УДК 510.52

О ГЕНЕРИЧЕСКОЙ СЛОЖНОСТИ ПРОБЛЕМЫ ДИСКРЕТНОГО ЛОГАРИФМА[1]

А. Н. Рыбалов

Институт математики им. С. Л. Соболева СО РАН, г. Новосибирск, Россия

Генерический подход к алгоритмическим проблемам предложен А. Мясниковым, И. Каповичем, П. Шуппом и В. Шпильрайном в 2003 г. В рамках этого подхода рассматривается поведение алгоритмов на множествах почти всех входов. Изучается генерическая сложность классической проблемы дискретного логарифма в полях $\mathrm{GF}(p)$, где p — простое. Доказывается, что её естественная подпроблема генерически трудноразрешима (то есть трудна для почти всех входов) при условии, что проблема дискретного логарифма трудноразрешима в классическом смысле.

Ключевые слова: *генерическая сложность, проблема дискретного логарифма, вероятностный алгоритм.*

DOI 10.17223/20710410/33/8

ON GENERIC COMPLEXITY OF THE DISCRETE LOGARITHM PROBLEM

A. N. Rybalov

Sobolev Institute of Mathematics SB RAS, Novosibirsk, Russia

E-mail: alexander.rybalov@gmail.com

Generic-case approach to algorithmic problems was suggested by Miasnikov, Kapovich, Schupp and Shpilrain in 2003. This approach studies behaviour of an algorithm on typical (almost all) inputs and ignores the rest of inputs. Many classical undecidable or hard algorithmic problems become feasible in the generic case. But there are generically hard problems. In this paper, we consider generic complexity of the classical discrete logarithm problem. We fit this problem in the frameworks of generic complexity and prove that its natural subproblem is generically hard provided that the discrete logarithm problem is hard in the worst case.

Keywords: *generic complexity, discrete logarithm problem, probabilistic algorithm.*

Введение

В работе [1] развита теория генерической сложности вычислений. В рамках этого подхода алгоритмическая проблема рассматривается не на всём множестве входов, а на некотором подмножестве «почти всех» входов. Такие входы образуют так называемое генерическое множество. Понятие «почти все» формализуется введением естественной меры на множестве входных данных. С точки зрения практики, алгоритмы, решающие быстро проблему на генерическом множестве, так же хороши, как и быстрые алгоритмы для всех входов. Классическим примером такого алгоритма является

[1]Работа поддержана грантом РФФИ № 15-41-04312.

симплекс-метод — он за полиномиальное время решает задачу линейного программирования для большинства входных данных, но имеет экспоненциальную сложность в худшем случае. Более того, может так оказаться, что проблема трудноразрешима или вообще неразрешима в классическом смысле, но легкоразрешима на генерическом множестве. В [1, 2] доказано, что таким поведением обладают многие алгоритмические проблемы алгебры, а в [3] построено генерическое множество, на котором разрешима классическая проблема остановки для машин Тьюринга с лентой, бесконечной в одном направлении. Для многих классических NP-полных проблем существуют полиномиальные генерические алгоритмы [4].

С точки зрения современной криптографии интересны такие алгоритмические проблемы, которые, являясь (гипотетически) трудными в классическом смысле, остаются трудными и в генерическом смысле, т. е. для почти всех входов. Это объясняется тем, что при случайной генерации ключей в криптографическом алгоритме происходит генерация входа некоторой трудной алгоритмической проблемы, лежащей в основе алгоритма. Если проблема генерически легкоразрешима, то для почти всех таких входов её можно быстро решить и ключи почти всегда будут нестойкими. Поэтому проблема должна быть генерически трудной. В данной работе изучается генерическая сложность классической проблемы криптографии — проблемы дискретного логарифма в полях $\mathrm{GF}(p)$, где p простое. До сих пор не известно полиномиальных алгоритмов её решения; на предположении об её трудноразрешимости основаны многие криптографические алгоритмы [5]. В работе доказывается, что эта проблема генерически неразрешима за полиномиальное время при условии отсутствия полиномиального вероятностного алгоритма её решения в худшем случае. Более того, существует правдоподобная гипотеза о том, что любой полиномиальный вероятностный алгоритм можно эффективно дерандомизировать, т. е. построить полиномиальный детерминированный алгоритм, решающий ту же задачу. Хотя это пока ещё не доказано, имеются серьёзные результаты в этом направлении [6]. При доказательстве основного результата работы использованы методы, развитые в [7, 8].

1. Генерические алгоритмы

Пусть I есть множество всех входов некоторой алгоритмической проблемы и I_n — множество всех входов размера n. Для подмножества $S \subseteq I$ определим последовательность

$$\rho_n(S) = \frac{|S \cap I_n|}{|I_n|},\ n = 1, 2, 3, \ldots$$

Заметим, что $\rho_n(S)$ — это вероятность попасть в S при случайной и равновероятной генерации входов из I_n. *Асимптотической плотностью* S назовём предел (если он существует)

$$\rho(S) = \lim_{n\to\infty} \rho_n(S).$$

Множество S называется *генерическим*, если $\rho(S) = 1$, и *пренебрежимым*, если $\rho(S) = 0$. Очевидно, что S генерическое тогда и только тогда, когда его дополнение $I \setminus S$ пренебрежимо.

Алгоритм $\mathcal{A}$ с множеством входов I и множеством выходов $J \cup \{?\}$ $(? \notin J)$ называется *генерическим*, если

1) $\mathcal{A}$ останавливается на всех входах из I;
2) множество $\{x \in I : \mathcal{A}(x) =?\}$ пренебрежимо.

Генерический алгоритм $\mathcal{A}$ вычисляет функцию $f : I \to J$, если для всех $x \in I$ $\mathcal{A}(x) = y \in J \Rightarrow f(x) = y$. Ситуация $\mathcal{A}(x) =?$ означает, что $\mathcal{A}$ не может вычислить функцию f на аргументе x. Но условие 2 гарантирует, что $\mathcal{A}$ корректно вычисляет f на почти всех входах (входах из генерического множества).

2. Проблема дискретного логарифма

Напомним, что проблема дискретного логарифма состоит в вычислении функции $dl : I \to \mathbb{N}$, где I — это множество троек (a, p, g_p), таких, что p — простое число, g_p — фиксированный первообразный элемент в поле $\mathrm{GF}(p)$ и $a \in \mathrm{GF}(p)$, $a \neq 0$. Сама функция dl определяется следующим образом:

$$dl(a, p, g_p) = x \Leftrightarrow g_p^x = a \in GF(p).$$

Под размером входа понимается число разрядов в двоичной записи числа p. В настоящее время неизвестно полиномиальных алгоритмов (даже вероятностных), решающих проблему дискретного логарифма. Это обстоятельство лежит в основе криптостойкости многочисленных криптографических алгоритмов [5].

Для изучения генерической сложности этой проблемы необходимо провести некоторую стратификацию на множестве входов. Рассмотрим любую бесконечную последовательность простых чисел

$$\pi = \{p_1, p_2, \ldots, p_n, \ldots\},$$

удовлетворяющую условию $2^n \leqslant p_n < 2^{n+1}$ для любого n. Такие последовательности существуют благодаря известному постулату Бертрана, доказанному П. Л. Чебышевым. Будем называть такую последовательность *экспоненциальной*. Теперь определим функцию dl_π как ограничение функции dl на множество троек (a, p, g_p), таких, что $p \in \pi$. Заметим, что для этой функции множество всех входов размера n состоит из троек (a, p, g_p) с фиксированными p, g_p и произвольным $a \in \{1, \ldots, p-1\}$. Очевидно, что проблема вычисления dl_π является подпроблемой вычисления dl. Следующая лемма показывает, что некоторые такие подпроблемы так же трудны, как и оригинальная проблема дискретного логарифма.

Лемма 1. Если не существует полиномиального вероятностного алгоритма для вычисления dl, то найдется такая экспоненциальная последовательность простых чисел π, что и для вычисления dl_π нет полиномиального вероятностного алгоритма.

Доказательство. Пусть $P_1, P_2, \ldots$ — все полиномиальные вероятностные алгоритмы. Из предположения о том, что не существует полиномиального вероятностного алгоритма для вычисления dl, следует, что для любого алгоритма P_n существует бесконечно много полей $\mathrm{GF}(p)$, в которых он не может вычислить dl. Из этого следует, что можно выбрать последовательность $\pi' = \{p_1, p_2, \ldots\}$ так, чтобы алгоритм P_n не вычислял dl в поле $\mathrm{GF}(p_n)$ и для любого n выполнялось бы $p_{n+1} > 2p_n$. Последовательность π' можно расширить до экспоненциальной последовательности π, добавив где нужно новые члены. Заметим теперь, что dl_π и будет той функцией, для вычисления которой не существует полиномиального алгоритма. ■

3. Основной результат

Следующий результат говорит о том, что проблема дискретного логарифма остаётся вычислительно трудной и в генерическом случае при условии её трудноразрешимости в худшем.

Теорема 1. Пусть π — любая экспоненциальная последовательность простых чисел. Если существует полиномиальный генерический алгоритм, вычисляющий функцию dl_π, то существует полиномиальный вероятностный алгоритм, вычисляющий dl_π для всех входов.

Доказательство. Пусть существует полиномиальный генерический алгоритм $\mathcal{A}$, вычисляющий функцию dl_π. Построим вероятностный полиномиальный алгоритм $\mathcal{B}$, вычисляющий dl_π на всем множестве входов. Алгоритм $\mathcal{B}$ на входе (a, p, g_p) будет работать следующим образом:

1) сгенерировать случайно и равномерно $y \in \{0, \ldots, p-1\}$ и вычислить $a' = ag_p^y$;
2) запустить алгоритм $\mathcal{A}$ на (a', p, g_p);
3) если $\mathcal{A}(a', p, g_p) = z \in \mathbb{N}$, то $a' = g_p^z = ag_p^y = g_p^{x+y}$, откуда $x = z - y \bmod (p-1)$ — дискретный логарифм для исходной задачи (a, p, g_p);
4) если $\mathcal{A}(a', p, g_p) = ?$, то выдать 0.

Заметим, что данный полиномиальный вероятностный алгоритм может выдать неправильный ответ только на шаге 4. Докажем, что вероятность этого меньше $1/2$. Действительно, $a' = ag_p^y$ при $y \in \{0, \ldots, p-1\}$ пробегает все ненулевые элементы поля $\mathrm{GF}(p)$, поэтому множество $\{(a', p, g_p) : y \in \{0, \ldots, p-1\}\}$ совпадает с множеством всех входов размера n. Но алгоритм $\mathcal{A}$ генерический, поэтому доля тех входов (a', p, g_p), на которых он выдаёт неопределённый ответ, стремится к 0 с ростом n и с некоторого момента становится меньше $1/2$. ■

Непосредственным следствием доказанной теоремы является

Теорема 2. Если для вычисления функции dl не существует полиномиального вероятностного алгоритма, то существует экспоненциальная последовательность π, такая, что для вычисления функции dl_π не существует генерического полиномиального алгоритма.

ЛИТЕРАТУРА

1. *Kapovich I., Miasnikov A., Schupp P., and Shpilrain V.* Generic-case complexity, decision problems in group theory and random walks // J. Algebra. 2003. V. 264. No. 2. P. 665–694.
2. *Kapovich I., Miasnikov A., Schupp P., and Shpilrain V.* Average-case complexity for the word and membership problems in group theory // Adv. Math. 2005. V. 190. P. 343–359.
3. *Hamkins J. D. and Miasnikov A. G.* The halting problem is decidable on a set of asymptotic probability one // Notre Dame J. Formal Logic. 2006. V. 47. No. 4. P. 515–524.
4. *Gilman R., Miasnikov A. G., Myasnikov A. D., and Ushakov A.* Report on generic case complexity // Herald of Omsk University. 2007. Special Issue. P. 103–110.
5. *Mao B.* Современная криптография: теория и практика. М.: Вильямс, 2005. 768 с.
6. *Impagliazzo R. and Wigderson A.* P = BPP unless E has subexponential circuits: Derandomizing the XOR Lemma // Proc. 29th STOC. El Paso: ACM, 1997. P. 220–229.
7. *Myasnikov A. and Rybalov A.* Generic complexity of undecidable problems // J. Symbolic Logic. 2008. V. 73. No. 2. P. 656–673.
8. *Rybalov A.* Generic complexity of presburger arithmetic // Theory Comput. Systems. 2010. V. 46. No. 1. P. 2–8.

REFERENCES

1. *Kapovich I., Miasnikov A., Schupp P., and Shpilrain V.* Generic-case complexity, decision problems in group theory and random walks. J. Algebra, 2003, vol. 264, no. 2, pp. 665–694.

2. *Kapovich I., Miasnikov A., Schupp P., and Shpilrain V.* Average-case complexity for the word and membership problems in group theory. Adv. Math., 2005, vol. 190, pp. 343–359.
3. *Hamkins J. D. and Miasnikov A. G.* The halting problem is decidable on a set of asymptotic probability one. Notre Dame J. Formal Logic, 2006, vol. 47, no. 4, pp. 515–524.
4. *Gilman R., Miasnikov A. G., Myasnikov A. D., and Ushakov A.* Report on generic case complexity. Herald of Omsk University, 2007, Special Issue, pp. 103–110.
5. *Mao W.* Modern Cryptography: Theory and Practice. Prentice Hall PTR, 2003.
6. *Impagliazzo R. and Wigderson A.* P = BPP unless E has subexponential circuits: Derandomizing the XOR Lemma. Proc. 29th STOC, El Paso, ACM, 1997, pp. 220–229.
7. *Myasnikov A. and Rybalov A.* Generic complexity of undecidable problems. J. Symbolic Logic, 2008, vol. 73, no. 2, pp. 656–673.
8. *Rybalov A.* Generic complexity of presburger arithmetic. Theory Comput. Systems, 2010, vol. 46, no. 1, pp. 2–8.