = (xx', K(x), p(n))$$

of RBH(U) to some RPCP(σ) such that $|x'| = O(\log |xx'|)$ and $|K(x)| = O(\log |xx'|)$. (Concerning the size bounds, see in particular Lemma 5.1 in [Gu1].) Extend the isomorphism J to sequences of pairs of binary strings. The function

$$G(x, n) = (J(xx'), J(K(x)), p(n))$$

is the desired reduction of RWH(U) to RPMC(σ). Clearly, G is AP time computable and has the correctness property. Ignoring factors bounded by a polynomial of $|x|+n$ from above and by an inverse polynomial of $|x| + n$ from below, we have:

$$\mathbf{P}_{\mathrm{RPMC}(\sigma)}[G(x, n)] \geq \mathbf{P}_{\mathrm{PM}}[J(x)] = \mathbf{P}_{\mathrm{RWH}}(x, n).$$

The inequality here depends on the fact that $|x'|$ and $|K(x)|$ are logarithmically small compared to $|x|$. This ensures that the entries in the matrices $J(K(x))$ have sizes logarithmic relative to $|x|$ and the entries in $J(xx') = J(x)J(x')$ have sizes $|J(x)| + O(\log |x|)$. These logarithms in the sizes contribute polynomials in $|x|$ as factors in the probabilities, and such factors are ignored. □

Remark 3 In [Gu2], Theorem 7.5 has been stated in a stronger form; instead of RPMC(σ) it referred to RPMC(σ, c) where the (A, S, n) were required to have $|S| = O(\log |A|)$. Our proof of the theorem does not establish the stronger result automatically. Although $|K(x)|$ is logarithmically small compared to $|x|$, we cannot conclude that $|J(K(x))|$ is logarithmically small compared to $|J(x)|$ or $|J(xx')|$, since J may shrink x or xx' much more than it shrinks $K(x)$. Since J shrinks only very few strings, the stronger form of the theorem may well be true, but we have not checked this since it does not seem to be worth the additional effort.

8 Matrix Correspondence Problem

In this section, a matrix is a unimodular matrix, a column is a column of two relatively prime (not necessarily positive) integers, and a matrix is seen as the pair of its columns. Call a matrix or a column positive (resp. negative) if all its entries are non-negative (resp. non-positive). If u is a column then u_1, u_2 are the upper and the lower entries of u, and $|u|$ is the positive column v such that $v_i = |u_i|$. Positive columns are ordered componentwise, as in Section 1. If u is a column then $\max(u) = \max(|u_1|, |u_2|)$. Any component of a column u with the absolute value $\max(u)$ is *major*, and the other component is *minor*. If X is a matrix (u, v) then $\max(X) = \max(\max(u), \max(v))$. Any entry of a matrix X with the absolute value $\max(X)$ is the *major* entry. If u, v are the two columns of a matrix X and $|u| > |v|$ then u is the *major* column and v is the *minor*; in the case of the unit matrix, both columns are *major* and both are *minor*.

Lemma 8.1 *For every matrix $X = (u, v)$,*

1. *It is impossible that one of the numbers u_1v_2, u_2v_1 is positive and the other is negative. If they are both positive then $|u_1v_2| - |u_2v_1| = 1$, and if they are both negative then $|u_2v_1| - |u_1v_2| = 1$.*

2. *If X is not one of the following four matrices*

$$\begin{pmatrix} 1 & 0 \\ 0 & 1 \end{pmatrix}, \begin{pmatrix} -1 & 0 \\ 1 & -1 \end{pmatrix}, \begin{pmatrix} 0 & 1 \\ -1 & 0 \end{pmatrix}, \begin{pmatrix} 0 & -1 \\ 1 & 0 \end{pmatrix}$$

   then either $(|u|) > (|v|)$ or $(|u|) < (|v|)$.

Proof (1) If one of the numbers u_1v_2, u_2v_1 is positive and the other is negative then $|u_1v_2 - u_2v_1| \geq 1 + 1$ which is impossible. If the two numbers are positive then $|u_1v_2| - |u_2v_1| = u_1v_2 - u_2v_1 = 1$, and if the two numbers are negative then $|u_2v_1| - |u_1v_2| = u_1v_2 - u_2v_1 = 1$.

(2) Suppose that X is not one of the four matrices, and suppose that neither $|u| > |v|$ nor $|v| > |u|$. Without loss of generality, $|u_1| > |v_1|$ and $|u_2| < |v_2|$; otherwise

replace (u, v) with the matrix $(v, -u)$. Since X is neither the unit matrix nor its negative, either u_2 or v_1 is not zero. Hence $|u_1v_2|-|u_2v_1| \geq (|v_1|+1)(|u_2|+1)-|u_2v_1| = |u_2| + |v_1| + 1 \geq 2$ which contradicts (1). □

Lemma 8.2 *If* $\max(u, v) > 1$ *then* (u, v) *has only one major entry.*

Proof The proof is very similar to the proof of the second part of Lemma 6.1. There are some minor differences however, and – for the reader's convenience – we present the proof. By contradiction suppose that $m = \max(u, v) > 1$ but (u, v) has two or more major entries. If two major entries occur in the same row or column then m divides the determinant which is impossible. Thus, there are exactly two major entries. Since (u, v) may be replaced with $(v, -u)$, we may suppose that the two major entries form the second diagonal, i.e., $|u_2| = |v_1| = m$. If $u_2v_1 > 0$ then the determinant is negative which is impossible. Hence $u_2v_1 < 0$. By Lemma 8.1, $u_1v_2 \leq 0$ and $1 = |u_2v_1| - |u_1v_2| \geq m^2 - (m-1)^2 = 2m - 1 \geq 3$ which is false. □

Lemma 8.3 *For every two matrices* (u, v) *and* (u, v')*, there exists an integer* k*, such that* $v' = v + ku$*.*

Proof $u_1v_2' - u_2v_1' = 1 = u_1v_2 - u_2v_1$ and therefore $u_1(v_2' - v_2) = u_2(v_1' - v_1) = u_1u_2k$ for some k. If neither component of u is zero, the claim is obvious. Suppose that one of components of u is zero. By symmetry, let $u_1 = 0$. Then $v_1' = v_1$, $|u_2| = 1$ and the claim is clear. □

Lemma 8.4 *Let* $X = (u, v)$ *be any matrix with* $\max(X) > 1$*. If* u *(resp.* v*) is the major column of* X *then there exists exactly one additional matrix of the form* (u, v') *(resp.* (u', v)*) where the column* v' *(resp.* u'*) is minor. Moreover,* $v' = v \pm u$ *(resp.* $u' = u \pm v$*). If the major column is positive or negative then one of the two possible minor columns is positive and the other one is negative.*

Proof It suffices to consider the case when u is the major column because if (u, v) is a counterexample with the major column on the right then $(-v, u)$ is a counterexample with the major column on the left. Further, it suffices to consider the case when the major entry is positive because if (u, v) is a counterexample with a negative major entry then $(-u, -v)$ is a counterexample with a positive major entry. Let u_i be the major entry of u and (u, v') be another matrix with major column u. By Lemma 8.3, $v' = v + ku$ for some k. Since $u_i > 1$, $v_i \neq 0$. If $v_i > 0$ then $k = -1$, and if $v_i < 0$ then $k = 1$. Notice also that if $v_i > 0$ (resp. $v_i < 0$) then indeed u is the major column of the matrix $(u, v - u)$ (resp. $(u, v + u)$). Now suppose that u is positive. Obviously, $u_1 > 0$ and $u_2 > 0$. By part 1 of Lemma 6.1, v is either negative or positive. If v is positive (resp. negative) then v' is negative (resp. positive). □

Let $\mathrm{SL}_2(Z)$ denote not only the modular group but also the uniform domain of unimodular matrices with $|X| = \ell(\max(X))$.

Lemma 8.5 *Let $m > 1$ and X be a random unimodular matrix with $\max(X) = m$. The probability that X is positive is 1/8, and the probability that X is the inverse of a positive matrix is 1/8 as well.*

Proof Let S_0 be the collection of matrices X with $\max(X) = m$. The inverse of a matrix $\begin{pmatrix} a & c \\ b & d \end{pmatrix}$ is the matrix $\begin{pmatrix} d & -c \\ -b & a \end{pmatrix}$; thus $\max(X^{-1}) = \max(X)$ and therefore S_0 is closed under inversion. It follows that the number of positive matrices in S_0 equals the number of the inverses of positive matrices. Hence it suffices to prove only the first statement of the lemma.

Let S_1 be the collection of S_0 matrices X such that the major entry of X is positive. For every (u, v) in S_0, exactly one of the two matrices (u, v), $(-u, -v)$ belongs to S_1. It remains to prove that the probability of a random S_1 matrix being positive is 1/4.

Since the major entry of an S_1 matrix exceeds 1, the minor component of the major column is not zero. Let S_2 be the collection of S_1 matrices such that the minor component of the major column is positive. For every S_1 matrix X, let X' be the result of multiplying by -1 the diagonal of X which contains the minor component of the major column. Exactly one of the two matrices X, X' belongs to S_2. It follows that S_2 contains exactly one half of the elements of S_1. It remains to prove that the probability of a random S_2 matrix being positive is 1/2. Now use Lemma 8.4. □

The direct product $\mathrm{SL}_2(Z) \times \mathrm{SL}_2(Z)$ is a domain and monoid of matrix pairs; the multiplication of matrix pairs is componentwise: $(X_1, Y_1) \times (X_2, Y_2) = (X_1X_2, Y_1Y_2)$. If S is a set of matrix pairs, let S^n comprise products $P_1 \times \ldots \times P_m$ where $m \leq n$ and each $P_i \in S$. Let σ be a positive integer. In the following definition, MC stands for Matrix Correspondence.

Definition 8.6 For each positive integer σ, $\mathrm{MC}(\sigma)$ is the decision problem with domain $\mathrm{SL}_2(Z) \times \mathrm{Set}_\sigma(\mathrm{SL}_2(Z) \times \mathrm{SL}_2(Z)) \times \mathrm{PI}$ where an instance (A, S, n) is positive if and only if there exists a pair $(X, Y) \in S^n$ such that $AX = Y$.

Let σ witness Theorem 7.5.

Theorem 8.7 *MC(σ) is hard for RNP. Moreover, so is its restriction to the subdomain of those instances (A, S, n) where each pair in S consists of positive matrices.*

Proof The identity function reduces $\mathrm{PMC}(\sigma)$ to the desired restriction of $\mathrm{MC}(\sigma)$ and therefore to $\mathrm{MC}(\sigma)$ itself. To check the domination property, use Lemmas 3.7 and 8.5. □

9 Linear transformations of the modular group

Theorem 9.1 *Suppose that $T : SL_2(Z) \to SL_2(Z)$ is linear in the sense that, if $X = \sum_{i=1}^{k} Y_i$ with X and all Y_i in $SL_2(Z)$, then $T(X) = \sum_{i=1}^{k} T(Y_i)$. Then there exist B and C in $SL_2(Z)$ such that either, for all $X \in SL_2(Z)$, $T(X) = BXC$ or, for all such X, $T(X) = BX^tC$ where the superscript t denotes transpose.*

Proof We first normalize T so that $T(I) = I$, where I is the identity matrix. If the given T does not fix I, then we consider T' given by $T'(X) = T(I)^{-1}T(X)$, and we observe that the hypotheses of the theorem about T imply the same hypotheses about T' and the conclusion about T' (with $C = B^{-1}$) implies the same conclusion about T. Thus we may as well work with T', which fixes I, instead of T. So, from now on, we assume that $T(I) = I$.

Notation $\mathrm{M}_2(Q)$ (resp. $\mathrm{M}_2(C)$) is the vector space of two-by-two matrices with rational (resp. complex) entries. As usual, $\mathrm{SL}_2(Q)$ (resp. $\mathrm{SL}_2(C)$) is the multiplicative group of two-by-two matrices X with rational (resp. complex) entries such that $\det(X) = 1$.

Our next goal is to show that the linearity hypothesis on T implies that T can be extended to a linear transformation (in the usual sense) on $\mathrm{M}_2(Q)$. Let $\mathcal{B}$ be the set of the following four matrices in $\mathrm{SL}_2(Z)$:

$$\begin{pmatrix} 1 & 0 \\ 0 & 1 \end{pmatrix} = I, \begin{pmatrix} 1 & 0 \\ 1 & 1 \end{pmatrix}, \begin{pmatrix} 1 & 1 \\ 0 & 1 \end{pmatrix}, \begin{pmatrix} 1 & 1 \\ -1 & 0 \end{pmatrix}.$$

It is easy to see that every matrix in the standard basis for $\mathrm{M}_2(Q)$

$$\begin{pmatrix} 1 & 0 \\ 0 & 0 \end{pmatrix}, \begin{pmatrix} 0 & 1 \\ 0 & 0 \end{pmatrix}, \begin{pmatrix} 0 & 0 \\ 1 & 0 \end{pmatrix}, \begin{pmatrix} 0 & 0 \\ 0 & 1 \end{pmatrix}$$

is a linear combination (with integer coefficients) of the $\mathcal{B}$ matrices, so that $\mathcal{B}$ is a basis for $\mathrm{M}_2(Q)$. There is no question what the linear extension of T should be; it is the unique linear transformation $\bar{T}$ that agrees with T on these four basis matrices. Our task is to show that it agrees with T on the other matrices in $\mathrm{SL}_2(Z)$ as well. Any $\mathrm{SL}_2(Z)$ matrix X is a linear combination, with rational coefficients, of the four $\mathrm{SL}_2(Z)$ matrices in the basis; in order to show that $T(X) = \bar{T}(X)$, it suffices to prove that $T(X)$ is the similar linear combination of T-images of the four matrices. Thus, it suffices to show that any linear dependence relation with rational coefficients that holds between some matrices in $\mathrm{SL}_2(Z)$ also holds between their T-images. Furthermore, we may suppose that the coefficients are integers (since we can multiply by a common denominator of the rational coefficients) and in fact that the coefficients are all ± 1 (as other coefficients can be replaced by repeated terms). Comparing what we need to prove with the hypothesis of the theorem, we find that we need only check

that $T(-X) = -T(X)$ for all $X \in \mathrm{SL}_2(Z)$. But this is easy; just apply the hypothesis to the linear relation $X = -X + X + X$.

From now on, we write T not only for the given function but also for the corresponding linear transformation of $\mathrm{M}_2(Q)$ (called $\bar{T}$ above), and for the unique extension of this to a linear transformation of $\mathrm{M}_2(C)$.

We note for future reference that if a matrix X from $\mathrm{M}_2(C)$ has integer entries then so does $T(X)$. Indeed, this claim is true by hypothesis if X has determinant 1 and in particular for the four $\mathcal{B}$ matrices. By linearity of T, the claim follows for any X that is a linear combination with integer coefficients of $\mathcal{B}$ matrices. In particular the claim is true for the 4 matrices in the standard basis of $\mathrm{M}_2(Q)$ and therefore it is true for all X in $\mathrm{M}_2(C)$.

We shall also need that T preserves determinants, i.e. that $\det T(X) = \det(X)$ for all $X \in \mathrm{M}_2(C)$. This is true by hypothesis if $X \in \mathrm{SL}_2(Z)$, but some work will be needed to extend it to more general X.

Begin by considering $X \in \mathrm{M}_2(Q)$. For such an X, the following two conditions are equivalent. (1) The determinant and the trace of X both vanish. (2) There are at least two distinct non-zero rational numbers r for which $I + rX \in \mathrm{SL}_2(Z)$. This follows from the formula

$$\det(I + rX) = 1 + r \cdot \mathrm{tr}(X) + r^2 \cdot \det(X).$$

If (1) holds, then $I + rX$ has determinant 1 for all r, so we can satisfy (2) by taking any two r's for which the entries of rX are integers. Conversely, if (2) holds then we have two linear equations satisfied by the determinant and the trace of X, namely

$$r \cdot \mathrm{tr}(X) + r^2 \cdot \det(X) = 0$$

for each of the two r's. As the two equations are linearly independent, (1) follows.

It is clear, from inspection of condition (2), that if X satisfies it then so does $T(X)$. Thus, T maps the set N of matrices satisfying (2) or equivalently (1) into itself. T therefore also maps the linear span $\bar{N}$ of N into itself. Notice that the trace of every $\bar{N}$ matrix is zero and that matrices with zero trace form a 3-dimensional subspace of $\mathrm{M}_2(Q)$. Since N contains the matrices

$$\begin{pmatrix} 0 & 1 \\ 0 & 0 \end{pmatrix}, \begin{pmatrix} 0 & 0 \\ 1 & 0 \end{pmatrix}, \begin{pmatrix} 1 & 1 \\ -1 & -1 \end{pmatrix},$$

it follows that $\bar{N}$ is exactly that 3-dimensional subspace. In this 3-dimensional vector space $\bar{N}$, N is a cone, the zero-set of the quadratic form det.

We check that, for some k, $\det(T(X)) = k \cdot \det(X)$ for all X of trace zero. Let X be

$$\begin{pmatrix} x & y \\ z & -x \end{pmatrix},$$

so that $\det(X) = -x^2 - yz$. As T is linear and det is quadratic,

$$\det(TX) = \alpha x^2 + \beta y^2 + \gamma z^2 + \delta xy + \varepsilon xz + \zeta yz$$

for some coefficients $\alpha, \beta, \gamma, \delta, \varepsilon, \zeta$. Since T maps N onto itself, $\det(TX) = 0$ if $\det(X) = 0$. Choose $y = x^2$ and $z = -1$ so that $\det(X) = 0$ and therefore $\det(TX) = 0$. We have that, for all x, $\alpha x^2 + \beta x^4 + \gamma + \delta x^3 - \varepsilon x - \zeta x^2 = 0$, so that $\alpha = \zeta$ and $\beta = \gamma = \delta = \varepsilon = 0$. Thus, $\det(TX) = \alpha x^2 + \alpha yz = -\alpha \det(X)$.

Consider in particular the matrices

$$X = \begin{pmatrix} 0 & 3 \\ 1 & 0 \end{pmatrix} \qquad 2I + X = \begin{pmatrix} 2 & 3 \\ 1 & 2 \end{pmatrix},$$

and notice that X has trace zero while $2I + X \in \mathrm{SL}_2(Z)$. So we have

$$\begin{aligned} 1 &= \det(T(2I + X)) \\ &= \det(2I + T(X)) \\ &= 4 + 2 \cdot \mathrm{tr}(T(X)) + \det(T(X)) \\ &= 4 + 2 \cdot 0 + k \cdot \det(X) \\ &= 4 - 3k, \end{aligned}$$

so $k = 1$ and $\det(T(X)) = \det(X)$ for all rational X of trace zero.

For rational X of arbitrary trace, we can write $X = rI + Y$ where r is rational and $Y \in \bar{N}$. Then by what we have already proved, $T(Y)$ has trace zero and the same determinant as Y. So

$$\begin{aligned} \det(T(X)) &= \det(T(rI + Y)) \\ &= r^2 + r \cdot \mathrm{tr}(T(Y)) + \det(T(Y)) \\ &= r^2 + \det(Y) \\ &= \det(rI + Y) = \det(X). \end{aligned}$$

This shows that T preserves determinants of rational matrices. It follows that it preserves determinants of real matrices (by continuity) and of complex matrices (by analytic continuation). (Here is a more elementary argument. We have the equation $\det(T(X)) = \det(X)$ when all four entries are rational. If we let one entry, say the upper left one, vary over complex numbers while the other three entries remain fixed rational numbers, then the equation remains true because a polynomial equation in one variable that holds infinitely often must hold identically. Then we let another entry vary, while the remaining three stay fixed, one being an arbitrary complex number and the other two rational. Repeating the process for each entry in turn, we find that the equation holds for all complex values of the entries.)

Summarizing what we have achieved so far, we have a linear, determinant-preserving transformation T on $\mathrm{M}_2(C)$, which sends I to itself and sends integer

matrices to integer matrices. Our immediate goal is to show that there is a matrix $B \in \mathrm{SL}_2(C)$ satisfying the conclusion of the theorem with $C = B^{-1}$ (as $T(I) = I$), not only for all $X \in \mathrm{SL}_2(Z)$ but for all $X \in \mathrm{M}_2(C)$. (This information is essentially contained in [W, pages 19–21], but for the reader's convenience we give a different, more detailed proof.) Once this is done, we shall complete the proof by showing that the entries of B must be integers.

Until we reach our intermediate goal, we shall be working in the complex vector space $\mathrm{M}_2(C)$, and it will be convenient to use the following basis for this space:

$$I = \begin{pmatrix} 1 & 0 \\ 0 & 1 \end{pmatrix}, \quad P = \begin{pmatrix} i & 0 \\ 0 & -i \end{pmatrix}, \quad Q = \begin{pmatrix} 0 & 1 \\ -1 & 0 \end{pmatrix}, \quad R = \begin{pmatrix} 0 & i \\ i & 0 \end{pmatrix}.$$

The advantage of this basis is that the determinant is given by a very simple formula

$$\det(aI + pP + qQ + rR) = \det \begin{pmatrix} a + pi & q + ri \\ -q + ri & a - pi \end{pmatrix} = a^2 + p^2 + q^2 + r^2.$$

As T is linear and preserves determinants and I, it preserves eigenvalues; indeed, if $X - xI$ has determinant zero then so does $T(X - xI) = T(X) - xI$. In particular, $T \begin{pmatrix} 1 & 0 \\ 0 & 0 \end{pmatrix}$ has eigenvalues 1 and 0; viewed as a transformation of 2-component vectors, it is a projection onto some line along some other line, which means that it has the form

$$T \begin{pmatrix} 1 & 0 \\ 0 & 0 \end{pmatrix} = \begin{pmatrix} p \\ q \end{pmatrix} (r \quad s) = \begin{pmatrix} pr & ps \\ qr & qs \end{pmatrix}$$

for some p, q, r, s. Furthermore, since the eigenvalues are 1 and 0, the trace is 1, so $pr + qs = 1$. This means that the matrix

$$A = \begin{pmatrix} p & -s \\ q & r \end{pmatrix}$$

has determinant 1. The transformation $X \mapsto AXA^{-1}$ sends I to itself and sends $\begin{pmatrix} 1 & 0 \\ 0 & 0 \end{pmatrix}$ to $\begin{pmatrix} pr & ps \\ qr & qs \end{pmatrix}$, just like T. So the linear transformation

$$T'(X) = A^{-1}T(X)A$$

preserves determinants and fixes both I and $\begin{pmatrix} 1 & 0 \\ 0 & 0 \end{pmatrix}$ and therefore also their linear combination P. If we achieve our intermediate goal for T', the same result will follow immediately for T. Indeed, if $T'(X) = BXB^{-1}$ then $T(X) = (AB)X(AB)^{-1}$, while if $T'(X) = BX^tB^{-1}$ then $T(X) = (AB)X^t(AB)^{-1}$. So we may work with T' instead of T.

Thus, we assume that T fixes both I and P. Furthermore, as T preserves the quadratic form det, it also preserves the associated bilinear form

$$\langle X, Y \rangle = \frac{1}{2}(\det(X + Y) - \det(X) - \det(Y)),$$

which has, relative to our chosen basis, the standard form

$$\langle aI + pP + qQ + rR, a'I + p'P + q'Q + r'R\rangle = aa' + pp' + qq' + rr'.$$

(Usually, complex linear spaces are equipped with inner products that are linear in one factor and conjugate-linear in the other. That is not the case here; our inner product is linear in both factors.)

As T preserves this inner product and fixes I and P, it must leave invariant the set of vectors orthogonal to both I and P, namely the linear span of Q and R. So we have $T(Q) = qQ + rR$ for some scalars q and r. Also, we have

$$1 = \det(Q) = \det(T(Q)) = \det(qQ + rR) = q^2 + r^2.$$

There is a complex number v such that $(v + (1/v))/2 = q$. (Just solve a quadratic equation for v; of course there is a second solution $1/v$.) Notice that

$$[\frac{1}{2}(v + \frac{1}{v})]^2 + [\frac{1}{2i}(v - \frac{1}{v})]^2 = 1 = q^2 + r^2,$$

so $r = \pm(v - (1/v))/2i$. Replacing v with $1/v$ if necessary, we can arrange that

$$q = \frac{1}{2}(v + \frac{1}{v}) \qquad r = \frac{1}{2i}(v - \frac{1}{v}).$$

Let u be either of the square roots of v, and let

$$M = \begin{pmatrix} u & 0 \\ 0 & \frac{1}{u} \end{pmatrix}.$$

Notice that

$$M \cdot \begin{pmatrix} x & y \\ z & w \end{pmatrix} \cdot M^{-1} = \begin{pmatrix} x & u^2 y \\ \frac{1}{u^2} z & w \end{pmatrix} = \begin{pmatrix} x & vy \\ \frac{1}{v} z & w \end{pmatrix}.$$

In particular, the transformation $X \mapsto MXM^{-1}$ fixes I and P (just as T does) and it sends $Q = \begin{pmatrix} 0 & 1 \\ -1 & 0 \end{pmatrix}$ to

$$\begin{aligned} \begin{pmatrix} 0 & v \\ -\frac{1}{v} & 0 \end{pmatrix} &= \frac{1}{2}(v + \frac{1}{v}) \begin{pmatrix} 0 & 1 \\ -1 & 0 \end{pmatrix} + \frac{1}{2i}(v - \frac{1}{v}) \begin{pmatrix} 0 & i \\ i & 0 \end{pmatrix} \\ &= qQ + rR \\ &= T(Q). \end{aligned}$$

So $T'(X) = M^{-1}T(X)M$ preserves determinants and fixes I, P, and Q. As before, it suffices to reach our intermediate goal for T' rather than T. So from now on we assume that T fixes I, P, and Q.

It follows that T fixes the subspace orthogonal to I, P, and Q, namely the subspace spanned by R. So $T(R)$ is a scalar times R, and the scalar can only be ± 1 because T

preserves determinants. If the scalar is 1, then T fixes all four basis matrices, hence is the identity. If, on the other hand, the scalar is -1, then $T(X) = P^{-1}X^tP$, because the right side of this equation defines a linear transformation which, like T fixes I, P, and Q and reverses the sign of R. In either case, T has the required form, so we have achieved our intermediate goal.

We now return to the original T, normalized to fix I and extended to $\mathrm{M}_2(C)$, which we now know to have the form $T(X) = BXB^{-1}$ or $T(X) = BX^tB^{-1}$ for some $B \in \mathrm{SL}_2(C)$. We also know that if the entries of X are integers then so are those of $T(X)$. What we still need to show is that the entries of B are integers. Without loss of generality, we may suppose that $T(X) = BXB^{-1}$.

Let X be the matrix with a single entry equal to 1, say the entry in position i, j, and all other entries zero. Then the entries of $T(X)$, namely

$$(BXB^{-1})_{k,l} = B_{k,i}(B^{-1})_{j,l}$$

are integers for all k and l. But, as B has determinant 1, the entries of B^{-1} are the same as those of B, except for their signs and their positions in the matrices. Thus we see that the product of any two entries of B is an integer.

In particular, the square of each entry of B is an integer, so each entry is the product of an integer and (possibly) the square roots of certain distinct primes.

Suppose p is a prime whose square root occurs in one of the entries. Then $\sqrt{p}$ must occur in every entry, for the product of an entry containing $\sqrt{p}$ as a factor and another entry not containing it could never be an integer. So $\sqrt{p}$ occurs as a factor of every entry of B. But then p is a factor of $\det(B) = 1$. This contradiction shows that no square roots occur.

So every entry of B is an integer, and the proof is complete. □

We saw that an arbitrary linear transformation T over $\mathrm{SL}_2(Z)$ extends uniquely to a linear transformation over the vector space $M_2(R)$ of all four-by-four real matrices. Let $\mathrm{Mat}(T)$ be the matrix of (the extension of) T in the standard basis:

$$\begin{pmatrix} 1 & 0 \\ 0 & 0 \end{pmatrix}, \begin{pmatrix} 0 & 1 \\ 0 & 0 \end{pmatrix}, \begin{pmatrix} 0 & 0 \\ 1 & 0 \end{pmatrix}, \begin{pmatrix} 0 & 0 \\ 0 & 1 \end{pmatrix}.$$

Lemma 9.2

$$T\begin{pmatrix} x_1 & x_2 \\ x_3 & x_4 \end{pmatrix} = \begin{pmatrix} y_1 & y_2 \\ y_3 & y_4 \end{pmatrix} \longleftrightarrow Mat(T)\begin{pmatrix} x_1 \\ x_2 \\ x_3 \\ x_4 \end{pmatrix} = \begin{pmatrix} y_1 \\ y_2 \\ y_3 \\ y_4 \end{pmatrix}.$$

Proof Obvious. □

The entries of $\mathrm{Mat}(T)$ are the entries of the 4 matrices obtained by applying T to the standard basis. By Theorem 9.1, these are integers.

By Theorem 9.1, there are unimodular matrices B and C such $T(X) = BXC$ for all unimodular matrices X or $T(X) = BX^tC$ for all unimodular matrices X. Call T *right* in the first case and *left* in the second. The determinant of $\mathrm{Mat}(T)$ is ± 1 because

$$\det(\mathrm{Mat}(T)) \times \det(\mathrm{Mat}(T^{-1})) = \det(T \times T^{-1}) = 1.$$

Lemma 9.3 *Right transformations are exactly those with determinant* $+1$.

Proof It suffices to prove that if T is right then $\det \mathrm{Mat}(T) = 1$, because a left T is the product of the transposing transformation (whose matrix has determinant 1) and a right transformation. So suppose that $T(X) = BXC$ for some unimodular matrices B, C and all unimodular (and therefore all two-by-two real matrices) X.

Now forget about unimodular matrices and think about real matrices. Every pair (B', C') of nonsingular two-by-two real matrices gives a linear transformation $T' = X \mapsto B'XC'$ over two-by-two real matrices which has an inverse, so that the matrix $\mathrm{Mat}(T')$ of T' in the standard bases is non-zero. Now continuously transform B and C to the unit matrix. In the process T is continuously transformed to the identity, whose matrix has determinant 1, and the determinant of T remains non-zero all the time. Therefore its initial value cannot be -1. □

Lemma 9.4 *There is a PTime algorithm that, given a four-by-four integer matrix* M*, determines whether* $M = Mat(T)$ *for some* T.

Proof If $M = \mathrm{Mat}(T)$ then, by Theorem 9.1, there exist unimodular matrices B and C such that either $T(X) = BXC$ for all unimodular matrices X or $T(X) = BX^tC$ for all unimodular matrices X. We show how to find out whether there is a pair (B, C) satisfying the first condition and even how to find all pairs (B, C) satisfying the first condition. The case of the second condition is similar.

Suppose that $T(X) = BXC$ for all unimodular matrices X. Due to the unique extendibility of T, $T(X) = BXC$ for all two-by-two real matrices. Let

$$B = \begin{pmatrix} b_1 & b_2 \\ b_3 & b_4 \end{pmatrix}, C = \begin{pmatrix} c_1 & c_2 \\ c_3 & c_4 \end{pmatrix}.$$

Computing the products BXC where X belongs to the standard basis and using Lemma 9.2, we have

$$M = \begin{pmatrix} b_1c_1 & b_1c_3 & b_2c_1 & b_2c_3 \\ b_1c_2 & b_1c_4 & b_2c_2 & b_2c_4 \\ b_3c_1 & b_3c_3 & b_4c_1 & b_4c_3 \\ b_3c_2 & b_3c_4 & b_4c_2 & b_4c_4 \end{pmatrix}.$$

Thus we can recover all b_i/b_j and all c_i/c_j. So we recover B and C up to scalar factor. We recover the scalar factor, except for the sign, using the equalities $\det(B) = \det(C) = 1$. It follows that the pair (B, C) is unique except for an over-all sign. □

10 Matrix Transformation

In this section we prove that Matrix Transformation is hard for RNP.

For an arbitrary numerical matrix X, let $\max(X)$ be the maximal absolute value of the entries of X. Recall that $\ell(n)$ is the length of the binary notation for n and $\mathrm{Mat}(T)$ is the matrix of a linear transformation T.

Definition 10.1 *LT (standing for "Linear Transformations") is the uniform domain of linear transformations of $SL_2(Z)$ with $|T| = \ell(\max(Mat(T)))$.*

It will be convenient to ignore distinction between a linear transformation T over $\mathrm{SL}_2(Z)$ and its matrix $\mathrm{Mat}(T)$.

For all unimodular matrices B and C, let $T_{B,C}$ be the linear transformation $X \mapsto C^{-1}XB$.

Lemma 10.2 *The function $(B, C) \mapsto M_{B,C}$ reduces the subdomain of positive pairs in $SL_2(Z) \times SL_2(Z)$ to LT.*

Proof We need to check only that the function $f(B, C) = M_{B,C}$ has the domination property. Recall that the inverse of a matrix $\begin{pmatrix} a & c \\ b & d \end{pmatrix}$ is the matrix $\begin{pmatrix} d & -c \\ -b & a \end{pmatrix}$. It follows that $\max(f(B, C)) = \max(B) \times \max(C)$ and therefore $|B| + |C| - 1 \leq |f(B, C)| \leq |B| + |C|$.

Let $l = |f(B, C)|$, let M range over LT and let B, C range over unimodular matrices. Using # as the cardinality symbol, we have

$$\#\{M : |M| = l\} \leq 2\#\{(B, C) : l \leq |B| + |C| \leq l + 1\} = \sum_{j=1}^{l+1} [\#\{B : |B| = j\} \times \#\{C : l - j \leq |C| \leq l - j + 1\}] .$$

By Lemma 8.5, the number of unimodular matrices of size $m > 1$ is 8 times the number of positive unimodular matrices of size m. According to Lemma 6.7, the later number is $\Theta(2^{2m})$. It follows that, modulo a constant factor,

$$\#\{M : |M| = l\} \leq \sum_{l=1}^{l+1} 2^{2j} 2^{2l-2j} = 2^{2l}(l + 1).$$

We saw in the previous section that, for each M in LT, the pre-image of $f^{-1}(M)$ has at most 2 elements. It follows that, modulo bounded factors,

$$\frac{\mathbf{P}(f^{-1}f(B, C))}{\mathbf{P}(f(B, C))} \leq \frac{2^{2l}(l + 1)}{2^{2l}} = l + 1 \leq |(B, C)| + 1$$

which is AP on $\mathrm{SL}_2(Z) \times \mathrm{SL}_2(Z)$. □

If S is a subset of LT, let S^n be the set of products $T_m \cdots T_1$ where $m \leq n$ and each $T_i \in S$. First we prove that an auxiliary version of Matrix Transformation is hard for RNP.

Definition 10.3 For each positive integer σ, MT(σ) is the decision problem with domain $\mathrm{SL}_2(Z) \times \mathrm{Set}_\sigma(\mathrm{LT}) \times \mathrm{PI}$ where an instance (A, S, n) is positive if and only if there exists $T \in S^n$ that transforms A to the unit matrix.

Theorem 10.4 *Some MC(σ) is hard for RNP.*

Proof Let σ witness Theorem 8.7. We reduce the subdomain of MC(σ) described in Theorem 8.7 to MC(σ). If S is a sequence of positive matrix pairs, let S' be the result of replacing each pair (B, C) in S with the linear transformation $T_{B,C}(X) = C^{-1}XB$. The desired reduction is $f(A, S, n) = (A, S', n)$. To check the correctness property, note that $AB_1 \dots B_m = C_1 \dots C_m$ if and only if the transformation $T_{B_m,C_m} \dots T_{B_1,C_1}$ takes A to the unit matrix.

It remains to check that f reduces the relevant subdomain of $\mathrm{SL}_2(Z) \times (\mathrm{SL}_2(Z) \times \mathrm{SL}_2(Z))^\sigma \times \mathrm{PI}$ to the domain $\mathrm{SL}_2(Z) \times (\mathrm{LT})^\sigma \times \mathrm{PI}$. It suffices to check that the function $S \mapsto S'$ reduces $(\mathrm{SL}_2(Z) \times \mathrm{SL}_2(Z))^\sigma$ to LT^σ. By Lemma 3.7, it suffices to check that the function $(B, C) \mapsto T_{B,C}$ reduces $\mathrm{SL}_2(Z) \times \mathrm{SL}_2(Z)$ to LT. Now use Lemma 10.2. □

Definition 10.5 MT is the decision problem with domain $\mathrm{SL}_2(Z) \times \mathrm{Set}(\mathrm{LT}) \times \mathrm{PI}$ such that an instance (A, S, n) is positive if and only if there exists $P \in S^n$ with $A = P(1)$.

Corollary 10.6 *MT is RNP complete.*

Proof The identity function deterministically reduces MT(σ) to MT. We omit checking that MT is in RNP. □

Remark 4 Let $\pi(j)$ be any PTime computable nondecreasing function from positive integers to positive integers such that the inverse function $\pi^{-1}(j) = \min_i[\pi(i) \geq j]$ is polynomially bounded. For example, $\pi(j) = j$. The restriction of MT (resp. MT(σ)) to instances (A, S, n) with $n = \pi(|A|)$ remains RNP complete. The proof is the same proof except we start with the corresponding version of the bounded halting problem, which has been proved RNP complete in [Gu1, Section 9].

11 Bounded membership problem

In this section, we briefly consider a natural simplification of MT(σ) where the question is whether the given unimodular matrix X is a product of at most n factors taken from a given finite subset (assumed closed under inverses) of $\mathrm{SL}_2(Z)$. This is a bounded version of the membership problem [Mi] for $\mathrm{SL}_2(Z)$. We show that it is NP complete. It is interesting open problem whether a natural randomization of it is RNP complete. We begin with the analogous bounded version of the membership problem for the additive group of integers.

Definition 11.1 *Integer Sum* is the following NP problem:

Instance: A positive integer K, a finite set S of positive integers, and a positive integer n.

Question: Can K be represented as $\sum_{i=1}^{m} \varepsilon_i b_i$ where $m \leq n$, the numbers b_i are (not necessarily distinct) elements of S, and $\varepsilon_i \in \{1, -1\}$?

The restriction n on the number of summands is important. It is easy to decide whether or not K can be represented as a sum of elements of $S \cup \{-b : b \in S\}$; just compute the greatest common divisor of the elements of S and check whether it divides K.

Lemma 11.2 *Integer Sum is NP complete.*

The fact may be well known. It was not known to us. Suzanne Zeitman, a graduate student of the second author, proved the lemma.

Proof The proof is by reduction from X3C, Exact Cover by 3-Sets [GJ], which is the following NP problem:

Instance: A positive integer q and a collection C of 3-element subsets of the set $\{1, 2, \ldots, 3q\}$.

Question: Is there an exact cover $C' \subseteq C$ for X (so that each element of X belongs to exactly one member of C')?

The transformation f we use resembles that used in the reduction of 3-Dimensional Matching to Partition [GJ]. Given an instance (q, C) of X3C, let l be the length of the binary notation for q and B be the collection of binary strings of length $3ql$. View a B string as a sequence of $3q$ blocks (substrings) of length l. For each i, $1 \leq i \leq 3q$, let a_i be the integer represented by a B string with exactly one 1 which appears at the rightmost position of the i-th block.

Claim 11.3 *If* $\sum_{i=1}^{3q} \alpha_i a_i = \sum_{i=1}^{3q} \beta_i a_i$ *and* $0 \leq \alpha_i, \beta_i \leq q$ *for each* i*, then* $\alpha_i = \beta_i$ *for each* i*.*

Proof By the definition of l. □

Define

$$f(q, C) = (K, \{y(T) : T \in C\}, q)$$

where $K = \sum_{i=1}^{3q} a_i$ and each $y(T) = \sum_{i \in T} a_i$. If C' is an exact cover of C, then the cardinality of C' is q and K is represented as the sum of the q numbers $y(T)$ such that $T \in C'$.

Now suppose that $K = \sum_{j=1}^{m} \varepsilon_j y(T_j)$ where $m \leq q$ and $\varepsilon_j \in \{1, -1\}$ and $T_j \in C$. Then let $K^+ = \sum\{y(T_j) : \varepsilon_j > 0\} = \sum_{i=1}^{3q} \alpha_i a_i$ and $K^- = \sum\{y(T_j) : \varepsilon_j < 0\} = \sum_{i=1}^{3q} \gamma_i a_i$, so that $K^+ = K + K^-$. Clearly, $\alpha_i \leq m \leq q$. Similarly, $\gamma_i \leq q$. For each $T \in C$, let $z_i(T)$ equal 1 if $i \in T$ and equal 0 otherwise.

First consider the case $K^- = 0$. By the Claim, we have that, for each i, $\sum_j z_i(T_j) = 1$. Thus, the sets T_j form an exact cover.

By contradiction, suppose that $K^- > 0$. Then the number of j's with $\varepsilon_j > 0$ is less than q and therefore there exists an i with $\alpha_i = 0$. Since $K^+ = K + K^-$, we have, by the Claim, that $0 = \alpha_i = 1 + \gamma_i$ which is impossible. □

Definition 11.4 The *bounded membership problem* for the modular group, in short BM, is the following NP decision problem:

Instance: A unimodular matrix X, a finite set S of unimodular matrices and a positive integer n.

Question: Can X be represented as $\prod_{i=1}^{m} Y_i$ where $m \leq n$ and, for each i, either Y or Y^{-1} is in S?

Corollary 11.5 *BM is NP complete.*

Proof For each integer y, let

$$g(y) = \begin{pmatrix} 1 & y \\ 0 & 1 \end{pmatrix}.$$

If $g(y) = Y$ and $g(z) = Z$ then $g(y + z) = YZ$ and $g(-y) = Y^{-1}$. This gives rise to the following reduction of IS to BM:

$$F(K, S, n) = (g(K), \{g(y) : y \in S\}, n).$$

□

One natural way to randomize BM is to view the domain of BM as $\mathrm{SL}_2(Z) \times \mathrm{Set}(\mathrm{SL}_2(Z)) \times \mathrm{PI}$. The corresponding randomized decision problem is probably decidable in AP time.

References

[BCGL] Shai Ben-David, Benny Chor, Oded Goldreich and Michael Luby, "On the Theory of Average Case Complexity", 21st Annual ACM Symposium on Theory of Computing, ACM, 1989, 204–216.

[BG1] Andreas Blass and Yuri Gurevich, "On the Reduction Theory for Average-Case Complexity", CSL'90, 4th Workshop on Computer Science Logic, Heidelberg, Germany, Springer Lecture Notes in Computer Science 533, 1991, 17–30.

[BG2] Andreas Blass and Yuri Gurevich, "Randomizing Reductions of Search Problems", SIAM J. on Computing, to appear. An extended abstract in FST&-TCS'91, 11th Conference on Foundations of Software Technology and Theoretical Computer Science, New Delhi, India, Springer Lecture Notes in Computer Science 560 (1991), 10–24.

[BG3] Andreas Blass and Yuri Gurevich, "Randomized Reductions of Decision Problems" (tentative title), in preparation.

[Ei] Samuel Eilenberg, "Automata, Languages, and Machines", Vol. A and B, Academic Press, NY & London, 1974 and 1976, xvi+451pp. and xiii+387 pp.

[GJ] Michael R. Garey and David S. Johnson, "Computers and Intractability: A Guide to the Theory of NP-Completeness", Freeman, New York, 1979.

[Gu1] Yuri Gurevich, "Average Case Completeness", J. Computer and System Sciences 42:3, June 1991, 346–398. (An extended abstract in FOCS'87.)

[Gu2] Yuri Gurevich, "Matrix Decomposition Problem is Complete for the Average Case", FOCS'90, 31st Annual Symposium on Foundations of Computer Science, IEEE Computer Society Press, 1990, 802–811.

[Gu3] Yuri Gurevich, "Average Case Complexity", ICALP'91, 18th International Colloquium on Automata, Languages and Programming, Madrid, Springer Lecture Notes in Computer Science 510, 1991, 615–628.

[GS] Yuri Gurevich and Saharon Shelah, "Expected Computation Time for Hamiltonian Path Problem", SIAM J. on Computing 16:3 (1987) 486–502.

[HW] G. H. Hardy and E. M. Wright, "An Introduction to the Theory of Numbers", Oxford University Press, 5th edition, 1988 printing.

[IL] Russel Impagliazzo and Leonid A. Levin, "No Better Ways to Generate Hard NP Instances than Picking Uniformly at Random", 31st Annual Symposium on Foundations of Computer Science, IEEE Computer Society Press, 1990.

[Jo] David S. Johnson, "The NP-Completeness Column", Journal of Algorithms 5 (1984), 284-299.

[Kn1] Donald E. Knuth, "The Art of Computer Programming", Vol. 1, 2nd edition, Addison-Wesley, Reading, Massachusetts, 1973.

[Kn2] Donald E. Knuth, "Big Omicron and Big Omega and Big Theta", SIGACT News, Apr.–June, 1976, 18–24.

[Le] Leonid A. Levin, "Average Case Complete Problems", SIAM Journal of Computing, No. 15, 1986, 285–286.

[VL] Ramarathnam Venkatesan and Leonid Levin, " Random Instances of a Graph Coloring Problem are Hard", 20th Symp. on Theory of Computing, ACM, 1988, 217–222.

[VR] Ramarathnam Venkatesan and Sivaramakrishnan Rajagopalan, "Average Case Intractability of Matrix and Diophantine Problems", 24th Symp. on Theory of Computing, ACM, 1992, 632–642.

[Mi] Charles F. Miller, III, "Decision Problems in Algebraic Classes of Groups (a Survey)", in Word Problems, ed. by W.W. Boone, F.B. Cannonito, and R.C. Lyndon, North-Holland, 1973, 507–523.

[YK] Andrew C. Yao and Donald E. Knuth, "Analysis of the subtractive algorithm for greatest common divisors", Proc. Nat. Acad. Sci USA 72:12 (1975), 4720–4722.

[W] Bartel L. van der Waerden, "Gruppen von Linearen Transformationen", Ergebnisse Math., IV.2, Springer-Verlag, Berlin, 1935.