УДК 510.52

О ГЕНЕРИЧЕСКОЙ СЛОЖНОСТИ ПРОБЛЕМЫ РАСПОЗНАВАНИЯ КВАДРАТИЧНЫХ ВЫЧЕТОВ[1]

А. Н. Рыбалов

Омский государственный университет им. Ф. М. Достоевского, г. Омск, 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. В данной работе изучается генерическая сложность классической проблемы распознавания квадратичных вычетов в группах вычетов. Доказывается, что её естественная подпроблема генерически трудноразрешима (то есть трудна для почти всех входов) при условии, что проблема распознавания квадратичных вычетов трудноразрешима в классическом смысле.

Ключевые слова: *генерическая сложность, квадратичный вычет, вероятностный алгоритм.*

DOI 10.17223/20710410/28/6

ON GENERIC COMPLEXITY OF THE QUADRATIC RESIDUOSITY PROBLEM

A. N. Rybalov

Omsk State University, Omsk, Russia

E-mail: alexander.rybalov@gmail.com

Generic-case approach to algorithmic problems was suggested by Myasnikov, Kapovich, Schupp and Shpilrain in 2003. This approach studies behavior 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. For example, this is the classical discrete logarithm problem. In this talk we consider generic complexity of the quadratic residuosity problem. We fit this problem in the frameworks of generic complexity and prove that its natural subproblem is generically hard provided that the quadratic residuosity problem is hard in the worst case.

Keywords: *generic complexity, quadratic residue, probabilistic algorithm.*

Введение

В работе [1] была развита теория генерической сложности вычислений. В рамках этого подхода алгоритмическая проблема рассматривается не на всем 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

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

быстрые алгоритмы для всех входов. Классическим примером такого алгоритма является 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. Многие классические NP-полные проблемы в генерическом случае оказываются легкоразрешимыми [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. Например, для проблемы дискретного логарифма такие результаты получены в [5].

В данной работе изучается генерическая сложность классической проблемы распознавания квадратичных вычетов в группах вычетов. Эта алгоритмическая проблема восходит ещё к Гауссу и является хорошо известной в криптографии (см., например, [6]). До сих пор не известно полиномиальных алгоритмов её решения. В данной работе доказывается, что эта проблема неразрешима за полиномиальное время на любых полиномиальных генерических множествах входов при условии отсутствия полиномиальных вероятностных алгоритмов 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 гипотеза пока не доказана, имеются серьёзные доводы в её пользу [7].

При доказательстве основного результата использованы методы, развитые в [8, 9].

1. Генерические и пренебрежимые множества

Пусть I — некоторое множество входов. На множестве I определена функция размера $\mathrm{size} : I \to \mathbb{N}$, сопоставляющая каждому элементу $a \in I$ его размер $\mathrm{size}(a)$. Допустим, что для любого n множество I_n элементов из I размера n конечно. Для любого подмножества $S \subseteq I$ определим следующую последовательность:

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

Величина $\rho_n(S)$ — это вероятность получить вход из множества S при случайной и равномерной генерации элементов из I_n. *Асимптотической плотностью* S назовём следующий 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. Понятие генерического множества является некоторой формализацией интуитивного понятия множества «почти всех» элементов множества I в том смысле, что при увеличении размера элемента вероятность попасть в генерическое множество при случайной и равновероятной генерации элементов стремится к 1.

Алгоритмическая проблема распознавания множества $S \subseteq I$ *генерически полиномиально разрешима*, если существует множество $G \subseteq I$, такое, что

1) G — генерическое;
2) G — разрешимое за полиномиальное время;
3) $G \cap S$ — разрешимое за полиномиальное время.

Генерический алгоритм, решающий проблему S, работает на входе $x \in I$ следующим образом. Сначала определяет, принадлежит ли x генерическому множеству G. Если да, то проверяет принадлежность входа S. Если нет, то отвечает НЕ ЗНАЮ. Такой алгоритм правильно решает проблему распознавания S на почти всех входах.

2. Проблема распознавания квадратичных вычетов

Пусть $\mathbb{Z}/(m)$ — мультипликативная группа вычетов по модулю $m \in \mathbb{N}$. Напомним, что квадратичным вычетом в группе $\mathbb{Z}/(m)$ называется любой элемент x, для которого существует $y \in \mathbb{Z}/(m)$, такой, что $x = y^2$. В противном случае элемент x называется квадратичным невычетом. Под проблемой распознавания квадратичных вычетов понимается проблема распознавания следующего множества:

$$QR = \{(m, x) \in \mathbb{N}^2 : m = pq, \text{ где } p, q \text{ — простые числа,}\\ x \text{ — квадратичный вычет в } \mathbb{Z}/(m)\}.$$

В настоящее время неизвестно полиномиальных алгоритмов (в том числе и вероятностных), решающих проблему распознавания квадратичных вычетов для всех таких модулей m. Более того, на предположении о её трудноразрешимости основаны некоторые криптографические алгоритмы [6].

Для изучения генерической сложности этой проблемы необходимо провести некоторую стратификацию на множестве входов. Рассмотрим любую бесконечную последовательность натуральных чисел $\mu = \{m_1, m_2, \ldots\}$, удовлетворяющую следующим условиям:

1) $2^n < m_n < 2^{n+1}$ для любого n;
2) $m_n = pq$, где p, q — различные простые числа, для любого $n > 1$.

Будем называть такую последовательность *экспоненциальной*. Из знаменитого постулата Бертрана, доказанного П. Л. Чебышевым, следует, что экспоненциальные последовательности существуют. Определим алгоритмическую проблему $QR(\mu)$ как ограничение проблемы распознавания квадратичных вычетов QR на следующее множество входных данных:

$$I = \{(m, x) : m \in \mu, x \in \mathbb{Z}/(m)\}.$$

Под размером входа (m, x) понимается количество бит в двоичной записи числа m минус 1. Заметим, что множество I_n входов проблемы $QR(\mu)$ размера n состоит из всех пар (m, x), где m — единственное число $m \in \mu$, удовлетворяющее условию $2^n < m < 2^{n+1}$, а x — любой элемент из $\mathbb{Z}/(m)$.

Таким образом, $QR(\mu)$ является подпроблемой проблемы QR. Тем не менее можно доказать, что среди проблем $QR(\mu)$ существуют проблемы такие же сложные, как и оригинальная проблема QR.

Лемма 1. Если не существует полиномиального вероятностного алгоритма для проблемы QR, то найдётся такая экспоненциальная последовательность μ, что и для проблемы $QR(\mu)$ не существует 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 проблемы QR, следует, что для любого алгоритма P_n существует бесконечно много групп $\mathbb{Z}/(m)$, в которых он не может решить QR. Из этого следует, что можно выбрать последовательность натуральных чисел $\mu' = \{m_1, m_2, \ldots\}$, являющихся произведениями двух простых, так, чтобы алгоритм P_n не решал QR в группе $\mathbb{Z}/(m_n)$ и для любого n выполнялось $m_{n+1} > 2m_n$. Последовательность μ' можно расширить до экспоненциальной последовательности μ, добавив, где нужно, новые члены. Заметим теперь, что $QR(\mu)$ и есть та проблема, для которой не существует полиномиального вероятностного алгоритма. ■

3. Основной результат

Теорема 1. Если проблема $QR(\mu)$ генерически полиномиально разрешима, то существует полиномиальный вероятностный алгоритм, решающий $QR(\mu)$ для всех входов.

Доказательство. Допустим, существует генерический полиномиальный алгоритм $\mathcal{A}$, разрешающий проблему $QR(\mu)$ на некотором полиномиальном генерическом множестве G. Построим вероятностный полиномиальный алгоритм $\mathcal{B}$, решающий $QR(\mu)$ на всём множестве входов. Алгоритм $\mathcal{B}$ на входе (m, x) работает следующим образом:

1) Проверяет, принадлежит ли (m, x) множеству G. Это делается за полиномиальное время, так как множество G разрешимо за полиномиальное время. Если $(m, x) \in G$, то с помощью алгоритма $\mathcal{A}$ определяет принадлежность множеству $QR(\mu)$. Если нет, переходит к шагу 2.
2) Генерирует случайно и равновероятно элемент $y \in \mathbb{Z}/(m)$. Вычисляет $z = xy^2$.
3) Проверяет, принадлежит ли (m, z) множеству G.
4) Если $(m, z) \in G$, то с помощью алгоритма $\mathcal{A}$ определяет, является ли z квадратичным вычетом. Очевидно, что $z = xy^2$ является квадратичным вычетом тогда и только тогда, когда таковым является x.
5) Если $(m, z) \notin G$, выдаёт ответ НЕТ.

Заметим, что полиномиальный вероятностный алгоритм $\mathcal{B}$ может выдать неправильный ответ только на шаге 5. Докажем, что вероятность этого меньше $1/2$. В группе $\mathbb{Z}/(m)$, где $m = pq$ с простыми p и q, ровно $1/4$ элементов являются квадратами, а потому $z = xy^2$ может равновероятно принимать $1/4$ значений в $\mathbb{Z}/(m)$. В то же время с ростом размера входа для более $3/4$ элементов $u \in \mathbb{Z}/(m)$ входы (m, u) попадают в генерическое множество G. Поэтому вероятность шага 5 стремится к нулю с ростом размера входа и становится меньше $1/2$. ■

Непосредственным следствием теоремы 1 является следующее утверждение.

Теорема 2. Если для проблемы QR не существует полиномиального вероятностного алгоритма, то существует экспоненциальная последовательность μ, такая, что проблема $QR(\mu)$ не является генерически полиномиально разрешимой.

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. *Blum M. and Micali S.* How to generate cryptographically strong sequences of pseudorandom bits // SIAM J. Comput. 1984. V. 13. No. 4. P. 850–864.
6. *Mao B.* Современная криптография: теория и практика. М.: Вильямс, 2005. 768 с.
7. *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.
8. *Myasnikov A. and Rybalov A.* Generic complexity of undecidable problems // J. Symbolic Logic. 2008. V. 73. No. 2. P. 656–673.
9. *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. *Blum M. and Micali S.* How to generate cryptographically strong sequences of pseudorandom bits. SIAM J. Comput., 1984, vol. 13, no. 4, pp. 850–864.
6. *Mao V.* Sovremennaya kriptografiya: teoriya i praktika [Modern Cryptography: Theory and Practice]. Moscow, Wil'yams Publ., 2005. 768 p. (in Russian)
7. *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.
8. *Myasnikov A. and Rybalov A.* Generic complexity of undecidable problems. J. Symbolic Logic, 2008, vol. 73, no. 2, pp. 656–673.
9. *Rybalov A.* Generic complexity of Presburger Arithmetic. Theory Comput. Systems, 2010, vol. 46, no. 1, pp. 2–8.