МАТЕМАТИЧЕСКИЕ ОСНОВЫ ИНФОРМАТИКИ И ПРОГРАММИРОВАНИЯ

УДК 510.52 DOI 10.17223/20710410/63/7

ГЕНЕРИЧЕСКИ НЕРАЗРЕШИМЫЕ И ТРУДНОРАЗРЕШИМЫЕ ПРОБЛЕМЫ[1]

А. Н. Рыбалов

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

E-mail: alexander.rybalov@gmail.com

Памяти Виталия Анатольевича Романькова

В рамках генерического подхода изучается поведение алгоритмов на типичных (почти всех) входах, а остальные входы игнорируются. А. Мясниковым и автором ранее был предложен метод генерической амплификации для построения генерически неразрешимых алгоритмических проблем. Основной идеей этого метода является объединение эквивалентных входов в достаточно большие множества. Эквивалентность входов означает, что рассматриваемая проблема на них решается одинаково. Предлагается обобщение этого метода, строится пример разрешимой в классическом смысле проблемы, не являющейся генерически разрешимой за полиномиальное время. Для этого используются другие методы, так как, скорее всего, метод генерической амплификации здесь не работает.

Ключевые слова: *генерическая сложность, амплификация, алгоритмическая проблема.*

GENERICALLY UNDECIDABLE AND HARD PROBLEMS

A. N. Rybalov

Sobolev Institute of Mathematics, Omsk, Russia

The generic-case approach to algorithmic problems examines the behavior of an algorithm on typical (almost all) inputs and ignores the rest of the inputs. The method of generic amplification was proposed by A. Myasnikov and author for constructing of generically undecidable problems. The main ingredient 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 in the same way for them. We present a generalization of this method. We also construct a problem that is decidable in the classical sense, but which is not generically decidable in polynomial time. We use a different method to generic amplification, because generic amplification is unlikely to be applicable here.

Keywords: *generic complexity, amplyfication, algorithmic problems.*

[1]Работа выполнена в рамках госзадания ИМ СО РАН, проект FWNF-2022-0003.

Введение

Генерический подход [1] — это один из подходов к изучению алгоритмических проблем для «почти всех» входов. Исследования вычислительной сложности для «почти всех» входов начались в 1970–80-х гг., после того как был выделен огромный пласт трудноразрешимых алгоритмических проблем — NP-полных проблем, для которых не удалось найти эффективных алгоритмов, работающих за полиномиальное время для всех входов. Оказалось, что если немного ослабить требование эффективности — рассматривать не все входы, а «почти все» или случайные входы, то иногда можно быстро решать задачу для таких типичных входов. Это имеет практический смысл, когда алгоритм должен решать задачу для случайных входных данных: если вероятность «наткнуться» на «плохой» вход пренебрежимо мала, то алгоритм будет быстро работать практически всегда. В рамках генерического подхода изучается поведение алгоритмов на множестве «почти всех» входов (это множество называется генерическим) и игнорируется его поведение на остальных входах, на которых алгоритм может работать медленно или вообще не останавливаться. Понятие «почти все» формализуется введением асимптотической плотности на множестве входных данных.

В исследованиях по генерической вычислимости и сложности вычислений можно выделить два основных направления. Первое связано с построением генерических (полиномиальных) алгоритмов для алгоритмических проблем, которые являются неразрешимыми или трудноразрешимыми в классическом смысле. Второе направление концентрируется на поиске алгоритмических проблем, которые остаются неразрешимыми или трудноразрешимыми и в генерическом смысле. Данная работа относится ко второму направлению исследований.

Первые генерически неразрешимые алгоритмические проблемы были найдены А. Г. Мясниковым и А. Н. Рыбаловым в [2]. Для доказательства генерической неразрешимости предложен метод генерической амплификации, который позволяет по проблеме, неразрешимой в классическом смысле, строить проблему, которая генерически неразрешима. Данный метод успешно применён к следующим алгоритмическим проблемам: проблема остановки для машин Тьюринга [3], проблема равенства для полугрупп [2], проблема разрешимости элементарных теорий [2, 4], десятая проблема Гильберта [5]. Однако формализация метода, предложенная в [2], оказалась не совсем удобной: напрямую её удается применить только для построения конечно определённой полугруппы с генерически неразрешимой проблемой равенства, а в остальных случаях генерическая амплификация используется неформально. В данной работе предлагается формализация более общей схемы генерической амплификации, которая работает во всех случаях.

Применение генерической амплификации для построения генерически трудноразрешимых проблем сталкивается с трудностями, которые связаны с необходимостью контролировать скорость сходимости последовательности частот множества «плохих» входов. Поэтому тут удается получить лишь результаты об отсутствии сильно генерических полиномиальных алгоритмов, которые решают проблему быстро на множестве входов, относительные частоты которых экспоненциально быстро стремятся к единице. Например, в таком виде генерическая амплификация применима для арифметики Пресбургера [6]. В данной работе с помощью других методов строится пример разрешимой в классическом смысле проблемы, для которой не существует полиномиального генерического алгоритма.

1. Предварительные сведения

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

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

где I_n — множество входов размера n, а $S_n = S \cap I_n$ — множество входов из S размера n. Здесь для конечного множества A через $|A|$ обозначено число его элементов. *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 $\rho(S) = 0$. Назовём множество S *сильно пренебрежимым*, если последовательность $\rho_n(S)$ экспоненциально быстро сходится к нулю, т. е. существуют константы σ, $0 < \sigma < 1$, и $C > 0$, такие, что для любого n

$$\rho_n(S) < C\sigma^n.$$

Теперь S называется *сильно генерическим*, если его дополнение $I \setminus S$ сильно пренебрежимо.

Алгоритм $\mathcal{A}$ с множеством входов I называется (*сильно*) *генерическим*, если множество $\{x \in I : \mathcal{A}(x)\downarrow\}$ (сильно) генерическое. Здесь $\mathcal{A}(x)\downarrow$ означает, что алгоритм $\mathcal{A}$ останавливается на входе x. Генерический алгоритм $\mathcal{A}$ вычисляет функцию $f : I \to J$, если

$$\forall x \in I\ \big(\mathcal{A}(x)\downarrow \Rightarrow\ f(x) = \mathcal{A}(x)\big).$$

Генерический алгоритм $\mathcal{A}$ работает за полиномиальное время, если существует полином $p(n)$, такой, что

$$\forall x \in I\ \big(\mathcal{A}(x)\downarrow \Rightarrow t_{\mathcal{A}}(x) < p(\mathrm{size}(x))\big),$$

где $\mathrm{size}(x)$ — размер входа x; а $t_{\mathcal{A}}(x)$ — время работы алгоритма $\mathcal{A}$ на входе x. Такие алгоритмы будем называть полиномиальными генерическими.

С практической точки зрения, когда требуется построить алгоритм, решающий конкретную алгоритмическую проблему для почти всех входов, удобнее рассматривать алгоритмы следующего типа [7]. Каждый такой алгоритм останавливается на всех входах, на входах из некоторого генерического множества выдает правильный ответ, а на пренебрежимом множестве остальных входов выдает специальный ответ «?» — «Не знаю».

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

1) $\forall x \in I\ \mathcal{A}(x)\downarrow$;
2) множество $\{x \in I : \mathcal{A}(x) = ?\}$ (сильно) пренебрежимо.

Эффективно генерический алгоритм $\mathcal{A}$ вычисляет функцию $f : I \to J$, если

$$\forall x \in I\ \big(\mathcal{A}(x) \neq ?\ \Rightarrow\ f(x) = \mathcal{A}(x)\big).$$

Множество $S \subseteq I$ и соответствующая проблема распознавания (S, I) (*эффективно*) (*сильно*) *генерически разрешимы* (*за полиномиальное время*), если существует (эффективно) (сильно) генерический (полиномиальный) алгоритм, вычисляющий характеристическую функцию S.

Легко видеть, что из эффективной генерической разрешимости следует генерическая разрешимость. Действительно, любой эффективный генерический алгоритм можно без труда переделать в генерический, заменив выдачу ответа «?» на бесконечное зацикливание. В обратную сторону это неверно — см., например, [8, теорема 2.22 и следствие 2.24]. Однако для полиномиальной сложности верно и обратное: из полиномиальной генерической разрешимости следует полиномиальная эффективная разрешимость. Действительно, если имеется полиномиальная оценка $p(n)$ на время работы генерического алгоритма в случае, когда он останавливается, то можно завести счётчик T числа шагов и если $T > p(n)$, обрывать вычисление и выдавать ответ «?», — в этом случае генерический алгоритм уже не остановится. Таким образом получается эффективно генерический полиномиальный алгоритм, решающий ту же проблему.

2. Генерическая амплификация

Опишем сначала схему, обобщающую конструкцию генерической амплификации, предложенную в [2].

Пусть I — множество входов. *Клонирование* множества I — это функция $C : I \to P(I)$, где $P(I)$ есть множество всех подмножеств множества I. Будем называть клонирование $C : I \to P(I)$ *эффективным*, если существует всюду определённая вычислимая функция $E : I \times \mathbb{N} \to I$, такая, что для любого $x \in I$

$$C(x) = \{E(x, 0), E(x, 1), \dots, \}.$$

Таким образом, с помощью алгоритма E можно эффективно перечислять все элементы каждого клона $C(x)$. Клонирование $C : I \to P(I)$ называется *непренебрежимым* (*не сильно пренебрежимым*), если для любого $x \in I$ множество $C(x)$ не является пренебрежимым (сильно пренебрежимым).

Пусть $S \subseteq I$. Будем говорить, что клонирование $C : I \to P(I)$ *сохраняет* множество S, если:

1) $\forall x \in S \big(C(x) \subseteq S\big)$;
2) $\forall x \notin S \big(C(x) \subseteq I \setminus S\big)$.

Теорема 1. Пусть I — множество входов, $S \subseteq I$ и $C : I \to P(I)$ — эффективное клонирование, сохраняющее S. Тогда:

1) если C непренебрежимое клонирование и проблема распознавания (S, I) генерически разрешима, то проблема распознавания (S, I) разрешима;
2) если C не сильно пренебрежимое клонирование и проблема распознавания (S, I) сильно генерически разрешима, то проблема распознавания (S, I) разрешима.

Доказательство. Докажем п. 1. Пусть $\mathcal{A}$ — генерический алгоритм, распознающий S, такой, что множество

$$G(\mathcal{A}) = \{x \in I : \mathcal{A}(x) \downarrow\}$$

генерическое. Построим алгоритм $\mathcal{B}$, который решает проблему распознавания S для всех входов. На входе $x \in I$ алгоритм $\mathcal{B}$ работает следующим образом:

1) установить $i = 0$;
2) сделать $i + 1$ шагов вычисления алгоритма $\mathcal{A}$ на $E(x, 0)$, $E(x, 1)$, …, $E(x, i)$;
3) если алгоритм $\mathcal{A}$ остановился на каком-то $E(x, k)$, $k \leqslant i$, и выдал ответ, остановиться и выдать этот ответ;
4) иначе увеличить i на 1 и вернуться на шаг 2.

Так как $C(x)$ непренебрежимо, то $C(x)$ имеет непустое пересечение с множеством $G(\mathcal{A})$. Поэтому, параллельно запуская алгоритм $\mathcal{A}$ на элементах $E(x,0)$, $E(x,1)$, ..., мы найдём зависящее от x число i_x, такое, что $x' = E(x,i_x) \in G(\mathcal{A})$. Очевидно, $x \in S$ тогда и только тогда, когда $x' \in S$, и тогда и только тогда, когда $\mathcal{A}$ выдаёт ответ «ДА» для x'. Поэтому мы можем эффективно решать, выполнено ли $x \in S$, и п. 1 доказан.

Доказательство п. 2 аналогично. ■

Теорему 1 можно переформулировать в терминах генерической неразрешимости.

Теорема 2. Пусть I — множество входов, $S \subseteq I$ и $C : I \to P(I)$ — эффективное клонирование, сохраняющее S. Если проблема распознавания (S, I) неразрешима, то имеет место следующее:

1) если C — непренебрежимое клонирование, то проблема распознавания (S, I) не является генерически разрешимой;
2) если C — не сильно пренебрежимое клонирование, то проблема распознавания (S, I) не является сильно генерически разрешимой.

Перейдём к описанию метода генерической амплификации [2]. Назовём клонирование $C : I \to P(I)$ *разделяющим*, если

$$\forall x, y \in I \ (x \neq y) \Rightarrow C(x) \cap C(y) = \varnothing.$$

Для множества $S \subseteq I$ определим *клон* $C(S)$ как объединение всех клонов элементов из S:

$$C(S) = \bigcup_{x \in S} C(x).$$

Легко видеть, что для любого $S \subseteq I$ разделяющее клонирование $C : I \to P(I)$ сохраняет множество $C(S)$. Поэтому непосредственным следствием из теоремы 2 является следующее утверждение:

Теорема 3 [2]. Пусть I — множество входов, $S \subseteq I$ и $C : I \to P(I)$ — эффективное разделяющее клонирование. Тогда если проблема распознавания (S, I) неразрешима, то имеет место следующее:

1) если C — непренебрежимое клонирование, то проблема распознавания $(C(S), I)$ не является генерически разрешимой;
2) если C — не сильно пренебрежимое клонирование, то проблема распознавания $(C(S), I)$ не является сильно генерически разрешимой.

Отметим, что в конкретных ситуациях клонирование редко получается разделяющим. Например, в доказательствах генерической неразрешимости проблемы остановки для нормализованных машин Тьюринга [3] и теорий первого порядка для нормализованных формул [4] клоны для различных элементов пересекаются. Однако соответствующие клонирования сохраняют рассматриваемые множества, а потому, по теореме 2, эти проблемы не являются генерически разрешимыми.

3. Генерически трудноразрешимые проблемы

Рассмотрим машины Тьюринга, которые распознают подмножества двоичных строк из $\{0,1\}^*$. Такие машины имеют два завершающих состояния: q_a — допускающее и q_r — отвергающее. Заканчивать работу они могут только в одном из этих состояний. Таким образом, эти машины выдают только ответы «ДА» или «НЕТ». Под размером строки w понимается её длина $|w|$, поэтому число входов размера n равно 2^n.

Под *эффективной нумерацией всех полиномиальных машин Тьюринга* будем подразумевать эффективную нумерацию всех пар $\{(M_i, p_k(n)) : i \in \mathbb{N},\ k \in \mathbb{N}\}$, где M_i — машина Тьюринга с номером i; $p_k(n) = n^k + k$. Такая пара на входе x моделирует работу некоторой полиномиальной машины Тьюринга следующим образом:

$$(M_i, p_k(n))(x) = \begin{cases} M_i(x), & \text{если } M_i(x)\downarrow \text{ за } \leqslant p_k(|x|) \text{ шагов,} \\ \text{НЕТ} & \text{иначе.} \end{cases}$$

Ясно, что любая полиномиальная машина Тьюринга встретится в этой последовательности.

Теорема 4. Существует рекурсивное множество, не являющееся генерически разрешимым за полиномиальное время.

Доказательство. Пусть есть эффективная нумерация полиномиальных машин Тьюринга $P_1, P_2, P_3, \ldots$ Образуем из них следующую последовательность:

$$\{M_i, i = 1, 2, 3, \ldots\} = \{P_1, P_1, P_2, P_1, P_2, P_3, P_1, P_2, P_3, P_4, P_1, P_2, \ldots\}.$$

В ней каждый раз, после того как были выписаны машины $P_1, \ldots, P_k$, выписываются машины $P_1, \ldots, P_{k+1}$. Таким образом, каждая полиномиальная машина P_i выписывается бесконечно много раз.

Построение нужного множества S будет проходить по шагам. Стартуя с множества всех двоичных строк $\{0,1\}^*$ на нулевом шаге, мы будем на шаге i вычеркивать или оставлять некоторые числа в зависимости от поведения машины M_i. Опишем подробно шаг $i > 0$. Запускаем машину M_i на каждом входе размера i и считаем количество ответов «ДА» и «НЕТ». Если ответов «ДА» получилось больше половины, то вычёркиваем все входы размера i, иначе все их оставляем. Предельное множество в этом процессе и есть искомое множество S.

Действительно, заметим, что для любой полиномиальной машины M множество входов, на которых M дает неправильный ответ, имеет вид

$$E(M) = \bigcup_{i=1}^{\infty} A_i,$$

где $A_i = \{w \in \{0,1\}^* : |w| = m_i\}$, $m_i > m_{i-1}$, причём $|A_i| \geqslant 2^{m_i}/2$ для любого i. Если теперь рассмотреть последовательность

$$\rho_n(E(M)) = \frac{|E(M)_n|}{2^n},\ n = 1, 2, 3, \ldots,$$

то легко видеть, что $\rho_n(E(M)) \geqslant 1/2$ для бесконечно большого числа значений n. Поэтому множество $E(M)$ непренебрежимо.

Допустим, что существует генерический полиномиальный алгоритм $\mathcal{A}$, распознающий множество S. Без ограничения общности можно считать, что $\mathcal{A}$ — полиномиальный эффективно генерический алгоритм. По нему легко получить полиномиальную машину M, которая на любом входе x работает следующим образом:

1) вычисляет $\mathcal{A}(x)$;
2) если $\mathcal{A}(x) = 1$, выдаёт 1;
3) если $\mathcal{A}(x) = 0$, выдаёт 0;
4) если $\mathcal{A}(x) = ?$, выдаёт 0.

Очевидно, что M, распознавая элементы S, ошибается на пренебрежимом множестве. Но это противоречит построению множества S.

Рекурсивность множества S следует из алгоритмической природы процедуры его построения. ■

Заключение

В работе предложена формализация схемы генерической амплификации, которая обобщает схему [2]. Новый метод работает во всех случаях, в которых напрямую не удаётся применить схему из [2]. Кроме того, с помощью других идей строится пример разрешимой в классическом смысле проблемы, для которой не существует полиномиального генерического алгоритма. В данном случае, по-видимому, метод генерической амплификации неприменим.

В качестве дальнейших направлений исследований представляет интерес связать понятие генерической разрешимости с мерой М. Громова, определяемой также с помощью асимптотической плотности. Особо интересны случаи, когда эта плотность отлична от нуля и единицы. В этом направлении Р. Гилманом, А. Г. Мясниковым и В. А. Романьковым получены очень интересные результаты о плотности разрешимых уравнений в свободных группах [9] и в нильпотентных группах [10].

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

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. *Myasnikov A. and Rybalov A.* Generic complexity of undecidable problems // J. Symbolic Logic. 2008. V. 73. No. 2. P. 656–673.
3. *Rybalov A.* On the generic undecidability of the Halting Problem for normalized Turing machines // Theory of Computing Systems. 2017. V. 60. No. 4. P. 671–676.
4. *Рыбалов А. Н.* О генерической сложности элементарных теорий // Вестник Омского университета. 2015. № 4. С. 14–17.
5. *Rybalov A.* Generic complexity of the Diophantine problem // Groups Complexity Cryptology. 2013. V. 5. No. 1. P. 25–30.
6. *Rybalov A.* Generic complexity of Presburger arithmetic // Theory of Computing Systems. 2010. V. 46. No. 1. P. 2–8.
7. *Hirschfeldt D.* Some questions in computable mathematics // LNCS. 2017. V. 10010. P. 22–55.
8. *Jockusch C. and Schupp P.* Generic computability, Turing degrees, and asymptotic density // J. London Math. Soc. 2012. V. 85. No. 2. P. 472–490.
9. *Gilman A., Myasnikov A., and Roman'kov V. A.* Random equations in free groups // Groups Complexity Cryptology. 2011. V. 3. No. 2. P. 257–284.
10. *Gilman A., Myasnikov A., and Roman'kov V. A.* Random equations in nilpotent groups // J. Algebra. 2012. V. 352. No. 1. P. 192–214.

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. *Myasnikov A. and Rybalov A.* Generic complexity of undecidable problems. J. Symbolic Logic, 2008, vol. 73, no. 2, pp. 656–673.
3. *Rybalov A.* On the generic undecidability of the Halting Problem for normalized Turing machines. Theory of Computing Systems, 2017, vol. 60, no. 4, pp. 671–676.
4. *Rybalov A. N.* O genericheskoy slozhnosti elementarnykh teoriy [On the generic complexity of elementary theories]. Vestnik OmSU, 2015, no. 4, pp. 14–17. (in Russian)
5. *Rybalov A.* Generic complexity of the Diophantine problem. Groups Complexity Cryptology, 2013, vol. 5, no. 1, pp. 25–30.

6. *Rybalov A.* Generic complexity of Presburger arithmetic. // Theory of Computing Systems, 2010. vol. 46, no. 1, pp. 2–8.
7. *Hirschfeldt D.* Some questions in computable mathematics. LNCS, 2017, vol. 10010, pp. 22–55.
8. *Jockusch C. and Schupp P.* Generic computability, Turing degrees, and asymptotic density. J. London Math. Soc., 2012, vol. 85, no. 2, pp. 472–490.
9. *Gilman A., Myasnikov A. and Roman'kov V. A.* Random equations in free groups. Groups Complexity Cryptology, 2011, vol. 3, no. 2, pp. 257–284.
10. *Gilman A., Myasnikov A. and Roman'kov V. A.* Random equations in nilpotent groups. J. Algebra, 2012, vol. 352, no. 1, pp. 192–214.