Generic Complexity

Robert Gilman

Stevens Institute of Technology

Methods of Logic in Mathematics III
St. Petersburg
June 2006

Introduction

Generic complexity is a new way to measure the difficulty of a computational problem.

The definition emerged in the last few years from the study of certain problems in combinatorial group theory.

The main idea is to focus attention on what happens most of the time instead of in the worst case or on average.

We are just beginning to understand the implications of this approach.

Generic Properties

The idea of ignoring rare cases is not new. Gromov's 1987 seminal paper on word hyperbolic groups contains

Theorem

Let $k \in \mathbf{Z}$ $(k \geq 2)$ *and let* $A = \{a_1^1, a_2^1, \cdots, a_k^1\}$ *be an alphabet. Let* $i \in \mathbf{Z}$ $(i \geq 0)$ *and let* $(n_1, \cdots, n_i)$ *be a sequence of positive integers. Let* $N = N(k, i, n_1, \cdots, n_i)$ *be the number of group presentations* $G = \langle a_1, \cdots, a_k \mid r_1, \cdots, r_i \rangle$ *such that* $r_1, \cdots, r_i$ *are reduced words in the alphabet* A *such that the length of* r_j *is* n_j *for* $j = 1, 2, \cdots, i$. *If* N_{h} *is the number of hyperbolic groups in this collection and if* $n = \min\{n_1, \cdots, n_i\}$ *then* $\lim_{n \to \infty} N_{\mathrm{h}}/N = 1$.

Proved by Ol′shanskii (1992)

In other words hyperbolicity is generic.

Other authors include Arzhantseva, Champetier Cherix, Ghys, Kapovich, Myasnikov, Schupp, Shpilrain, Ollivier, Zuk.

Generic versions of the Freiheitssatz for all finitely presented groups were proved by Arzhantseva, Ol′shanskii (1996) and by Cherix, Schaeffer, Gilles (1998).

Complexity

Around the same time as the developments mentioned above several people including Dima Bormotov, Roger Kuhlman, Alexei Myasnikov, Roger Needham, Hamish Short, Vladimir Shpilrain were working under the leadership of Gilbert Baumslag on the Magnus Computational Group Theory Package, a symbolic algebra package for problems in combinatorial group theory.

The Magnus Package was designed for experiments and solutions to instances of recursively unsolvable problems, and it had what was then an advanced feature, a graphical user interface.

Quotpick by Derek Holt and Sarah Rees was another step in this direction.

Combinatorial group theory has its own computational tradition extending back for more that a century. Almost all the problems are recursively unsolvable. Here is an example.

The Word Problem

Given a presentation $G = \langle a, b, c \mid r, s, t \rangle$ and a word w in the generators decide if w represents the identity in G.

The people working on Magnus noticed that simple strategies could work well for many instances of difficult or recursively unsolvable problems. Computer scientists had come to similar conclusions a little earlier.

Garey and Johnson in their book *Computers and Intractability* state that while the worst case complexities of the simplex algorithm for linear programming and of the branch and bound algorithm for the knapsack problem are exponential, nevertheless these algorithms run quickly in practice. They also say that examples like these are quite rare.

Experience with Magnus suggested however that this phenomenon might not be uncommon. The difficult instances of a problem might be hard to find while the set of easy instances might be ubiquitous.

This theme was developed in the paper

Generic case complexity - decision problems in group theory and random walks, by Kapovich, Myasnikov, Schupp, and Shpilrain.

They define generic complexity and show that for a large class of finitely generated groups the word, conjugacy and membership problems are generically linear time.

That is, there is an algorithm which runs in linear time on a generic set of inputs.

Generic Sets

Generic complexity depends on the notion of a generic set.

A subset S of A^* is generic if

$$\lim_{n\to\infty} \frac{|S \cap B_n|}{|B_n|} = 1$$

where B_n is the set of all words in A^* of length at most n and $|\ |$ denotes cardinality.

If the limit converges exponentially fast, S is exponentially generic.

In general for any countable set X express X as an ascending union of finite subsets.

$$X = \cup B_n$$

B_n may be thought of as the elements of size at most n.

A subset S of X is generic if

$$\lim_{n\to\infty} \frac{|S \cap B_n|}{|B_n|} = 1$$

If the limit exists and is equal to ρ, S has asymptotic density ρ. If $\rho = 0$, S is called negligible.

More generally consider probability distributions $\mu_n : B_n \to [0,1]$ and require $\lim_{n\to\infty} \frac{\mu_n(S\cap B_n)}{\mu_n(B_n)} = 1$. But we restrict ourselves to the uniform case above.

A Sample Generic Set

Fix $A = \{a, a^{-1}, b, b^{-1}, \ldots\}$

Let $f : A^* \to Z$ be a surjective monoid homomorphism from A^* to the integers, Z, under addition.

Since Z is infinite, we expect that a random word $w \in A^*$ is very unlikely to have image 0 (or any other fixed integer) in Z. Our intuition is correct.

The set of all words with non-zero image in Z is generic in A^*.

An Unsolvable Problem with Linear Generic Time Complexity

Let $A^* \to G$ be a choice of generators for a group G and assume G has an infinite cyclic quotient, $G \to Z$.

For example $G = H \times Z$ where H has unsolvable word problem.

Let $f : A^* \to Z$ be the composition of $A^* \to G \to Z$.

Algorithm

Given w compute $f(w)$.
If $f(w) \neq 0$, we say "w does not represent 1 in G"
Else say "Don't know".

The Algorithm succeeds in linear time on a generic set.

The paper by Kapovich, Myasnikov, Schupp, and Shpilrain has more and deeper results along these lines.

If we were building a cryptosystem which could be broken by solving the word problem for G, we would be on shaky ground if we knew only that the word problem for G were recursively unsolvable.

Two Generic Complexity Results

- **The positive word problem.** This is a search problem in which the input is a word defining the identity $G = \langle a, b, c \mid r, s, t \rangle$ and the output is a sequence of conjugates of relators whose product is w. In his dissertation Alexander Ushakov shows that the problem is generically polynomial and there is a practical algorithm.
- **The Halting Problem.** Joel Hamkins and Alexei Myasnikov have shown that the Halting Problem is generically decidable in polynomial time for Turing machines with a semi-infinite tape. Alexander Rybalov has shown that convergence cannot be exponential.

Notice that for the positive word problem we have a complexity result on a non-recursive set of inputs.

Post Correspondence Problem

PCP

Input a finite set of pairs of words $\{(w_1, v_1) < \dots\}$ over $\{a, b\}^*$.
Decide if $w_{i_1} \cdots w_{i_k} = v_{i_1} \cdots v_{i_k}$ for some sequence of indices.

B_n is the collection of inputs with n pairs of words of length between 1 and n

Algorithm

Check if for some (w_i, v_i) one is a prefix of the other.
If yes, say "Don't Know"
Else say "No"

PCP is exponentially generically linear.

The Subset Sum Decision Problem

Subset Sum

Decide for a sequence of natural numbers $c, w_1, \ldots, w_n$ whether the equation $\sum x_i w_i = c$ has a solution with $x_i = 0, 1$.

Subset Sum is NP-complete, but the corresponding optimization problem is routinely solved in practice and gives a solution to the decision problem. It seems that difficult instances of subset sum are rare.

Theorem

Subset sum is generically linear.

How to define an appropriate generic set?

B_n be the set of inputs of length n where length means the number of bits.

Pick an alphabet $\{0, 1, \hat{1}\}$ and consider the language L of all strings beginning with $\hat{1}$.

For any string in L, a substring which begins with $\hat{1}$ and continues up to but not including the next $\hat{1}$ is interpreted as a binary number by taking $\hat{1} = 1$.

For example $c = 5, w_1 = 3, w_2 = 4$ is encoded as $\hat{1}01\hat{1}1\hat{1}00$.

B_n consists of all words in L of length at most n; and a subset $M \subset L$ is generic if

$$\lim_{n\to\infty} \frac{|M \cap B_n|}{|B_n|} = 1.$$

Let S_n be the sphere of radius n; that is, S consists of the 3^{n-1} strings in L of length exactly n. It is straightforward to show that M is generic if

$$\lim_{n\to\infty} \frac{|M \cap S_n|}{|S_n|} = 1.$$

Algorithm

Input $c, w_1, \ldots, w_n$.
If c is equal to some w_i, say "Yes"
Else "Don't know".

Check that the set of inputs for which c matches some w_i is generic.

Equivalently check that the set of inputs for which c does not match some w_i is negligible.

1. There is no match if c is the whole input. The fraction of S_n for which this happens is $(2/3)^{n-1}$.
2. A similar count works if c is sufficiently large. In fact if c includes at least $m = \lfloor (\log n)/2 \log 3 \rfloor$ bits.
3. Otherwise c involves at most m bits. The input may be viewed as a product of words of length m (with a little bit left over) in which the first word occurs only once.

Another version of Subset Sum

Here is a practical algorithm which is generic with respect to a different formulation of Subset Sum.

Consider only instances in which all weights w satisfy $w \leq b$ for some fixed number b.

B_n consists of all instances with n weights $w_1, \ldots, w_n$ and $1 \leq c \leq nb$.

The following algorithm is adapted from one by G. d'Atri and C. Puech.

Algorithm

Compute $w_1 + w_2 + \cdots$ until one of the following happens.

1. If $w_1 + w_2 + \cdots + w_j = c$, say "Yes" and halt.
2. If $w_1 + w_2 + \cdots + w_n < c$, say "No" and halt.
3. If $w_1 + w_2 + \cdots + w_{j-1} < c < w_1 + w_2 + \cdots + w_j$, then
    1. If $w_1 + w_2 + \cdots + w_{j-1} + w_k = c$ for some k with $j < k \leq n$, say "yes" and halt.
    2. Else say "Don't know" and halt.

Clearly the Algorithm is correct and runs in linear time.

Estimate the probability of a “Don't know” answer.

Since c is chosen uniformly from the interval $[1, nb]$, the probability of $w_1 + w_2 + \cdots + w_{j-1} < c < w_1 + w_2 + \cdots + w_j$, is $(w_j - 1)/nb \leq 1/n$.

Further the probability that there is no suitable x_k is $((b-1)/b)^{(n-j)}$.

Thus the probability of “Don't know” is at most $\sum_{j=1}^{n}(1/n)((b-1)/b)^{(n-j)} \leq b/n$.

The Algorithm says “Yes” or “No” on a generic set.

3-Satisfiability

Consider the following heuristic argument for 3-Sat. An instance is a finite sequence of clauses

$$[10' \vee 101 \vee 1] \wedge [110 \vee 11' \vee 111] \wedge \cdots$$

where the variables are $1, \ldots, n$ (base 2) and $'$ means negation.

If the eight different clauses involving variables $1, 10, 11$ and their negations all appear in the input, then the formula is not satisfiable.

Think of inputs as finite words over the countable alphabet of clauses.

The set of finite words which omit some fixed letter of the alphabet should be asymptotically negligible, and likewise for the set of words omitting any of the eight clauses just mentioned.

The algorithm which searches the input for all these clauses should find them in linear time on a generic set of inputs.

To make this argument more rigorous start with the regular language of clauses

$$R = [1(0+1)^*(\vee +' \vee)1(0+1)^*(\vee +' \vee)1(0+1)^*(]+'])$$

over the finite alphabet $\Sigma = \{[, 0, 1, \vee, ']\}$.

Instances of 3-Sat are words in the free submonoid $(R\wedge)^*$ of Σ^*.

Let B_n be the ball of radius n in Σ^*. Instances of 3-Sat are selected uniformly from $R \cap B_n$.

It suffices to show that for any $w \in R$, there exist constants $C > 0$ and $\lambda < 1$ such that $|(S\wedge)^* \cap B_n| \leq C\lambda^n |(R\wedge)^* \cap B_n|$ where $S = R - \{w\}$.

Relative density of regular monoids

Lemma

Let R be a regular language over Σ and $\#$ a letter not in Σ. Suppose R has words of length n for all n greater than some constant. Let S be a proper subset of R. Then there exist constants $C > 0$ and λ, $0 \leq \lambda < 1$, such that

$$|(S\#)^* \cup B_n| \leq C\lambda^n |(R\#)^* \cup B_n|$$

for n large enough.

The proof is an application of an extension of the Perron Frobenius Theorem.

Average Case Complexity

We note that generic complexity is more general that average case complexity in that it makes sense for recursively unsolvable problems. Also it is not too difficult to prove that with some mild restrictions on the probability distributions $\mu_n : B_n \longrightarrow [0,1]$ the following theorem.holds.

Theorem

If a computational problem has average case complexity $O(f(n))$ then it has generic complexity $O(f(n))$.

Questions

The most interesting theoretical question is what is the structure of computational problems? Many of them seem to consist mostly of easy cases with a so-called black hole of rare and difficult instances. There is some indication that the concept of black hole can be made precise. Alexei Myasnikov may discuss this point in his talks.

One may also ask about practical applications such as a systematic approach to evaluating the difficulty of cryptoprimitives.

Another question concerns the effect of different choices of the B_n's.

Bibliography

- G. d'Atri and C. Puech, Probabilistic analysis of the subset sum problem, Discrete Applied Mathematics 4:329-334, 1982.
- Debreu, G. and Herstein I., Nonnegative square matrices, Econometrica, **21** 1953, 597-607
- M. Garey and J. Johnson, *Computers and Intractability, A Guide to NP-Completeness*, W. H. Freeman, 1979.
- I. Kapovich, A. Myasnikov, P. Schupp and V. Shpilrain, Generic case complexity - decision problems in group theory and random walks, J. Algebra vol 264 (2003) 665–694
- H. Kellerer, U. Pferschy and D. Pisinger, *Knapsack Problems*, Springer
- A. Miasnikov, V. Shpilrain and A. Ushakov, A practical attack on a braid group based cryptographic protocol, Lecture Notes in Computer Science **3621**, CRYPTO 2005, Springer Verlag.