On sampling generating sets of finite groups

*Sergey Bratus**, *Igor Pak*†

Abstract. Let G be a finite group. For a given k, what is the probability that a group is generated by k of its random elements? How can one uniformly sample these generating k-tuples? In this paper we answer these questions for the class of nilpotent groups. Applications to product replacement algorithms and random random walks are discussed.

1. Introduction

Let G be a finite group. A sequence of group elements $(g_1, \ldots, g_k)$ is called a *generating k-tuple* of G if these elements generate G (we write $\langle g_1, \ldots, g_k \rangle = G$). Let $\mathcal{N}_k(G)$ be the set of all generating k-tuples of G, and let $N_k(G) = |\mathcal{N}_k(G)|$.

We consider two related problems on generating k-tuples. Given G and $k > 0$,

1) *Determine $N_k(G)$.*

2) *Generate a random element of $\mathcal{N}_k(G)$, so that each element of $\mathcal{N}_k(G)$ appears with probability $1/N_k(G)$.*

The problem of determining the structure of $\mathcal{N}_k(G)$ is of interest in several contexts. The counting problem goes back to Hall, who expressed $N_k(G)$ as a Möbius type summation of $N_k(H)$, taken over all maximal subgroups $H \subset G$ (see [33]). Recently, the counting problem has been studied for large simple groups, where remarkable progress has been made (see [39, 40, 41, 49]). In this paper we analyze $N_k(G)$ for nilpotent groups. We also show that $N_k(G)$ minimizes when $G \simeq \mathbb{Z}_2^r$, $r \geq \log_2 |G|$.

The sampling problem, while often used in theory as a tool for approximate counting, recently began a life of its own. In [14] Celler et al. proposed a *product replacement* Markov chain on $\mathcal{N}_k(G)$, which is believed to be rapidly mixing. The subject was further investigated in [8, 15, 21, 22, **?**]. We present an efficient and economical algorithm for sampling in case when G is nilpotent.

*Department of Mathematics, Northeastern University, Boston, MA 02115, E-mail: **sbratus@ccs.neu.edu**

†Department of Mathematics, Yale University, New Haven, CT 06520, E-mail: **pak@math.yale.edu**

The generating k-tuples also occur in connection with the so-called *random random walks*, which are ordinary random walks on G *with random generating sets*. The analysis of these "average case" random walks was inspired by Aldous and Diaconis in [1] and was continued in a number of papers (see e.g. [26, 52, 47, 55]). We will show that one can use the sampling problem to simulate these random random walks.

2. Definitions and main results

2.1. Counting problem Let G be a finite group. By $|G|$ denote the order of G. As in the introduction, let $N_k(G) = |\mathcal{N}_k(G)|$ be the number of generating k-tuples $\langle g_1, \ldots, g_k \rangle = G$. It is often convenient to consider the probability $\varphi_k(G)$ that k uniform independent group elements generate G :

$$\varphi_k(G) = \frac{N_k(G)}{|G|^k}$$

Theorem 1. *For any finite group G, $1 > \epsilon > 0$, we have*

$$\varphi_k(G) > 1 - \epsilon$$

given $k > \log_2 |G| + \max\{3, 2 \log_2 1/\epsilon\}$.

This is a slight improvement over a more general classical result by Erdős and Rényi in [28] (see also [27]).

Define $\varkappa(G)$ to be the smallest possible number of generators of G. In other words, let

$$\varkappa(G) = \min\{k \,|\, N_k(G) > 0\}.$$

The problem of evaluating $\varkappa(G)$ has been of intense interest for classes of groups as well as for individual groups (see [17]).

It is known that $\varkappa(G) = 2$ for all simple, nonabelian groups, and that $\varkappa(G) \leq n/2$ for $G \subset S_n$, with equality achieved when $G \simeq \mathbb{Z}_2^{n/2}$, and n is even. Also, it is easy to see that $\varkappa(G) \leq \log_2 |G|$, with equality for $G \simeq \mathbb{Z}_2^n$.

Let $\vartheta(G)$ be the smallest k such that at least $1/3$ of the random k-tuples $(g_1, \ldots, g_k)$ generate the whole group. In other words, let

$$\vartheta(G) = \min\left\{k \,|\, \varphi_k(G) > \frac{1}{3}\right\}.$$

Note that Theorem 1 immediately implies that

$$\vartheta(G) \leq \log_2 |G| + 3$$

By definition $\vartheta(G)/\varkappa(G) > 1$. It is unclear, however, how big this ratio can be (see [49, **?**]).

Here are few known results. When G is simple, it is known that $\varphi_2(G) \to 1$ as $|G| \to \infty$ (see [53]). This was the famous Kantor–Lubotzky conjecture, now a theorem. For $G = A_n$, this is a famous result of Dixon (see [25]). For classical simple groups of Lie type the result was confirmed by Kantor and Lubotzky (see [39]). In full generality it was recently proved by Liebeck and Shalev (see [40]). This immediately implies that $\vartheta(G) < C$ for any simple group G and some universal constant C. It was noted in [22] that when G is nilpotent, then $\varphi_{\varkappa+1}(G) > Const$. The following result is an improvement.

Theorem 2. *Let G be a finite nilpotent group. Then $\vartheta(G) \leq \varkappa(G) + 1$.*

We refer the reader to [23, 49, **?**] for further discussion.

2.2. On presentations of groups There are several ways a finite group G can be presented as input to an algorithm. Any group–theoretic algorithm needs to be able to perform the group operation, to find inverse elements and to compare the results of these operations with the identity element of G. The complexity of an algorithm is expressed as a function of the times necessary to perform these operations and other parameters. Thus, regardless of the presentation of G, denote by μ the time necessary for group operations (multiplication, taking an inverse, comparison with id[1]). Further, randomized algorithms usually assume the ability to generate random elements of the group G. Denote by ρ the complexity of generating a (nearly) uniform group element (call this the *random generation* subroutine). It is also convenient to denote by η the time required to check whether given k group elements generate the entire group. We call this task a *generation test*.

We start with *permutation groups*, which are defined as subgroups of a permutation group S_n. The group is presented by a set of generators. This is the best understood class of groups with efficient management, random elements generation, generation test, etc., based on the fundamental algorithms by C. Sims (see e.g. [54, 16, 43]. In particular one has $\rho = O(\mu n)$, and $\eta = O(\mu n^4)$ (one can show that in this case by reducing the problem to group membership).

A *matrix group* is a group defined as a subgroup of $GL(n; q)$. This is a harder class of groups to work with (see [37, 8]). Recently some important advances have been made in this setting (see [10, 13, 45, 42]). Still, polynomial time management for matrix groups is yet to be discovered.

One of the most general and widely accepted is the *black box* setting (see [8]), in which group elements are encoded by bit strings of a uniform fixed length n (possibly non-uniquely, i.e. several different strings may correspond to the same element of G). A *black box oracle* is given, that can multiply elements, take their inverses (returning the results as bit strings of the same

[1]For some presentations, such as the presentation by generators and relations, the latter task can be non-trivial. The black box model discussed below makes the assumption that the identity test, i.e. comparison with id, can be performed efficiently.

encoding), and compare elements with identity (see [8]), in time polynomial in n. Note that n gives a bound on $\log_2 |G|$. This presentation of a group generalizes both permutation groups and matrix groups. This setting proved itself to be useful for working with general finite groups about which we have limited information.

In his pioneering work [6], Babai was able to find a polynomial time algorithm for generating (nearly) uniform group elements. The product replacement algorithm of [14] was designed to give a *practical* algorithm for random generation. These algorithms were used in a number of subsequent works, particularly on recognition of various classes of finite groups (see [11, 12, 38, 45]). Following Babai (see [6]), there is no subexponential in n algorithm which can perform the general generation test. When necesary, we isolate the complexity of performing the generation test in a separate subroutine of complexity η.

Finally, a finite solvable group G can be given by a polycyclic generating sequence, also referred to as an *AG-system* (see [54], Sec. 9.4). In this case, both the random generation subroutine and the generation test can be easily performed in nearly linear time. While no subexponential algorithm for finding such a presentation is known, the existing algorithms implemented in GAP and MAGMA are known to be very efficient in practice. We will consider nilpotent groups that come in such a presentation.

2.3. Sampling problem Now consider the sampling problem (see introduction) from the computational point of view. We immediately obtain the following result.

Proposition 2.1. *Let G be a black box group with a generation test oracle, and a random generation oracle. Let ρ be the time required for the random generation oracle to generate a (nearly) uniform random element of G, and η be the time in which the generation test oracle can perform a generation test. Let $k \geq \vartheta(G)$. Then there exists a randomized algorithm for sampling from $\mathcal{N}_k(G)$ in time $O(\rho k + \eta)$.*

Indeed, given $k \geq \vartheta(G)$, we can always sample from $\mathcal{N}_k(G)$ by simply generating a uniform k-tuple and testing whether it generates the whole group G. We call this method *Choose–and–Check*. The sampling problem is open for $\varkappa(G) \leq k < \vartheta(G)$. We do not believe that there exists an efficient sampling algorithm for general black box groups. However, such algorithm does exist in special cases, when the group satisfies additional properties. In particular, below we present such an algorithm for a finite nilpotent group given by an AG-system.

Theorem 2.2. *Let G be a nilpotent group, given by an AG-system. $k \geq \vartheta(G)$. Then there exists a randomized algorithm for sampling from $\mathcal{N}_k(G)$ with running time $O(k\rho)$, and which requires $k \log_2 |G| (1 + o(1))$ random bits.*

By random bits we mean, roughly speaking, the number of coin flips required in the algorithm. Algorithms which require fewer random bits are considered to be better for applications. A formal definition will be given in section 4.8.

Observe that the number of random bits cannot be smaller that $\log_2 N_k(G)$. To demonstrate the strength of the algorithm in Theorem 6, consider the case $G = \mathbb{Z}_2^n$. Then $\varkappa = n$ and $\mathcal{N}_n(G)$ is in one-to-one correspondence with the set of nonsingular matrices $GL(n; 2)$. It is known that $\varphi_n(G) = c > 1/4$ (see e.g. [44, 46]). The standard approach to sample from $GL(n; 2)$ would be to sample a random matrix and then check by Gaussian elimination whether it is nonsingular. The expected number of random bits required for this is $\frac{1}{c}\lceil \log_2(n^2) \rceil$. On the other hand, our algorithm requires only $\log_2 n^2(1 + o(1))$ random bits. The problem of saving random bits when sampling from $GL(n; q)$ was considered earlier by Randall (see [51]) and the second author (see [46]). Thus Theorem 2.2 can be thought of as an advanced generalization of these results.

3. Applications

3.1. Product replacement algorithm This algorithm is an important recent advancement in computational group theory. In [14] Celler et al. defined a Markov chain X_t on $\mathcal{N}_k(G)$ as follows. Let $X_t = (g_1, \ldots, g_k) \in \mathcal{N}_k(G)$. Define $X_{t+1} = (g_1, \ldots, h_j, \ldots, g_k)$, where $h_j = g_j g_i^{\pm 1}$ or $h_j = g_i^{\pm 1} g_j$, where the pair (i, j), $1 \leq i, j \leq k$, $i \neq j$ is sampled uniformly; the multiplication order and the ± 1 degree are determined by independent flips of a fair coin. By $\widetilde{\varkappa}(G)$ denote the maximum size of the minimum generating set (i.e. of the set such that no generator can be omitted). The authors showed that when $k \geq 2\,\widetilde{\varkappa}$ this Markov chain is reversible, aperiodic and irreducible, and has a uniform stationary distribution. Thus the chain is ergodic and can be used for approximate sampling from $\mathcal{N}_k(G)$, $k > 2\widetilde{\varkappa}(G)$.

At the moment it is not known whether the Markov chain converges in time polynomial of k and $\log |G|$. The empirical tests seem to indicate that it mixes rapidly (see [14]). We refer the reader to the review article [**?**] by the second author for references and detailed discussion of the matter.

Let us point out that if one knows how to sample generating k-tuples, one can also test how close the product replacement Markov chain is to a stationary distribution. Indeed, one can simply compare any given statistics on $\mathcal{N}_k(G)$

on samples obtained by the *exact* sampling and on samples obtained by the product replacement algorithm. The authors in [14] use a chi-square statistic, while the latter checking method allows more freedom.

3.2. Random random walks Let G be a finite group, and let $(g_1, \ldots, g_k) \in \mathcal{N}_k(G)$ be a generating k-tuple. A *random walk* X_t on G is defined by $X_0 = id$, $X_{t+1} = X_t \cdot g_i$, where i is chosen uniformly in $[1, \ldots, k]$. One can think of the walk X_t as of a nearest neighbor random walk on a Cayley graph.

It is known that under minor conditions the random walk converges to a uniform distribution on G (see e.g. [2]). An important problem is to estimate how long will it take to converge to stationary. Formally, let $Q^t(g) = \mathbf{P}(X_t = g)$ be the probability distribution of the walk after t steps. Define the *separation distance* $s(t)$ as follows:

$$s(t) = |G| \max_{g \in G} \left(\frac{1}{|G|} - Q^t(g) \right)$$

(see [18, 2]). In other words, $s(t) = \max\{\epsilon \,|\, Q^t \text{ is } \epsilon - \text{uniform}\}$.

Usually estimating $s(t)$ is a hard problem, from both theoretical and computational points of view. Good estimates in cases of importance normally require a detailed knowledge of the behavior of a random walk. In [1] Aldous and Diaconis proposed to study "average case" random walks, and conjectured that they must be rapidly mixing. Such random walk with random support are often called *random random walks*.

A breakthrough was made by Dou and Hildebrand, who confirmed the conjecture for superlogarithmic values of k. Roughly speaking, they showed that after $t > C \frac{a}{a-1} \log_k |G|$ steps we have $E(s(t)) \to 0$ as $|G| \to \infty$, given $k > \log^a |G|$. Different proofs and better bounds in special cases, such as abelian groups, were obtained by subsequent investigators (see [4, 32, 35, 47, 50, 52, 55]). For fairly small k, such as $k = o(\log_2 |G|)$, the problem is largely unresolved (see [35, 47]). Say, for $G = S_n$ it is believed that $t = \Omega(n^3 \log n)$ implies $s(t) \to 0$ as $n \to \infty$ for *any* generating k-tuple, $k = C \geq 2$ (see above). However, no polynomial bound is known even for random random walks, the best one in existence being due Babai and Hetyei (see [9, 47]).

Now, given this poor state of the art for $k = o(\log_2 |G|)$, one may wish to collect experimental evidence about the behavior of random random walks. That is where one can apply the sampling procedures. Note also that in general, if we can compute $s(t)$ for random walks generated by random k-tuples, there is no need to check whether this is a generating k-tuple. Indeed, if a k-tuple does not generate G, the corresponding Cayley graph is disconnected and $s(t) = 1$ for all $t > 0$. Thus if $k > C\, \vartheta(G) \log(1/\epsilon)$, then $\varphi_k(G) > 1 - \epsilon$. Let $\epsilon \to 0$. We conclude that the expectation over all k-tuples $E(s(t)) \to 0$ if and only if so does the expectation taken over all generating k-tuples.

4. Sampling problem

Let us first consider the elementary 'building blocks' of finite abelian groups. Clearly, $\mathbb{Z}_p$ is generated by any one of its non-zero elements.

4.1. Generating k-tuples of $G = \mathbb{Z}_{p^m}$ We can think of elements of $\mathbb{Z}_{p^m}$ as integers $0, \ldots, p^m - 1$ written as numbers base p. Write for $g \in \mathbb{Z}_{p^m}$

$$g = a_{m-1}p^{m-1} + \cdots + a_1 p + a_0, \quad a_i \in \mathbb{Z}_p$$

and call a_i the 'coordinates' of g. Let us also write

$$g \bmod p = a_0.$$

for the last coordinate of g.

Then a k-tuple $(g_1, \ldots, g_k)$ generates $\mathbb{Z}_{p^m}$ if and only if at least one of g_i has a non-zero coordinate a_0. The rest of the coordinates can be chosen randomly. If we write our k-tuple in such coordinates as a matrix with k rows and m columns, then any matrix with at least one non-zero element in the last column will correspond to a generating k-tuple of $\mathbb{Z}_{p^m}$ and vice versa. This establishes a simple algorithm for sampling generating k-tuples of $\mathbb{Z}_{p^m}$.

4.2. Preparation for general abelian groups In order to produce generating k-tuples of more general abelian groups, we need the following two algorithms. The first algorithm will sample permutations from S_m and the second one $(k-m)$-subsets of the set $\{1, \ldots, k\}$, with certain specific distributions. Namely, we need to sample

- *a permutation $\sigma \in S_m$ with the probability proportional to $q^{-inv(\sigma)}$,*
- *a subset $\{i_1, \ldots, i_{k-m}\} \subset \{2, \ldots, k\}$, $1 < i_1 < i_2 < \ldots < i_{k-m} \leq k$, with the probability proportional to $q^{|I|}$,*

where q is a prime power, $inv(\sigma)$ is the number of inversions of $\sigma \in S_m$, and $|I| = \sum_{\alpha=1}^{k-m} i_\alpha$. More precisely, the probability of sampling $\sigma \in S_m$ is

$$P(\sigma, q) = \frac{q^{-inv(\sigma)}}{f_m(\frac{1}{q})},$$

where

$$f_m(t) = \sum_{\sigma \in S_m} t^{inv(\sigma)} = \prod_{i=1}^{m} (1 + t + \cdots + t^{i-1}).$$

Note that $f_m(q)$ is equal to the number of complete flags over $\mathbb{F}_q$ (see e.g. [46]).

The following algorithms for the two tasks above have been adopted from [46]. We will form a permutation by placing numbers $1, 2, \ldots, m$ on a line, one after another. First place 1, then place 2 either to the right or to the left of 1, choosing one of these two positions with certain probability, then place 3 in one the three possible intervals into which the previously placed numbers divide the line etc. Denote $(i)_x = (1+x+\cdots+x^{i-1})$, where $x > 0$. When placing i, place it in the right-most position with probability $1/(i)_{q^{-1}}$, in the second position from the right with the probability $q^{-1}/(i)_{q^{-1}}$, etc., in the left-most position with probability $q^{-i+1}/(i)_{q^{-1}}$. By the multiplicative property of $f_m(t)$ we will have the desired sampling distribution when this process stops (i.e. when m is placed into one of the m available intervals).

The second sampling problem is equivalent to sampling an $(n-m)$-subset of $\{1, \ldots, n\}$ with probability depending on the sum of the chosen $n-m$ numbers (in our actual sampling 1 will be prohibited from being chosen). It is, in turn, equivalent to sampling an $(n-m)$-subset with probability depending on the number of *inversions* in it, where we define the number of inversions $inv(A)$ in $A \subset \{1, \ldots, n\}$ as the number of pairs (i, j), $1 \le i < j \le n$, such that $i \in A$, $j \notin A$. It can be checked that

$$\sum_{\alpha=1}^{n-m} i_\alpha = \frac{1}{2}(n+m+1)(n-m) - inv(A).$$

The method of sampling a $(n-m)$-subset A out of $\{1, \ldots, n\}$ with the probability proportional to $q^{-inv(A)}$ is analogous to the one for permutations. For each number from 1 to n we will decide whether we want to include it in the subset A or not, i.e. include it (or not include it) with a certain fixed probability, depending only on the number itself and the number of elements chosen for A so far. Namely, suppose we are considering $i \in \{1, \ldots, n\}$ and have l elements to go before A is complete (i.e. contains $n-m$ elements). Then we include i in A with probability

$$\frac{1+\beta+\beta^2+\cdots+\beta^{l-1}}{1+\beta+\beta^2+\cdots+\beta^{n-i+1}},$$

where $\beta = 1/q$. We stop when $l = n-m$. This algorithm is discussed in detail in [46].

4.3. Generating uniform invertible matrices over $\mathbb{F}_q$ Consider $G = \mathbb{Z}_p \oplus \ldots \oplus \mathbb{Z}_p = (\mathbb{Z}_p)^{\oplus m}$. The group G is then an m-dimensional vector space over $\mathbb{Z}_p$, and we may think of its elements as m-dimensional vectors. Therefore no k-tuple can generate G for $k < m$.

For $k = m$, an m-tuple $(g_1, \ldots, g_m)$ generates G if and only if the matrix with the coordinates of g_j's as rows is non-singular. Generating a random non-singular $m \times m$ matrix isn't hard, since a matrix with randomly chosen

entries is, with high probability, non-singular. For a detailed discussion of this issue and an efficient method for generating uniformly distributed random non-singular matrices see [46]. The method offered in [46] is based on Bruhat decomposition of $GL(m,q)$ and easily generalizes for other classical matrix groups.

Define $\mathcal{S}(\sigma, \mathbb{F}_q)$, where $\sigma \in S_m$, to be a set of matrices $M = (r_{i,j})$ such that $r_{i,\sigma(i)} = 1$ and $r_{i,j} = 0$ if $j < \sigma(i)$ or $\sigma^{-1}(j) < i$. For example, if $m = 5$, $\sigma = (2,4,3,1,5) \in S_5$ we get all $M \in \mathcal{S}(\sigma, \mathbb{F}_q)$ as

$$M = \begin{pmatrix} 0 & 1 & * & * & * \\ 0 & 0 & 0 & 1 & * \\ 0 & 0 & 1 & 0 & * \\ 1 & 0 & 0 & 0 & * \\ 0 & 0 & 0 & 0 & 1 \end{pmatrix}$$

where by "*" we mean any number $x \in \mathbb{F}_q$. Denote by $B(m, \mathbb{F}_q)$ the set of invertible upper-triangular matrices. Then the algorithm that outputs a uniform random element of $GL(m, \mathbb{F}_q)$ is as follows.

Algorithm 1.

1. *Input m, q;*

2. *Sample $\sigma \in S_m$ with the distribution in which each $\sigma \in S_m$ appears with probability*

$$\frac{q^{-inv(\sigma)}}{\prod_{i=1}^{m}(1 + \frac{1}{q} + \cdots + \left(\frac{1}{q}\right)^{i-1})}$$

3. *Sample $M \in \mathcal{S}(\sigma, \mathbb{F}_q)$ uniformly;*

4. *Sample $B \in B(m, \mathbb{F}_q)$ uniformly;*

5. *Output $M' = B^T \cdot M$.*

The sets $\mathcal{S}(\sigma, \mathbb{F}_q)$ are the Schubert cells of $GL(m,q)$. They are in one-to-one correspondence with the elements of the Weyl group S_m of $GL(m,q)$. The group $GL(m,q)$ decomposes as a disjoint union of orbits of its upper-triangular group $B(m, \mathbb{F}_q)$ on those Schubert cells (Bruhat decomposition). Elements of both $B(m, \mathbb{F}_q)$ and $\mathcal{S}(m, \mathbb{F}_q)$ are easy to generate randomly with uniform distribution.

Denote the above matrix of 1's and stars corresponding to σ by M_σ. There are exactly $\binom{n}{2} - inv(\sigma)$ "stars" in M_σ corresponding to a permutation σ, so

$$|\mathcal{S}(\sigma, \mathbb{F}_q)| = q^{\binom{n}{2} - inv(\sigma)}, \qquad (*)$$

The probability

$$\frac{q^{-inv(\sigma)}}{f_m(\frac{1}{q})},$$

of sampling $\sigma \in S_m$ on step 2 is proportional to the size of the Schubert cell corresponding to σ, where $f_m(t)$ is the generating function for the number of complete flags over $\mathbb{F}_q$ mentioned in 4.2.

4.4. Generating k-tuples of $G = \mathbb{Z}_p \oplus \ldots \oplus \mathbb{Z}_p = (\mathbb{Z}_p)^{\oplus m}$ Let $G = \mathbb{Z}_p \oplus \ldots \oplus \mathbb{Z}_p = (\mathbb{Z}_p)^{\oplus m}$. As above, any generating k-subset of G must have $k \geq m$ elements. The case $k = m$ is done above, the case $k > m$ can be handled in a similar fashion as follows.

Denote the set of generating k-tuples of G by Γ_k. It decomposes as a disjoint union analogous to the Bruhat decomposition for $GL(m, q)$.

Pick a subset $I = \{i_1, \ldots, i_{k-m}\}$ of $\{2, \ldots, k\}$ such that $1 < i_1 < i_2 < \cdots < i_{k-m} \leq k$. Denote by M'_σ the $k \times m$ matrix obtained by inserting $k - m$ rows of zeros into M_σ so that they become rows $i_1, \ldots, i_{k-m}$ in the resulting matrix.

Denote by $\mathcal{C}(\sigma, \{i_1, \ldots, i_{k-m}\})$ the set of all matrices of the shape M'_σ, and by $B(k, I, \mathbb{F}_q)$ the set of all invertible lower-triangular matrix that in columns $i_1, \ldots, i_{k-m}$ have 1's on the diagonal and zeros below the diagonal. For example, if $m = 5$, $k = 7$, $\sigma = (2, 4, 3, 1, 5) \in S_5$, and $I = \{3, 5\}$ we have

$$M'_\sigma = \begin{pmatrix} 0 & 1 & * & * & * \\ 0 & 0 & 0 & 1 & * \\ 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & * \\ 0 & 0 & 0 & 0 & 0 \\ 1 & 0 & 0 & 0 & * \\ 0 & 0 & 0 & 0 & 1 \end{pmatrix}, \quad B(k, I, \mathbb{F}_q) = \left\{ \begin{pmatrix} * & 0 & 0 & 0 & 0 & 0 & 0 \\ * & * & 0 & 0 & 0 & 0 & 0 \\ * & * & 1 & 0 & 0 & 0 & 0 \\ * & * & 0 & * & 0 & 0 & 0 \\ * & * & 0 & * & 1 & 0 & 0 \\ * & * & 0 & * & 0 & * & 0 \\ * & * & 0 & * & 0 & * & * \end{pmatrix} \right\}$$

The algorithm yielding uniform generating k-subsets ($k > m$) of $G = (\mathbb{Z}_p)^{\oplus m}$ (uniform elements of Γ_k) is as follows.

Algorithm 2.

1. *Input k, m, q;*

2. *Sample a pair (σ, I), $\sigma \in S_m$, $I = \{i_1, \ldots, i_{k-m}\} \subset \{2, \ldots, k\}$ with the distribution in which each pair (σ, I) appears with probability proportional to*

$$q^{|I| - inv(\sigma)}$$

where $|I| = \sum_{\alpha=1}^{k-m} i_\alpha$.

3. Sample $M \in \mathcal{C}(\sigma, I)$ *uniformly;*

4. Sample $B \in B(k, I, \mathbb{F}_q)$ *uniformly;*

5. Output $M' = B \cdot M$.

Let us prove that the algorithm is correct. Consider a generating k-tuple $(g_1, \ldots, g_k) \in \Gamma_k$ for $k > m$. Written as a $k \times m$ matrix, it must have a minor of the full rank m. Such a minor is an invertible $m \times m$ matrix (and thus lies in the orbit of some Schubert cell $\mathcal{S}(\sigma, \mathbb{F}_q)$ of $GL(m, q)$), and the rest of the rows are linear combinations of the minor's rows. The full rank minor in question can be chosen as the 'topmost' of all full rank minors of the matrix, in the following sense.

Suppose that rows $i_1, \ldots, i_{k-m}$ are each a linear combination of their respective preceding rows. As we go from the top to the bottom row of the original $k \times m$ matrix of the k-tuple, each row is either linearly independent of the rows above it and is added to our maximal minor (up to m rows) or is linearly dependent on the preceding rows (up to $k - m$ rows) .

The 'topmost' maximal m-minor obtained in this fashion is an invertible matrix in a Schubert cell corresponding to some permutation $\sigma \in S_m$. So Γ_k decomposes into disjoint subsets numbered by a permutation $\sigma \in S_m$ and a list of rows $i_1, \ldots, i_{k-m}$ in which the next linear dependence occurs.

More precisely, any matrix with rows $i_1, \ldots, i_{k-m}$ linearly dependent on their respective preceding rows and the rest forming an invertible $m \times m$ matrix from the Schubert cell corresponding to $\sigma \in S_m$ can be obtained by multiplying a *unique* matrix of type M'_σ on the left by a *unique* $k \times k$ invertible lower-triangular matrix, which in columns $i_1, \ldots, i_{k-m}$ has 1's on the diagonal and zeros below the diagonal,

$$\Gamma_k = \bigcup_{I \in \mathcal{I},\, \sigma \in S_m} B(k, I, \mathbb{F}_q) \cdot \mathcal{C}(\sigma, I) \qquad \text{(disjoint)}$$

where $\mathcal{I}$ is the set of all $I = \{i_1, \ldots, i_{k-m}\}$ such that $1 < i_1 < \cdots < i_{k-m} \leq k$. All such sets I can be better enumerated as follows: take $i'_1 = i_1 - 1$, $i'_n = i_n - i_{n-1}$, $n = 2, \ldots, k - m$. Then any $k - m$ ordered set $\{i'_1, \ldots, i'_{k-m}\}$ such that $i'_n \geq 1$ and $\sum_{n=1}^{k-m} i'_n \leq k - 1$.

The size of the $B(k, I, \mathbb{F}_q)$ orbit of $\mathcal{C}(\sigma, I)$ is

$$(q-1)^m \, q^{\binom{k}{2} - \sum_{l=1}^{k-m}(k - i_l)} \, q^{\binom{m}{2} - inv(\sigma)} = (q-1)^m q^{\binom{m+k}{2} - k^2 + |I| - inv(\sigma)},$$

where $inv(\sigma)$ is the number of inversions in permutation σ, and $|I| = i_1 + \cdots + i_{k-m}$. Therefore the probability of sampling a pair (σ, I) is

$$\frac{q^{|I| - inv(\sigma)}}{\sum_{\sigma \in S_m,\, I \in \mathcal{I}} (q^{|I| - inv(\sigma)})} = \frac{q^{|I| - inv(\sigma)}}{\mathcal{M}_{k \times m}},$$

where $\mathcal{M}_{k\times m}$ is the number of full rank matrices of size $k \times m$. This is exactly the probability from the algorithm above, which concludes the proof.

4.5. Generating k-tuples of $G = \mathbb{Z}_{p^{i_1}} \oplus \ldots \oplus \mathbb{Z}_{p^{i_m}}$ Let us now consider the general case of a finite abelian group made of elementary components with the same p. Namely, consider

$$G = \mathbb{Z}_{p^{i_1}} \oplus \ldots \oplus \mathbb{Z}_{p^{i_m}},$$

where $i_1, \ldots, i_m$ need not be distinct.

A k-tuple $(g_1, \ldots, g_k)$ generates G if and only if the k-tuple modulo p,

$$(g'_1, \ldots, g'_k) = (g_1 \bmod p, \ldots, g_k \bmod p) \in (\mathbb{Z}_p)^{\oplus m}$$

generates $(\mathbb{Z}_p)^{\oplus m}$.

Indeed, let P_j be a straight line program in $(\mathbb{Z}_p)^{\oplus m}$ to $e'_j = (0, \ldots, 1, \ldots, 0)$ (with 1 on the j-th place):

$$(0, \ldots, 1, \ldots, 0) = P_j(g'_1, \ldots, g'_k).$$

Take the same program with $\{g_i\}$ instead of their images $\{g'_i\}$. With a suitable $\alpha \in \mathbb{Z}$, for this element $P_j(g_1, \ldots, g_k)$ of G we have

$$(a_1, \ldots, a_{j-1}, 1, a_{j+1}, \ldots, a_m) = \alpha \cdot P_j(g_1, \ldots, g_k), \quad \text{where} \quad a_i \bmod p = 0\,.$$

Therefore the subgroup of G generated by the original k-tuple contains elements with 1 on the l-th place and $a_i \bmod p = 0$, where $i \neq l$, in all other places. It follows that all $e_j = (0, \ldots, 1, \ldots, 0) \in G$ also lie in the subgroup generated by the k-tuple (to see this, we only need to apply Gaussian elimination.)

Write the elements $g_1, \ldots, g_k$ of the k-tuple in 'coordinates' as follows. For each

$$g_j = (h_j^1, \ldots, h_j^m) \in \mathbb{Z}_{p^{i_1}} \oplus \ldots \oplus \mathbb{Z}_{p^{i_m}}$$

write in a single row the coordinates of h^1 in $\mathbb{Z}_{p^{i_1}}$, $\ldots$, h^m in $\mathbb{Z}_{p^{i_m}}$, and form a block matrix with k columns (and $\sum_{n=1}^{m} i_n$ columns).

As we have just shown, only the last coordinate of each h_j^n (i.e. the last column of each block) matters in determining whether each $\mathbb{Z}_{p^{i_n}}$ (and therefore G) gets generated by the k-tuple.

Therefore, to obtain a random generating k-tuple of G, we can first find a generating k-tuple of $(\mathbb{Z}_p)^{\oplus m}$ (the desired, so to say, $\bmod p$) using *Algorithm 2.* Interpret the resulting matrix as that of the last coordinates of h_j^l and fill in the rest of their coordinates randomly.

4.6. Generating k-tuples of a general finite abelian group G Let $p_1, \ldots, p_m$ be distinct primes. A general finite abelian group G has a unique presentation

as

$$G = \mathbb{Z}_{{p_1}^{i_1}} \oplus \ldots \oplus \mathbb{Z}_{{p_1}^{i_s}} \oplus \ldots \oplus \mathbb{Z}_{{p_m}^{j_1}} \oplus \ldots \oplus \mathbb{Z}_{{p_m}^{j_t}} = F_{p_1} \oplus \ldots \oplus F_{p_m},$$

where $F_{p_i} = \mathbb{Z}_{{p_i}^{l_1}} \oplus \ldots \oplus \mathbb{Z}_{{p_i}^{l_s}}$, the sum of all elementary components with the same p_i.

Write an element $g \in G$ as $g = (f^1, \ldots, f^m)$, $f_i \in F_i$. A k-tuple $(g_1, \ldots, g_k)$ generates G if and only if the k-tuples of its components $(f_1^j, \ldots, f_k^j)$ generate F_{p_j}. Indeed, the following trivial lemma is true:

Lemma 7. *Let $F_1, \ldots, F_m$ be abelian groups, $|F_i| = p_i^{n_i}$. Then a k-tuple $(g_1, \ldots, g_k)$ generates $G = F_1 \oplus \ldots \oplus F_m$ if and only if for each $\alpha = 1, \ldots, m$ the k-tuple $(f_1^\alpha, \ldots, f_k^\alpha)$ generates F_α, where $g_j = (f_j^1, \ldots, f_j^m)$, $f_j \alpha \in F_\alpha$.*

Indeed, consider the following maps

$$\varphi_j \,:\, g \mapsto p_1 \cdots p_{j-1} p_{j+1} \cdots p_m \cdot g.$$

Clearly,

$$\varphi_j(\, f^1, \ldots, f^m\,) = (0, \ldots, f'^j, \ldots, 0).$$

Moreover, on F_j this map acts as an automorphism (that is, $\pi_j \circ \varphi_j \circ \iota$ is an automorphism of F_j, where $\pi_j : G \to F_j$ and $\iota : F_j \to G$ are the standard projection and inclusion). Applying φ_j to a k-tuple, we will get a k-tuple in F_j which should generate the entire F_j if the original k-tuple generated G, and vice versa, if the images of the original k-tuple generate F_j, then their preimages generate the entire j-th summand of G.

Therefore the algorithm that yields a uniformly distributed generating k-tuple of the general finite abelian group

$$G = \mathbb{Z}_{{p_1}^{i_1}} \oplus \ldots \oplus \mathbb{Z}_{{p_1}^{i_s}} \oplus \ldots \oplus \mathbb{Z}_{{p_m}^{j_1}} \oplus \ldots \oplus \mathbb{Z}_{{p_m}^{j_t}} = F_{p_1} \oplus \ldots \oplus F_{p_m},$$

is as follows.

Algorithm 3.

1. *For each prime $p_\alpha\,(\alpha = 1, \ldots, m)$ find a random uniform generating k-tuple $(f_1^\alpha, \ldots, f_k^\alpha)$ for $(\mathbb{Z}_{p_\alpha})^{\oplus n_\alpha}$, using* Algorithm 2.
2. *Choosing additional coordinates randomly, complete*
$$(f_1^\alpha, \ldots, f_k^\alpha) \in (\mathbb{Z}_{p_\alpha})^{\oplus n_\alpha}$$
to $(f'^\alpha_1, \ldots, f'^\alpha_k) \in F_{p_\alpha}$ (see Section 4.5).
3. *Output $g_j = (f_j^1, \ldots, f_j^m)$ for $j = 1, \ldots, k$.*

4.7. Nilpotent groups Let G be a nilpotent group. Consider its lower central series G_i $(G_0 = G)$, $G_{i+1} = [G, G_i]$, $i = 0, 1, \ldots, n$ and

$$G = G_0 \supset G_1 \supset G_2 \supset \cdots \supset G_n = \{e\}$$

By definition, all subgroups G_i are normal in G, and all groups $H_i = G_i/G_{i+1}$ are abelian. For each factor-group H_i define its *transversal* $\widetilde{H}_i \subset G$ to be the set of coset representatives of G_i/G_{i+1} with exactly one representative from each coset. Then every element $g \in G$ can be represented as $g = \widetilde{h}_0\widetilde{h}_1 \dots \widetilde{h}_n$, for some $\widetilde{h}_i \in \widetilde{H}_i$. Moreover, if $\widetilde{h}_i \in \widetilde{H}_i$, $i = 0, 1, \dots, n$ are random uniformly distributed elements of $\widetilde{H}_i$ for each i, then their product $g = \widetilde{h}_0\widetilde{h}_1 \dots \widetilde{h}_n$ is uniformly distributed in G. Therefore, if we can sample uniform random elements from the transversals $\widetilde{H}_i$, we can easily sample uniform random elements of G itself.

For nilpotent groups it is known that $\widetilde{H}_0 \subset G$ generates the entire group G. In view of this, producing uniformly distributed generating k-tuples can be done as follows:

Algorithm 4.

1. *Sample a uniform generating k-tuple $h_0^1, \dots, h_0^k$ of $H_0 = G/G_1$ using Algorithm 1.3. Denote $\widetilde{h}_0^1, \dots, \widetilde{h}_0^k$ the corresponding elements in $\widetilde{H}_0$.*

2. *Choose uniformly elements $\widetilde{h}_j^i \in \widetilde{H}_j$ for each $i = 1, \dots, k$, $j = 1, \dots, n$. Compute*
$$g_i = \widetilde{h}_0^i \cdot \widetilde{h}_1^i \cdot \dots \cdot \widetilde{h}_n^i \in G$$

3. *Output $(g_1, \dots, g_k)$.*

Consider a classical example. Let G be the group of upper-triangular $m \times m$ matrices over the finite filed $\mathbb{F}_p$ (p is prime) with 1's on the diagonal. Then G is nilpotent and each G_i consists of upper-triangular matrices with 1's on the main diagonal and zeros on i subdiagonals above the main diagonal (all the non-zero entries concentrated in the smaller triangle above the ith subdiagonal), and $H_i \cong (\mathbb{F}_q)^{m-i-1}$. Then the algorithm above amounts to picking a generating k-tuple in $(\mathbb{F}_q)^{m-1}$ to fill the 1st subdiagonals of the corresponding $m \times m$ matrices $g_1, \dots, g_k$, and picking all other non-zero entries of those matrices at random.

In general, we claim that the above algorithm will indeed produce a generating k-tuple of G. Indeed, it is well known that a generating k-tuples in H_0 are in one–to–one correspondence with k-tuple in $\widetilde{H}_0$ which generates the entire group G. Now simply take any elements in the remaining $\widetilde{H}_i$, and we obtain a random generating k-tuple of G. This finishes proof of the claim.

4.8. On random bits Our aim is to present algorithms for obtaining uniformly distributed generating k-tuples of groups G with known structure (abelian, nilpotent and supersolvable), which are effective (and, in a sense, optimal) in terms of *random bits*. By this we mean that the number of random bits (say,

calls to procedures for generating random elements of a finite field) is not significantly higher than that required for generating an arbitrary element of G in some standard presentation. For instance, when obtaining generating k-tuples of $G = (\mathbb{F}_q)^n$ (that is to say, $n \times n$ non-singular matrices with entries in $\mathbb{F}_q$) the number of times a random element of $\mathbb{F}_q$ needs to be generated is no greater than n^2.

It is a routine exercise to check that the algorithm outlined above uses a nearly optimal number of random bits. We will skip the details for the sake of simplicity. This completes the proof of Theorem 6.

5. Counting problem

5.1. Nilpotent groups Let G be a nilpotent group, $p_1, \ldots, p_m$ are distinct primes, and

$$G = F_{p_1} \oplus \ldots \oplus F_{p_m} ,$$

where F_p is the Sylow p-subgroup of G (see section 4.6 above). Assume $|F_{p_i}| = p_i^{\lambda_i}$. By the results of section 4.7 we have

$$\varphi_k(G) = \prod_{i=1}^{m} \varphi_k\big(F_{p_i}\big)$$

Also, for a p-group F_p,

$$F_p = \mathbb{Z}_p^{\alpha_1} \oplus \mathbb{Z}_{p^2}^{\alpha_2} \oplus \ldots$$

Let $\varkappa = \varkappa(F_p) = \alpha_1 + \alpha_2 + \ldots$. We have

$$\varphi_k(F_p) = \varphi_k\big(\mathbb{Z}_p^{\varkappa}\big) = \prod_{i=1}^{\varkappa} \left(1 - \frac{1}{p^{i-\varkappa+k}}\right) \geq \prod_{i=k-\varkappa+1}^{\infty} \left(1 - \frac{1}{p^i}\right)$$

Let $z = 1/p$. By Euler's pentagonal theorem

$$\prod_{i=1}^{\infty}(1 - z^i) = \sum_{m=1}^{\infty} z^{m(3m\pm1)} = 1 - z - z^2 + z^5 + z^7 - \ \ldots \geq 1 - z - z^2$$

Therefore

$$\varphi_k(F_p) \geq 1 - \frac{1}{p} - \frac{1}{p^2} \geq 1 - \frac{1}{2} - \frac{1}{4} = \frac{1}{4}$$

for $k = \varkappa(G)$, and

$$\varphi_k(F_p) \geq \varphi_{\varkappa+1}(F_p) = \prod_{i=2}^{\infty} \left(1 - \frac{1}{p^i}\right) \geq \frac{1 - \frac{1}{p} - \frac{1}{p^2}}{1 - \frac{1}{p}} = 1 - \frac{1}{p\,(p-1)}$$

for $k \geq \varkappa(G) + 1$. We conclude that when $k \geq \varkappa(G) + 1$ we have

$$\varphi_k(G) \geq \prod_{p=2}^{\infty} \left(1 - \frac{1}{p\,(p-1)}\right) > .373 > 1/3$$

where the product is over all primes p. The constant .373 is computed directly (see Remark 8 below). This proves Theorem 2 for nilpotent groups (cf. [23]).

6. General groups

In this section we will prove Theorem 1.

Let G be a finite group. Consider the following random process. Pick a uniform group element $g_1 \in G$. If $H_1 = \langle g_1 \rangle \neq G$, pick a uniform group element $g_2 \in G$. If $H_2 = \langle g_1, g_2 \rangle \neq G$, pick a another groups element, etc. Denote by τ the first time we generate the whole group. We claim that for all k and all G, and $r = \lceil \log |G| \rceil$, the probability $\mathbf{P}(\tau = k)$ minimizes when $G \simeq \mathbb{Z}_2^r$. Indeed, regardless of the group structure, for any given i the probability that $H_i \neq H_{i+1}$ is $1 - |H_i|/|G|$. Notice that then $|H_{i+1}|/|H_i| \geq 2$ with the equality always achieved when $G \simeq \mathbb{Z}_2^r$. Therefore $\mathbf{P}(\tau = k)$ minimizes in this case.

Now observe that $\varphi_k(G) = \mathbf{P}(\tau \leq k)$. Thus $\varphi_k(G)$ minimizes when $G \simeq \mathbb{Z}_2^r$, and it remains to compute $\varphi_k(\mathbb{Z}_2^r)$. Recall that $\varphi_k(\mathbb{Z}_2^r)$ is equal to the probability that a random $k \times r$-matrix over $\mathbb{F}_2$ has full rank and is equal to

$$\begin{aligned}
\varphi_k(\mathbb{Z}_2^r) \;&\geq\; \prod_{i=0}^{r-1} \left(1 - \frac{1}{2^{k-i}}\right) \\
&\geq \prod_{i=k-r+1}^{\infty} \left(1 - \frac{1}{2^i}\right) = \prod_{i=1}^{\infty} \left(1 - \frac{1}{2^{i\,m}}\right) \cdot \ldots \cdot \left(1 - \frac{1}{2^{i\,m+m-1}}\right) \\
&\geq \prod_{i=1}^{\infty} \left(1 - \frac{1}{2^{i\,m}}\right)^m = \left(\prod_{i=1}^{\infty} \left(1 - \frac{1}{(2^m)^i}\right)\right)^m,
\end{aligned}$$

where $m = k - r + 1$. Now use Euler's pentagonal theorem (see section 5.1) to obtain

$$\varphi_k \geq \left(1 - \frac{1}{2^m} - \frac{1}{2^{2m}}\right)^m \geq 1 - \frac{m+3}{2^m}$$

Now let $m = \lceil 2 \log_2(1/\epsilon) \rceil$, $m \geq 3$. Then $\varphi_k = \varphi_{r+m-1} > 1 - \epsilon$. This implies the result and finishes the proof of Theorem 1.

7. Final remarks

For the general finite group G constructing a direct generation algorithm as above seems to be a very complicated problem, regardless of the standard presentation we may select. For example, little is known of the classification of such pairs of permutations that generate an alternating group A_n, although the probability that a randomly chosen pair does generate A_n is high. Indeed, recall that $\varphi_2(A_n) = 1 - 1/n - O(1/n^2)$ (see [5]).

In such cases it seems best to use the Choose–and–Check method, i.e. produce k random elements of G and check if they form a generating k-tuple (cf. Proposition 2.1). As discussed earlier, for permutation groups and solvable matrix groups (see [42, 43]) this testing can be done in polynomial time.

On the other hand, if we are interested in a Monte Carlo algorithm which will work with high probability, it is known that if two random elements do not fix any point, they generate S_n with high probability (see [5, 53]). A similar result holds for other simple groups of Lie type (see [39, 40, 53]). This result is the key point in the proof of Theorem 3 (see [49]).

Regarding the efficient generation of the uniform group elements, let us recall that the well-known Lie–Kolchin theorem, which states that every connected solvable algebraic group is supersolvable, applies here. This hints at a possibility of a natural generalization of Theorem 6 and the results of [42]. We would like to note that the general subgroup algorithm for matrix groups in a white box representation was outlined in [24].

Also notice that, unlike the product replacement Markov chain on $\mathcal{N}_k(G)$, which requires that $k \geq 2\,\widetilde{\varkappa}(G)$ (see section 3.1), the Choose–and–Check method will efficiently work in case of solvable groups for all $k \geq \varkappa(G) + 1$, in view of Theorem 2.

8. Acknowledgments

We would like to thank L. Babai, G. Cooperman, P. Diaconis, W. Feit, L. Finkelstein, W. Kantor, L. Lovász, A. Lubotsky, G. Margulis, D. Randall, A. Shalev and E. Zelmanov for helpful conversations.

The second author was supported by NSF Postdoctoral Research Fellowship in Mathematical Sciences.

References

[1] D. Aldous, P. Diaconis, Shuffling cards and stopping times, Amer. Math. Monthly, vol. 93 (1986), 155-177

[2] D. Aldous, J. Fill, Reversible Markov Chains and Random Walks on Graphs, monograph in preparation, 1996

[3] N. Alon, J. H. Spencer, P. Erdős, The Probabilistic Method, Wiley, New York, 1992

[4] N. Alon, Y. Roichman, Random Cayley graphs and expanders, Rand. Str. Alg. vol. 5 (1994), 271–284

[5] L. Babai, The probability of generating the symmetric group, J. Comb. Th. Ser. A, vol. 52 (1989), 148–153

[6] L. Babai, Local expansion of vertex-transitive graphs and random generation in finite groups, Proc 23^{rd} ACM STOC, 1991, pp. 164–174

[7] L. Babai, Automorphism groups, isomorphism, reconstruction, *Handbook of Combinatorics* (R. L. Graham et al., eds.), Elsevier, 1996

[8] L. Babai, Randomization in group algorithms: Conceptual questions Groups and Computation II, DIMACS Series, vol. 28, AMS, Providence, 1997

[9] L. Babai, G. Hetyei, On the Diameter of Random Cayley Graphs of the Symmetric Group, Comb. Prob. Comp., vol. 1 (1992), 201–208

[10] R. Beals, L. Babai, Las Vegas Algorithms for Matrix Groups. Proc. 24^{th} IEEE FOCS (1993), 427–436.

[11] S. Bratus, G. Cooperman, L. Finkelstein, S. Linton, Constructive recognition of a black box group isomorphic to $GL(n, q)$, preprint (1998)

[12] S. Bratus, I. Pak, Fast constructive recognition of black box S_n and A_n, preprint, (1997)

[13] F. Celler, C.R. Leedham-Green, A non-constructive recognition agorithm for the special linear and other classical groups. DIMACS Series, vol. 28, AMS, Providence, 1997, 61–68

[14] F. Celler, C.R. Leedham-Green, S. Murray, A. Niemeyer and E.A. O'Brien, Generating random elements of a finite group, Comm. Alg., 23 (1995),4931–4948

[15] F.R.K. Chung, R.L. Graham, Random walks on generating sets for finite groups, The Electronic J. of Comb., 4, no. 2 (1997) #R7

[16] G. Cooperman, L. Finkelstein, Combinatorial tools for computational group theory, DIMACS Series, vol. 28, AMS, Providence, 1997

[17] H.S.M. Coxeter, W.O.J. Moser, *Generators and relations for discrete groups* (third edition), Springer, Berlin, 1972

[18] P. Diaconis, *Group Representations in Probability and Statistics*, IMS, Hayward, CA, 1988

[19] P. Diaconis, The cutoff phenomenon in finite Markov chains, Proc. Nat. Acad. Sci. U.S.A., vol. 93, 1996, 1659–1664

[20] P. Diaconis, B. Efron, Testing for independence in a two-way table: new interpretations of the chi-square statistic, Ann. Stat., vol. 13, 845–913

[21] P. Diaconis, L. Saloff-Coste, Comarison techniques for random walk on finite groups, Ann. Prob., vol. 21 (1993), 2131–2156

[22] P. Diaconis, L. Saloff-Coste, Walks on generating sets of groups, Prob. Th. Rel. Fields, vol. 105 (1996), 393–421

[23] P. Diaconis, L. Saloff-Coste, Walks on generating sets of abelian groups, Invent. Math. vol. 134 (1998), 251–199

[24] P. Diaconis, M. Shahshahani, The subgroup algorithm for generating uniform random variables, Prob. in the Eng. and Inf. Sci., vol. 1 (1987), 15–32

[25] J.D. Dixon, B. Mortimer, *Permutation Groups*, Springer, 1996

[26] C. Dou, M. Hildebrand, Enumeration and random random walks on finite groups, Ann. Prob., vol. 24 (1996), 987–1000

[27] P. Erdős, R.R. Hall, Probabilistic methods in group theory II, Houston J. Math., vol. 2 (1976), 173–180

[28] P. Erdős, A. Rényi, Probabilistic methods in group theory, Jour. Analyse Mathématique, vol. 14 (1965), 127–138

[29] F. Chen, L. Lovász, I. Pak, Lifting Markov chains to speed up mixing, To appear in the Thirty-First Annual ACM Symposium on Theory of Computing (STOC'99)

[30] W. Gaschütz, Die Eulershe funktion auflösbarer gruppen, Ill. J. Math., vol. 3 (1959), 469–476

[31] D. Gorenstein, *Finite Simple Groups*, Plenum, New York, 1982

[32] A. Greenhalgh, A model for random random-walks on finite groups, Comb. Prob. Comp., vol. 6 (1997), 49–56

[33] P. Hall, The Eulerian functions of a group, Quart. J. Math., vol. 7 (1936), 134–151

[34] M. Hall, *The Theory of Groups*, Chelsea, New York, 1976

[35] M. Hildebrand, Random walks supported on random points of $\mathbb{Z}/n\mathbb{Z}$, Prob. Th. Rel. Fields, vol. 100 (1994), 191–203

[36] N. Jacobson, *Basic Algebra I.*, Freeman, San Francisco, 1974

[37] W. Kantor, Simple groups in computational group theory. Proc. Int. Congress of Math., Vol. II (Berlin, 1998)

[38] W.M. Kantor, A. Seress, Black Box Classical Groups, preprint (1997)

[39] W.M. Kantor, A. Lubotzky, The probability of generating a finite classical group, Geom. Dedicata, vol. 36 (1990) 67–87

[40] M.W. Liebeck, A. Shalev The probability of generating a finite simple group, Geom. Dedicata, vol. 56 (1995) 103–113

[41] M.W. Liebeck and A. Shalev, Simple groups, probabilistic methods, and a conjecture of Kantor and Lubotzky, J. Algebra, vol. 184 (1996), 31–57.

[42] E. Luks, Computing in solvable matrix groups, Proc. 33rd IEEE Symp. Found. Comp. Sci. (1992), 111–120

[43] E. Luks, Computing the composition factors of a permutation groups in polynomial time, Combinatorica, vol. 7 (1987), 87–99

[44] P. Neumann, C. Praeger, Cyclic matrices over finite fields, J. London Math. Soc. (2), vol.52 (1995), pp. 263–284

[45] A.C. Niemeyer, C.E. Praeger, A recognition algorithm for classical groups over finite fields, Proc. London Math. Soc. vol. 77 (1998), 117–169

[46] I. Pak, When and how n chose k, DIMACS Series, vol. 43 (1998), 191–238, AMS, Providence

[47] I. Pak, Random walks on finite groups with few random generators, Electronic J. Prob., vol. 4 (1999), 1–11.

[48] I. Pak, Speeding up random walks on groups, in preparation

[49] I. Pak, How many elements surely generate a group, in preparation

[50] I. Pak, V.H. Vu, On finite geometric random walks, preprint (1998)

[51] D. Randall, Efficient random generation of nonsingular matrices, Rand. Str. Alg., vol. 4 (1993), 111–118

[52] Y. Roichman, On random random walks, Ann. Prob., vol. 24 (1996), 1001–1011

[53] A. Shalev, Probabilistic group theory, St. Andrews Lectures, Bath, 1997

[54] C.C. Sims, Computation with Permutation Groups., Proc. Second Symp. Symb. Alg. Man. (1971), 23–28, ACM

[55] D. Wilson, Random random walks on $\mathbb{Z}_2^d$, Prob. Th. Rel. Fields, vol. 108 (1997), 441–457