МАТЕМАТИЧЕСКИЕ ОСНОВЫ ИНФОРМАТИКИ И ПРОГРАММИРОВАНИЯ

УДК 510.52 DOI 10.17223/20710410/66/10

О ГЕНЕРИЧЕСКОЙ СЛОЖНОСТИ ПРОБЛЕМЫ ДИСКРЕТНОГО ЛОГАРИФМА В ПОСЛЕДОВАТЕЛЬНОСТЯХ ЛЮКА[1]

А. Н. Рыбалов

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

E-mail: alexander.rybalov@gmail.com

Изучается генерическая сложность проблемы дискретного логарифма в последовательностях Люка. Эта проблема была использована в 1990-е годы новозеландским криптографом П. Смитом для создания аналога классического протокола Диффи — Хеллмана, в котором возведение в степень по целому модулю заменяется на операцию сложения элементов последовательности Люка. Доказывается, что при условии трудноразрешимости проблемы дискретного логарифма в последовательностях Люка в худшем случае и P = BPP существует подпроблема этой проблемы, для которой нет полиномиального генерического алгоритма. Таким образом, обосновано применение данной алгоритмической проблемы в криптографии с открытым ключом, где важна генерическая трудноразрешимость, то есть трудноразрешимость для почти всех входов. Для доказательства используется метод генерической амплификации, который позволяет строить генерически трудные проблемы из проблем, трудных в худшем случае. Основным этапом этого метода является объединение эквивалентных входов в достаточно большие множества. Эквивалентность входов означает, что рассматриваемая проблема на них решается одинаково.

Ключевые слова: *генерическая сложность, последовательности Люка, дискретный логарифм.*

ON THE GENERIC COMPLEXITY OF THE DISCRETE LOGARITHM PROBLEM IN LUCAS SEQUENCES

A. N. Rybalov

Sobolev Institute of Mathematics, Omsk, Russia

In this paper, we study the generic complexity of the discrete logarithm problem over Lucas sequences. This problem was exploited in the 1990s by the New Zealand cryptographer P. Smith to create an analogue of the classic Diffie — Hellman protocol, in which exponentiation modulo an integer is replaced by the operation of adding the elements of the Lucas sequence. It is proved that, given the worst-case intractability of the discrete logarithm problem in Lucas sequences and P = BPP, there exists a subproblem of this problem for which there is no polynomial generic algorithm.

[1]Работа поддержана грантом Российского научного фонда № 22-11-20019.

Thus, this result justifies the application of this algorithmic problem to public-key cryptography, where generic hardness is very important, i.e., hardness for almost all inputs. To prove the theorem, we use the method of generic amplification, which allows us to construct generically hard problems from problems that are hard in the classical sense. The main component of this method is the cloning technique, which combines the input data of a problem into sufficiently large sets of equivalent input data. Equivalence is understood in the sense that the problem is solved similarly for them.

Keywords: *generic complexity, Lucas sequences, discrete logarithm.*

Введение

Последовательности Люка — это линейные рекуррентные последовательности целых чисел второго порядка, являющиеся обобщением классической последовательности чисел Фибоначчи. В 1990-е годы новозеландский криптограф П. Смит предложил использовать последовательности Люка для создания криптосистемы с открытым ключом [1], являющейся аналогом знаменитой системы RSA, а также для создания аналога протокола Диффи — Хеллмана [2], основанного на проблеме дискретного логарифма в этих последовательностях. Основной чертой этих алгоритмов является использование операции сложения элементов последовательностей Люка вместо возведения в степень по целому модулю, как в классических алгоритмах RSA и Диффи — Хеллмана. Впоследствии проводился криптоанализ предложенных систем [3], однако они до сих пор считаются стойкими.

В современной криптографии важно, чтобы алгоритмическая проблема, лежащая в основе стойкости криптосистемы с открытым ключом, являясь (гипотетически) трудной в классическом смысле, оставалась трудной и в генерическом смысле [4], т. е. для почти всех входов. Это объясняется тем, что при случайной генерации ключей в криптографическом алгоритме происходит генерация входа алгоритмической проблемы, лежащей в основе алгоритма. Если эта проблема будет легкоразрешимой почти всегда, то для почти всех таких входов её можно будет быстро решить и ключи почти всегда будут нестойкими. Поэтому проблема должна быть трудной для почти всех входов. Например, таким поведением обладают классические алгоритмические проблемы криптографии: проблема распознавания квадратичных вычетов [5], проблема дискретного логарифма [6], проблема извлечения корня в группах вычетов [7] (обращения функции RSA).

В данной работе изучается генерическая сложность проблемы дискретного логарифма в последовательностях Люка. Доказывается, что при условии её трудноразрешимости в худшем случае и P = BPP можно выделить подпроблему этой проблемы, для которой не существует полиномиального генерического алгоритма. Класс BPP состоит из проблем, разрешимых за полиномиальное время на вероятностных машинах Тьюринга. Считается, что класс BPP совпадает с классом P, то есть любой полиномиальный вероятностный алгоритм можно эффективно дерандомизировать, построив полиномиальный алгоритм, не использующий генератор случайных чисел и решающий ту же самую проблему. Хотя равенство P = BPP до сих пор не доказано, имеются веские основания в его пользу [8].

1. Предварительные сведения

Пусть I — некоторое множество входов. Для подмножества $S \subseteq I$ определим *последовательность относительных плотностей*

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

где I_n — множество входов размера n; $S_n = S \cap I_n$. *Асимптотической плотностью* множества S назовём верхний предел

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

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 $\{x \in I : \mathcal{A}(x) \neq ?\}$ является генерическим.

Генерический алгоритм $\mathcal{A}$ вычисляет функцию $f : I \to I$, если для всех $x \in I$

$$\Big(\mathcal{A}(x) \neq ?\Big) \Rightarrow \Big(\mathcal{A}(x) = f(x)\Big).$$

Напомним некоторые понятия классической теории сложности вычислений. *Время работы* $t_M(x)$ машины Тьюринга M на входе $x \in I$ — это число шагов машины от начала работы до остановки. Машина Тьюринга M *полиномиальна*, если существует полином $p(n)$, такой, что для любого $x \in I$ имеет место $t_M(x) < p(\mathrm{size}(x))$. Класс P состоит из подмножеств I, распознаваемых полиномиальными машинами Тьюринга.

Вероятностная машина Тьюринга — это машина Тьюринга, в программе которой допускаются пары недетерминированных правил, которые одновременно применимы в данной ситуации. В процессе работы такой машины с вероятностью 1/2 выбирается первое правило и с вероятностью 1/2 — второе. Время работы $t_M(x, \tau)$ вероятностной машины Тьюринга на входе x зависит от вычислительного пути (последовательности выполненных команд) τ. Вероятностная машина Тьюринга M называется *полиномиальной*, если существует полином $p(n)$, такой, что для любого x и для любого вычислительного пути τ машины M на x имеет место $t_M(x, \tau) < p(\mathrm{size}(x))$.

Обозначим через $\Pr[M(x) = y]$ вероятность того, что машина M на входе x выдаёт ответ y. Вероятностная машина M *вычисляет* функцию $f : I \to I$, если для любого $x \in I$ выполняется

$$\big(f(x) = y\big) \Rightarrow \Pr[M(x) = y] > 2/3.$$

Проблема распознавания множества $S \subseteq I$ принадлежит классу BPP, если существует вероятностная полиномиальная машина Тьюринга M, вычисляющая характеристическую функцию множества S:

$$\chi_S(x) = \begin{cases} 1, & \text{если } x \in S, \\ 0, & \text{если } x \notin S. \end{cases}$$

Вероятностные машины Тьюринга формализуют понятие алгоритма, использующего генератор случайных чисел. Класс BPP — это класс проблем, эффективно решаемых такими вероятностными алгоритмами.

2. Последовательности Люка и проблема дискретного логарифма

Последовательность Люка — это рекуррентная последовательность пар целых чисел $\big(U_n(P,Q), V_n(P,Q)\big)$, определяемая следующим образом:

$$\begin{gathered} U_0(P,Q) = 0, \quad U_1(P,Q) = 1, \\ V_0(P,Q) = 2, \quad V_1(P,Q) = P, \\ U_{n+2}(P,Q) = P \cdot U_{n+1}(P,Q) - Q \cdot U_n(P,Q), \quad n \geqslant 0, \\ V_{n+2}(P,Q) = P \cdot V_{n+1}(P,Q) - Q \cdot V_n(P,Q), \quad n \geqslant 0. \end{gathered}$$

Числа Фибоначчи являются частным случаем последовательности Люка $\{U_n(1,-1)\}$. Последовательности Люка обладают массой интересных свойств [3]. Перечислим те из них, которые нам понадобятся в дальнейшем. Пусть p — простое число, тогда для любых n, P, Q имеет место

$$V_n(P \bmod p, Q \bmod p) = V_n(P,Q) \bmod p.$$

Для любых n, k, P, Q выполняется

$$V_{nk}(P,Q) = V_n(V_k(P,Q), Q^k). \tag{1}$$

По простому модулю p последовательность $\{V_n(P,Q) : n = 1, \ldots\}$ периодична с некоторым периодом, являющимся делителем числа $p^2 - 1$. Другими словами, имеет место

$$V_n(P,Q) \bmod p = V_{n \bmod (p^2-1)}(P,Q) \bmod p.$$

Опишем реализацию протокола Диффи — Хеллмана (LUCDIF) с помощью последовательностей Люка [2]:

1) Сначала Алиса выбирает простое число p, число $g < p$ и секретное число $a < p$.
2) Затем Алиса вычисляет $V_a(g,1) \bmod p$.
3) После этого Алиса отправляет Бобу сообщение $(V_a(g,1) \bmod p, p, g)$.
4) Боб выбирает своё секретное число $b < p$. Используя его, он получает общий секретный ключ $V_b(V_a(g,1),1) \bmod p$.
5) Затем Боб отправляет Алисе сообщение $V_b(g,1) \bmod p$.
6) Алиса, в свою очередь, вычисляет общий секретный ключ $V_a(V_b(g,1),1) \bmod p$.

Из свойства (1) следует, что общий ключ у них будет один и тот же:

$$V_b(V_a(g,1),1) = V_{ab}(g,1) = V_a(V_b(g,1),1).$$

Криптостойкость этого протокола базируется на сложности проблемы дискретного логарифма для последовательности Люка $\{V_n(g,1) : n \in \mathbb{N}\}$. Опишем эту проблему.

Пусть $L(g,p)$ — множество всех чисел Люка $V_n(g,1)$ по модулю p. *Проблема дискретного логарифма для последовательности Люка* состоит в вычислении функции $\mathrm{dll} : I \to \mathbb{N}$, где I — множество троек (a,g,p), таких, что p — фиксированное простое число, g — фиксированное натуральное число $g < p$, a — произвольное число из $L(g,p)$. Функция dll определяется следующим образом:

$$\mathrm{dll}(a,g,p) = x \;\Leftrightarrow\; a = V_x(g,p).$$

Под размером входа понимается число разрядов в двоичной записи числа p. В настоящее время неизвестно [3] полиномиальных алгоритмов (даже вероятностных) для вычисления функции dll.

Для изучения генерической сложности этой проблемы необходимо провести 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. Будем называть такую последовательность *экспоненциальной*. Определим функцию dll_π как ограничение функции dll на множество троек (a, g, p), таких, что $p \in \pi$. Для этой функции множество всех входов размера n состоит из троек (a, g, p) с фиксированными g, p и произвольным $a \in L(g, p)$. Очевидно, что проблема вычисления dll_π является подпроблемой вычисления dll. Следующая лемма показывает, что некоторые такие подпроблемы так же вычислительно трудны, как и оригинальная проблема.

Лемма 1. Если не существует полиномиального вероятностного алгоритма для вычисления функции dll, то найдётся такая экспоненциальная последовательность простых чисел π, что и для вычисления функции dll_π нет полиномиального вероятностного алгоритма.

Доказательство. Пусть $P_1, P_2, \ldots$ — все полиномиальные вероятностные алгоритмы. Из предположения о несуществовании полиномиального вероятностного алгоритма для вычисления dll следует, что для любого алгоритма P_n найдётся бесконечно много троек (a, g, p), для которых он не может вычислить dll. Тогда можно выбрать последовательность $\pi' = \{p_1, p_2, \ldots\}$ так, чтобы алгоритм P_n не вычислял dll для (a_n, g_n, p_n) и для любого n выполнялось бы $p_{n+1} > 2p_n$, и расширить π' до экспоненциальной последовательности π, добавив, где нужно, новые члены. Заметим, что для вычисления функции dll_π не существует полиномиального алгоритма. ■

3. Основной результат

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

Доказательство. Пусть полиномиальный генерический алгоритм $\mathcal{A}$ вычисляет функцию dll_π. Построим вероятностный полиномиальный алгоритм $\mathcal{B}$, вычисляющий dll_π на всём множестве входов. Алгоритм $\mathcal{B}$ на входе (a, g, p) размера n работает следующим образом:

1) Повторяет n раз:
2) генерирует случайно и равномерно $b \in \{1, \ldots, p^2 - 2\}$;
3) с помощью алгоритма Евклида вычисляет $(b, p^2 - 1)$;
4) если $(b, p^2 - 1) = 1$, переходит на шаг 7;
5) возвращается на шаг 2;
6) если после n попыток не получено b, взаимно простое с $p^2 - 1$, выдаёт 0;
7) вычисляет $a' = V_b(a, 1) \bmod p$;
8) запускает алгоритм $\mathcal{A}$ на (a', g, p);
9) если $\mathcal{A}(a', g, p) = y \in \mathbb{N}$, то по свойству (1)

$$a' = V_b(a, 1) \bmod p = V_b(V_x(g, 1), 1) \bmod p = V_{bx}(g, 1) \bmod p.$$

Значит, $y = bx \pmod{(p^2 - 1)}$, откуда $x = yb^{-1} \bmod (p^2 - 1)$ находится эффективно с помощью расширенного алгоритма Евклида;

10) если $\mathcal{A}(a', g, p) = ?$, то выдаёт 0.

Алгоритм $\mathcal{B}$ может выдать неправильный ответ на шаге 6 или шаге 10. Докажем, что вероятность этого меньше $1/2$.

Вероятность выдать ответ на шаге 6 равна

$$\Big(1-\frac{\varphi(p^2-1)}{p^2-3}\Big)^n < \Big(1-\frac{C}{\log n}\Big)^n < e^{-Cn/\log n} < 1/4$$

для достаточно больших n. Здесь $\varphi(x)$ — функция Эйлера (количество натуральных чисел, меньших x и взаимно простых с x); использована следующая оценка для функции Эйлера [9]:

$$\varphi(x) > \frac{Cx}{\log\log x}$$

для некоторой константы $C > 0$.

Оценим вероятность выдать ответ на шаге 10. Значение $a' = V_b(a,1) \bmod p = V_{bx \bmod (p^2-1)}(g,1) \bmod p$ пробегает все элементы $L(g,p)$, так как $bx \bmod (p^2-1)$ при $b \in \{1,\dots,p^2-2\}$, таких, что $(b, p^2-1) = 1$, принимает все значения из множества $\{1,\dots,p-1\}$. Поэтому множество

$$\big\{(a',g,p) : b \in \{1,\dots,p^2-2\}\big\}$$

совпадает с множеством всех входов размера n. Но алгоритм $\mathcal{A}$ генерический, поэтому доля тех входов (a',g,p), на которых он выдаёт неопределённый ответ, стремится к нулю с ростом n и с некоторого момента становится меньше $1/4$. ■

Непосредственным следствием теоремы 1 является следующая

Теорема 2. Если для вычисления функции dll не существует полиномиального вероятностного алгоритма, то существует экспоненциальная последовательность простых чисел π, такая, что для вычисления функции dll_π не существует генерического полиномиального алгоритма.

Автор выражает благодарность рецензенту за полезные замечания и предложения по улучшению текста статьи.

ЛИТЕРАТУРА

1. *Smith P. J. and Lennon M. J. J.* LUC: a new public key system // Proc. IFIP/Sec'93. Toronto, Canada, 1993. P. 103–117.
2. *Smith P. and Skinner C.* A public-key cryptosystem and a digital signature system based on the Lucas function analogue to discrete logarithms // LNCS. 1995. V. 917. P. 355–364.
3. *Bleichenbacher D., Bosma W., and Lenstra A.* Some remarks on Lucas-based cryptosystems // LNCS. 1995. V. 963. P. 386–396.
4. *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.
5. *Рыбалов А. Н.* О генерической сложности проблемы распознавания квадратичных вычетов // Прикладная дискретная математика. 2015. № 2 (28). С. 54–58.
6. *Рыбалов А. Н.* О генерической сложности проблемы дискретного логарифма // Прикладная дискретная математика. 2016. № 3 (33). С. 93–97.
7. *Рыбалов А. Н.* О генерической сложности проблемы извлечения корня в группах вычетов // Прикладная дискретная математика. 2017. № 38. С. 95–100.
8. *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.
9. *Rosser J. B. and Schoenfeld L.* Approximate formulas for some functions of prime numbers // Illinois J. Math. 1962. V. 6. No. 1. P. 64–94.

REFERENCES

1. *Smith P. J. and Lennon M. J. J.* LUC: a new public key system. Proc. IFIP/Sec'93, Toronto, Canada, 1993, pp 103–117.
2. *Smith P. and Skinner C.* A public-key cryptosystem and a digital signature system based on the Lucas function analogue to discrete logarithms. LNCS, 1995, vol. 917, pp. 355–364.
3. *Bleichenbacher D., Bosma W., and Lenstra A.* Some remarks on Lucas-based cryptosystems. LNCS, 1995, vol. 963, pp. 386–396.
4. *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.
5. *Rybalov A. N.* O genericheskoy slozhnosti problemy raspoznavaniya kvadratichnykh vychetov [On generic complexity of the quadratic residuosity problem]. Prikladnaya Diskretnaya Matematika, 2015, no. 2 (28), pp. 54–58. (in Russian)
6. *Rybalov A. N.* O genericheskoy slozhnosti problemy diskretnogo logarifma [On generic complexity of the discrete logarithm problem]. Prikladnaya Diskretnaya Matematika, 2016, no. 3 (33), pp. 93–97. (in Russian)
7. *Rybalov A. N.* O genericheskoy slozhnosti problemy izvlecheniya kornya v gruppakh vychetov [On generic complexity of the problem of finding roots in groups of residues]. Prikladnaya Diskretnaya Matematika, 2017, no. 38, pp. 95–100. (in Russian)
8. *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.
9. *Rosser J.B. and Schoenfeld L.* Approximate formulas for some functions of prime numbers. Illinois J. Math., 1962, vol. 6, no. 1, pp. 64–94.