Algorithms in computational group theory: random selection

John D. Dixon

Carleton University, Ottawa, Canada

AofA (Maresias, Brazil, April 2008)

Another reason to come to Brazil!

Computational group theory (CGT)

After some early work in the 1950s and 1960s CGT really began with Sims' (1969) computations in permutation groups.

- 1970s and 1980s: Computation in permutation groups. Character tables (CAS). Construction of sporadic simple groups. Restricted Burnside problem. P-quotient algorithm. Coset enumeration. Matrix representations over finite fields (MEATAXE). Cohomology computations.
- CGT systems: GAP 3.1 (1992) and MAGMA (1993 out of CAYLEY)
- Since 1990: extensive development of underlying theory, improved algorithms, applications packages
- "Practical" algorithms vs. "Asymptotic" analysis

Concise description of groups

We shall consider here only **finite** groups. Different ways in which groups are described:

- Generators and relations: for example, $Dih(2n) := \langle x, y \mid x^2 = y^2 = (xy)^n = 1 \rangle$
- Through generators as a permutation group or as a matrix group (usually over a finite field)
- By a polycyclic presentation (for solvable groups)
- Implicitly as groups of automorphisms of geometric or algebraic objects
- Monster M of size $\sim 8 \times 10^{53}$ generated by two 196822×196822 matrices over $GF(2)$ (8×10^{10} bits)

```
                  +--------------+
                  |              |
                  |  1    2    3 |
                  |              |
                  |  4  top    5 |
                  |              |
                  |  6    7    8 |
                  |              |
   +--------------+--------------+--------------+--------------+
   |              |              |              |              |
   |  9   10   11 | 17   18   19 | 25   26   27 | 33   34   35 |
   |              |              |              |              |
   | 12  left  13 | 20 front  21 | 28 right  29 | 36  rear  37 |
   |              |              |              |              |
   | 14   15   16 | 22   23   24 | 30   31   32 | 38   39   40 |
   |              |              |              |              |
   +--------------+--------------+--------------+--------------+
                  |              |
                  | 41   42   43 |
                  |              |
                  | 44 bottom 45 |
                  |              |
                  | 46   47   48 |
                  |              |
                  +--------------+
```

Example by Martin Schönert

RUBIK'S CUBE

```
gap> cube := Group(
> ( 1, 3, 8, 6)( 2, 5, 7, 4)( 9,33,25,17)(10,34,26,18)(11,35,27,19),
> ( 9,11,16,14)(10,13,15,12)( 1,17,41,40)( 4,20,44,37)( 6,22,46,35),
> (17,19,24,22)(18,21,23,20)( 6,25,43,16)( 7,28,42,13)( 8,30,41,11),
> (25,27,32,30)(26,29,31,28)( 3,38,43,19)( 5,36,45,21)( 8,33,48,24),
> (33,35,40,38)(34,37,39,36)( 3, 9,46,32)( 2,12,47,29)( 1,14,48,27),
> (41,43,48,46)(42,45,47,44)(14,22,30,38)(15,23,31,39)(16,24,32,40) );;

gap> Size( cube );  43252003274489856000  # approximately 4.10^19
```

How can we generate random elements in a group?

Randomization is used in many CGT algorithms - ideally we should like to have a fast generator which produces a sequence of independent elements which are uniformly distributed.

The remainder of this talk considers the problem of generating random elements in a finite group G.

In some cases it is easy to generate random elements. We shall look at the other cases, the methods proposed and the questions which arise.

- A fast random element generator should not take more than $O(\lg |G|)$ group operations to produce each element (lg means log to base 2).

Virtual enumeration of a group

Let

$$G = G_0 \geq G_1 \geq \ldots \geq G_m = 1$$

be a series of subgroups of a finite group G. Let T_i be a set of right coset representatives of G_{i+1} in G_i ($i = 0, \ldots, m-1$), so $G_i = G_{i+1}T_i$.
Each element x of G can be written uniquely in the form $x = t_{m-1} \ldots t_1 t_0$ with each $t_i \in T_i$. In favorable situations $|T_0| + |T_1| + \ldots + |T_{m-1}|$ is much smaller than $|G|$ (closer to $O(\lg |G|)$).
A random selection of $t_i \in T_i$ for each i gives a random $x \in G$ for an average cost of $\Theta(m)$ group operations.

- (Sims 1969) Permutation groups with G_i as the stabilizer subgroup of $\{1, 2, \ldots, i\}$ (base and strong generating set).
- (Laue, Neubüser and Schoenwaelder 1982) Solvable groups with a normal series in which the successive indices equal primes (polycyclic presentation).

Linear groups

Sims' virtual enumeration trick may not work for matrix groups over finite fields because they do not have chains of subgroups where the successive indices are small.
For example, the important group $\mathrm{SL}(2,q)$ $(q > 3$ a prime power) has order $g := q(q^2-1)$ but the smallest index of a proper subgroup is $q+1 \approx g^{1/3}$.

- (P.M. Neumann and Praeger 1992) **constructive recognition program** seeks to recognize the composition factors of a linear group over a finite field in a way in which useful computations can be carried out. Currently, all known methods use selection of random elements extensively, so a different kind of random generator is needed.

Cubes in groups

If we do not have have a virtual enumeration of G, then we can approximate one as follows.
In place of the subgroups and sets of right coset representatives, choose $T_1, T_2, \ldots, T_m$ where each $T_i := \{1, x_i\}$, and define

$$C := \left\{x_1^{\varepsilon_1} x_2^{\varepsilon_2} \ldots x_m^{\varepsilon_m} \mid \text{each } \varepsilon_i = 0 \text{ or } 1\right\}.$$

C is called a **cube** in G.

- (Babai and Erdös 1982) If $m \geq \lg |G| + \lg \lg |G| + 0.5$ then there exist $x_1, x_2, \ldots, x_m \in G$ such that each element of G can be written in the form $x_1^{\varepsilon_1} x_2^{\varepsilon_2} \ldots x_m^{\varepsilon_m}$ in at least one way.

Black box groups

Black box groups are a computational model for a group G:

- We know a set of generators $x_1, \ldots, x_d$ for G
- We have a rough estimate of $\lg |G|$
- We can determine whether $x, y \in G$ are distinct
- We can compute the inverse x^{-1} and product xy of known elements of G

Probability distributions on groups

Suppose that P is a probability distribution on a group G of size g, and that U is the uniform distribution ($U(x) = 1/g$ for all $x \in G$).

- P is **ε-uniform** if $P(x) \geq (1-\varepsilon)/g$ for all x
- Difference between P and U in the **variational norm** is

$$\|P - U\|_{var} := \frac{1}{2} \sum_{x \in G} |P(x) - U(x)| = \max_{A \subseteq G} |P(A) - U(A)|$$

How do we find random elements in a black box group?

In a group for which we have a virtual enumeration with small factors it is straightforward to generate random elements. In favourable situations this requires $\Theta(\lg |G|)$ group operations to generate each random element.
For a black box group, we have not got a virtual enumeration, but want a method of generating elements which gives a sequence of (“almost") random elements. Some approaches:

- Random walks on a Cayley graph
- Product replacement algorithm
- Cooperman’s algorithm

Random cubes

For any list $x_1, x_2, ..., x_m$ of elements of G, the **random cube**

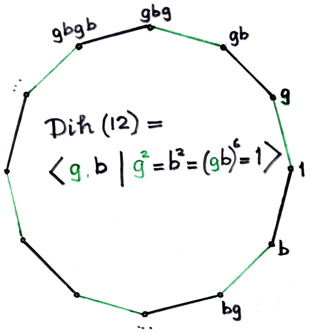
$$Cube(x_1, x_2, ..., x_m)$$

of **length** m is the probability distribution on G induced by $(\varepsilon_1, \varepsilon_2, ..., \varepsilon_m) \mapsto x_1^{\varepsilon_1} x_2^{\varepsilon_2} ... x_m^{\varepsilon_m}$ from the the uniform distribution on $\{0, 1\}^m$. A typical element generated this way is called a **random product**.

PROPERTIES OF RANDOM CUBES

- (Babai, Luks and Seress 1988) If $x_1, x_2, \ldots, x_m$ generate G, and H is a proper subgroup of G then an element chosen from $Cube(x_1, x_2, \ldots, x_m)$ has probability $\geq \frac{1}{2}$ of *not* lying in H (random subproduct lemma) [Easy exercise]
- (Erdös and Renyi 1965) If $m > 2\lg|G| + 2\lg(1/\varepsilon) + \lg(1/\delta)$ with $\varepsilon, \delta > 0$, then with probability $> 1 - \delta$ a random choice of $x_1, x_2, \ldots, x_m$ give a cube which is ε-uniform.

Generating random elements by random walks on Cayley graph Cayley(G,S)



A random walk on a Cayley graph of a group where the arcs correspond to a set of generators eventually reaches every vertex, **but** it may take a long time!

BABAI'S RANDOM WALK ALGORITHM (1991)

Given a set $S = \{y_1, ..., y_d\}$ of generators of G. Put $S_d := S$.
Algorithm: for $k = d, ..., m - 1$:

- compute y_{k+1} as the destination of a simple random walk on $\mathit{Cayley}(G, S_k)$ after $\Theta(\lg^4 |G|)$ steps starting at 1
- put $S_{k+1} := S_k \cup \{y_{k+1}\}$

Theorem (Babai 1991)**:** If $m = d + \Theta_{\delta,\varepsilon}(\lg |G|)$ then with probability $> 1 - \delta$ the distribution of $\mathit{Cube}(y_1, ..., y_m)$ is ε-uniform.

Remark The number of steps to construct the random element generator is $\Theta_{\delta,\varepsilon}(\lg^5 |G|)$

Nielsen transformations

Assume that G can be generated by k elements. Let Γ_k be the set of all k-tuples which generate G, and define the following **Nielsen transformations** on $(x_1, \ldots, x_k) \in \Gamma_k$ (for $i \neq j$):

- $R_{ij}^{\pm}$ replaces x_i by $x_i x_j^{\pm 1}$ and leaves other components fixed
- $L_{ij}^{\pm}$ replaces x_i by $x_j^{\pm 1} x_i$ and leaves other components fixed

The **Nielsen graph** N_k has vertex set Γ_k and edges defined by the transpositions $R_{ij}^{\pm}$ and $L_{ij}^{\pm}$.

Product replacement algorithm

F. Celler, C.R. Leedham-Green, S. Murray, A. Niemeyer and E.A. O'Brien (1995)

- Starting from a known k-tuple of generators of G, carry out and m-step random walk on N_k (they suggest that k be at least 10 and m be between 50 and 100). A sequence of 'random' elements of G is now made using the following procedure: make a single step in N_k (affecting the ith component, say) and output the new value of x_i.
- There is considerable evidence that the elements generated by this process can work well in some algorithms which require random elements.
- The algorithm has been analysed extensively by I. Pak, Babai and others. Pak has proved that one version of it produces close to uniform elements when $k = \Theta(\lg |G|)$ and $m = \Theta(\lg^5 |G|)$, but this does not explain the apparently superfast generator which has been observed in practice.

Cooperman's algorithm

G. Cooperman, "Towards a practical, theoretically sound algorithm for random generation in a finite group" (posted on arXiv:math 2002)

Cooperman claims to show the following:

- Let G be a black box group generated by $x_1, \ldots, x_d$. Then we can construct a ε-uniform random cube X of length $O(\lg(1/\varepsilon) \lg |G|)$ using $O(\lg^2 |G| + d \lg |G|)$ operations. We can take $X = Cube(x_m^{-1}, \ldots x_1^{-1}, x_1, \ldots, x_m)$ for sufficiently large m where, for each $i > d$, x_i is chosen at random from G using the distribution $Cube(x_{i-1}^{-1}, \ldots x_1^{-1}, x_1, \ldots, x_{i-1})$.

[Proof in the preprint is incomplete and has never been published, but the result is true.]

Group ring and probability distributions

- The **group ring** $\mathbb{R}[G]$ of a group G over the reals $\mathbb{R}$ consists of all formal sums $\sum_{x \in G} \alpha_x x$ (with $\alpha_x \in \mathbb{R}$) with the natural addition and the product given by convolution:

$$\left(\sum_{x \in G} \alpha_x x \right) \left(\sum_{y \in G} \beta_y y \right) := \sum_{z \in G} \left(\sum_{xy=z} \alpha_x \beta_y \right) z$$

- If Z is a probability distribution on G, identify Z with the element $\sum_{x \in G} \zeta_x x$ in the group ring $\mathbb{R}\left[G\right]$ where $\zeta_x = Z(x)$.
- If W is another probability distribution, then ZW (product in the group ring) is the distribution of the product of independent random variables from Z and W, respectively.
- Uniform distribution $U := (1/g) \sum_{x \in G} x$ where $g := \left| G \right|$.
- $Cube(x_1, x_2, ..., x_m) = 2^{-m} \prod_{i=1}^{m} (1 + x_i)$.

Group ring (cont'd)

- Involution $*$ on $\mathbb{R}[G]$ given by $\sum_{x\in G}\zeta_x x \mapsto \sum_{x\in G}\zeta_x x^{-1}$, and inner product on $\mathbb{R}[G]$ given by $\langle X, Y\rangle := tr(X^*Y)$ $(= \langle Y, X\rangle)$ where the **trace** $tr(\sum_{x\in G}\zeta_x x) := \zeta_1$. The inner product is just the dot product of the vectors of coefficients with respect to the obvious basis.
- If $Z = \sum_{x\in G}\zeta_x x$, then $\|Z\|^2 := \langle Z, Z\rangle = \sum_{x\in G}\zeta_x^2$.
- In general it is **not** true that $\|XY\| \leq \|X\|\,\|Y\|$, but $\|Xx\| = \|X\|$ for all $x \in G$.
- For a probability distribution Z we have $ZU = UZ = U$ and

$$4\,\|Z-U\|_{var}^2 \leq g\,\|Z-U\|^2 = g\,\|Z\|^2 - 1.$$

Main theorem (Cooperman's algorithm)

Theorem

Let $x_1, \ldots, x_d$ generate G with $d \le \lg |G|$ and consider the sequence of cubes $Z_m := Cube(x_1, \ldots, x_m)$ for $m \ge d$ where for $m > d$ we choose x_m at random from the distribution of the cube $Z_{m-1}^* Z_{m-1}$. Then for each $\varepsilon, \delta > 0$ there exists $C_{\varepsilon,\delta} > 0$ such that with probability at least $1 - \delta$ the cube $Z_m^* Z_m$ is ε-uniform whenever $m > C_{\varepsilon,\delta} \lg |G|$.

Note $Z_m^* Z_m = Cube(x_m^{-1}, \ldots x_1^{-1}, x_1, \ldots, x_m)$.
It takes $\Theta_{\delta,\varepsilon}(\lg^2 |G|)$ operations to construct the random element generator.

We shall outline a fairly simple proof based on properties of the group ring.

The main lemma

Lemma

Suppose that $Z := \mathit{Cube}(x_1, x_2, \ldots, x_m)$ *where* $x_1, x_2, \ldots, x_m$ *generate* G*. Then* $\|Z(1+x)/2\| \leq \|Z\|$ *for all* $x \in G$*, and either*

(a) Z^*Z *is* 0.2*-uniform, or*

(b) the probability that

$$\|Z(1+x)/2\|^2 < 0.975\,\|Z\|^2$$

holds for $x \in G$ *(under the distribution* Z^*Z*) is at least* 0.3*.*

An open problem

- The product replacement algorithm which is widely used as a “practical” means of generating random elements in a group has not been theoretically justified with parameters anywhere near those for which it is applied.
- On the other hand the theoretically justified algorithm of Cooperman appears to be too slow for many of the applications which are needed in practice.

Is it possible to find a random element generator which is both theoretically justifiable and faster than Cooperman’s algorithm?

References

Books on CGT

Á. Seress, “Permutation Group Algorithms" (Cambridge UP, 2003)

D.F. Holt (with B. Eick and E.A. O’Brien), “Handbook of Computational Group Theory" (Chapman & Hall, 2005)

Random element generator via Cayley graphs

L. Babai, Local expansion of vertex-transitive graphs and random generation in finite groups, Proc. 23rd ACN Symp. Theory of Comput. (1991) 164–174.

References (cont'd)

Product replacement algorithm

F. Cellar etal., Generating random elements of a finite group, Comm. Algebra 23 (1995) 4931–4948.

I. Pak, What do we know about the product replacement algorithm? in "Groups and Computation III" (Kantor and Seress, eds.), Berlin, 2001, pp. 301–347.

C.R. Leedham-Green and S.H. Murray, Variants of product replacement, Contemp. Math. 298, Amer. Math. Soc., 2002 (pp. 97–104).

Cooperman's algorithm

G. Cooperman, Towards a practical, theoretically sound algorithm for random generation in finite groups (unpublished ms. posted on arXiv:math May 2002).

J.D. Dixon, Generating random elements in finite groups (submitted to Electronic J. Comb.)