А. П. Бельтюков, Иерархии сложности вычисления частичных функций со значениями 0 и 1,
Матем. заметки, 1980, том 28, выпуск 3, 423–431

МАТЕМАТИЧЕСКИЕ ЗАМЕТКИ

т. 28, № 3 (1980)

ИЕРАРХИИ СЛОЖНОСТИ ВЫЧИСЛЕНИЯ ЧАСТИЧНЫХ ФУНКЦИЙ СО ЗНАЧЕНИЯМИ 0 И 1

А. П. Бельтюков

Для решения некоторых задач достаточно построить вычислительную процедуру, которая дает ответ лишь для некоторых из всех возможных наборов входных данных, расходуя при этом не более определенного количества какого-либо вычислительного ресурса. Например, для информационно-поисковой системы может понадобиться программа, которая быстро дает ответ на вопрос, содержится ли документ с данным свойством в системе. Хотя можно требовать, чтобы эта программа находила ошибки в предъявленном ей запросе, она, разумеется, не обязана проверять все наборы данных, которыми она пользуется, и может в случае их неправильности иногда давать какой-либо ответ, а иногда не давать никакого ответа в приемлемое время.

Математической моделью такой ситуации является задача построения для частичной функции алгоритма, вычисляющего некоторое (возможно, тоже частичное) ее продолжение. Причем, в тех точках, где эта функция определена, сложность работы этого алгоритма не должна превышать заданную границу сложности — функцию от длины входных данных.

В настоящей статье исследуется, как влияет увеличение верхней границы сложности на возможность решения вычислительными машинами таких задач. Другими словами, изучается, какое увеличение, например, времени работы или памяти машины ведет к увеличению класса вычисляемых частичных функций. Полученные результаты говорят о том, что этот класс расширяется и при таком

небольшом увеличении границы сложности, при котором множество всюду определенных функций этого класса остается неизменным, или изменение его не удается доказать известными методами (см. [1] — [4]). Результаты настоящей статьи имеют более общие условия, чем результаты работ [5] — [8]. Однако результаты этих работ не усиливаются здесь, так как их заключения говорят не о классах частичных функций, а о классах множеств, перечислимых некоторыми способами с заданной сложностью. Важно, что изучение сложности вычисления частичных функций гораздо более интересно, чем изучение сложности перечисления множеств.

Пусть Ξ — некоторый тип абстрактных вычислительных машин со входным алфавитом Σ, содержащим символы 0 и 1, Φ — некоторая мера сложности, удовлетворяющая аксиомам Блюма (см. [9] и [10]). Пусть натуральное число $\Phi(M, x)$ — сложность работы машины M на слове x. Пусть F — общерекурсивная функция.

Определение. Частичная словарная функция f (см. [10]) вычислима машиной M в пределах сложности F, если на любом достаточно длинном слове x из области определения f результат работы машины M определен и совпадает с $f(x)$, а сложность работы не превосходит $F(|x|)$, где $|x|$ — длина слова x. Множество всех частичных словарных функций со значениями 0 и 1, вычислимых машинами типа Ξ с мерой сложности Φ в пределах сложности F будем обозначать через $P(\Xi, \Phi, F)$. Когда зафиксированы тип машин Ξ и мера Φ, запись $P(\Xi, \Phi, F)$ будем сокращать до $P(F)$.

Заметим, что у каждой частичной функции из $P(F)$ есть продолжение из $P(F)$ с разрешимой областью определения. Если $F(n) \leqslant G(n)$ для всех достаточно больших n то $P(F) \subseteq P(G)$.

Будем рассматривать многоленточные машины Тьюринга (см. [1], [11]), первая лента которых имеет не менее двух головок. Алфавит Σ для записи входных слов содержит символы 0, 1 и не содержит пробела, принадлежащего ленточному алфавиту. В начале работы все головки на каждой ленте установлены в одну и ту же позицию. На первой ленте они обозревают ячейку, предшествующую первому символу записанного на этой ленте входного слова. Во всех ячейках не занятых входным словом, первоначально записан пробел.

Выберем произвольно и зафиксируем верхние границы числа лент, числа головок на каждой ленте и размера ленточного алфавита. Через Θ будем обозначать множество всех описанных выше машин, удовлетворяющих этим ограничениям.

Через Time (M, x) будем обозначать число шагов работы машины M на слове x, через Space (M, x) — общее число ленточных ячеек, используемых машиной M при работе на слове x. Число Time (M, x) соответствует времени работы, а Space (M, x) — объему памяти вычислительных машин.

ТЕОРЕМА. *Пусть* T_1, T_2, S_1, S_2 — *общерекурсивные функции. Если*

$$T_2/T_1 \to +\infty,\ \forall_n^\infty\ (T_1(n+1) + 6 \cdot n \leqslant T_2(n)),$$

то

$$P(\Theta, \text{Time}, T_1) \subsetneqq P(\Theta, \text{Time}, T_2).$$

Если

$$S_2 - S_1 \to +\infty,\ \forall_n^\infty\ (S_1(n+1) \leqslant S_2(n)),$$

то

$$P(\Theta, \text{Space}, S_1) \subsetneqq P(\Theta, \text{Space}, S_2).$$

Здесь и далее $\forall_n^\infty$ — сокращение для $\exists n_0 \forall n \geqslant n_0$.

Из этой теоремы, например, следует, что для любой общерекурсивной неубывающей неограниченной f и для любого $c \geqslant 1$

(а) $P(\Theta, \text{Time}, \lambda n.n^c) \subsetneqq P(\Theta, \text{Time}, \lambda n.n^c \cdot f(n))$;

(б) $P(\Theta, \text{Space}, \lambda n.c \cdot n) \subsetneqq P(\Theta, \text{Space}, \lambda n.c \cdot n + f(n))$;

где $\lambda n.t$ — функция со значением t в точке n.

Пусть $R(\Theta, \Phi, F)$ — обозначение для класса всех функций из $P(\Theta, \Phi, F)$, определенных на всех словах алфавита Σ. Предложения, полученные из двух утверждений теоремы заменой P на R, не верны в силу результата [12]. Не известно, истинны ли предложения, получающиеся заменой в (а) и (б) оператора P на R. Методы доказательства работ [1] — [4] дают более слабые результаты, чем эти предложения.

Первоначально основной результат получим в общем виде для машин произвольного типа Σ и произвольной меры сложности Φ. Для доказательства будут использованы методы, близкие к методам работ [13], [5] — [7].

Запись $!s$ будет означать, что выражение s определено; запись $s \leftarrow t$ будет означать: если $!t$, то во-первых, $!s$ и, во-вторых, $s = t$; запись $s \simeq t$ будет означать: $s \leftarrow t$ и $t \leftarrow s$. Через x^α будем обозначать слово $xx \ldots x$ (α раз). Через $M\,[F]$ будем обозначать следующую частичную функцию:

$$M\,[F]\,(x) \simeq \begin{cases} M\,(x), \text{ если } \Phi\,(M, x) \leqslant F\,(|\,x\,|); \\ \text{неопределено иначе.} \end{cases}$$

Знак $\leftrightharpoons$ далее заменяет слова: «есть сокращение для».

ОСНОВНАЯ ЛЕММА. *Пусть зафиксированы тип машин* Ξ *и мера сложности* Φ. *Пусть общерекурсивные функции* F *и* G *удовлетворяют следующим условиям*:

(1) $\forall_n^\infty\, F\,(n) \leqslant G\,(n)$;

(2) *существует такая машина* U, *что для любой машины* M *найдется натуральное число* α *такое, что для всех достаточно длинных слов* w *вида* $0^\alpha 1x$ *имеет место*

$$U\,[G]\,(w) \leftarrow M\,[F]\,(w);$$

(3) *для любых частично рекурсивной словарной функции* g *со значениями* 0, 1 *и машины* M *существуют такие частичная функция* h (*с той же областью определения, что и* g) *и машина* B, *что для всех достаточно длинных слов* y *при любом натуральном* β

$$B\,[G]\,(w) \leftarrow \begin{cases} g\,(y), \textit{ если } !\,h\,(y) \textit{ и } h\,(y) \leqslant \beta, \\ M\,[F]\,(w0) \textit{ иначе}, \end{cases}$$

где $w \leftrightharpoons y\ 1\ 0^\beta$.

Тогда $P\,(F) \subsetneqq P\,(G)$.

Доказательство. Из (1) следует, что $P\,(F) \subseteq P\,(G)$. Пусть

$$P\,(F) = P\,(G). \tag{4}$$

По условию (2) существует такая машина U, что для любой машины M, вычисляющей функцию f из $P\,(F)$ в пределах сложности F, при некотором α для достаточно длинных слов w вида $0^\alpha 1x$

$$U\,[G]\,(w) \leftarrow M\,[F]\,(w) \leftarrow f\,(w). \tag{5}$$

Из (4) следует, что $U\,[G] \in P\,(F)$.

Рассмотрим произвольную частично рекурсивную словарную функцию φ со значениями 0 и 1. Пусть g такова, что для любого y вида $0^\alpha 1z$ имеет место $g\,(y) \simeq \varphi\,(z)$, а для

остальных y значение g неопределено. Для любой машины M, вычисляющей $U\ [G]$ в пределах сложности F по условию (3), существуют такие машина B и частичная функция h с областью определения, равной области определения функции g, что для достаточно длинных y при любом β

$$B\ [G]\ (w) \leftarrow \begin{cases} g\ (y), & \text{если } !h\ (y) \text{ и } h\ (y) \leqslant \beta; \\ U\ [G]\ (w0) & \text{иначе,} \end{cases}$$

где $w \rightleftharpoons y10^{\beta}$. Поскольку в силу (4) $B\ [G] \in P\ (F)$, то согласно (5), для некоторого натурального α при всех достаточно длинных словах w вида $0^{\alpha}1x$ имеет место

$$U\ [G]\ (w) \leftarrow B\ [G]\ (w).$$

Следовательно, для достаточно длинных z при любом β

$$B\ [G]\ (w) \leftarrow \begin{cases} \varphi\ (z), & \text{если } !h\ (y) \text{ и } h\ (y) \leqslant \beta, \\ B\ [G]\ (w0) & \text{иначе,} \end{cases}$$

где $y \rightleftharpoons 0^{\alpha}1z$, $w \rightleftharpoons y10^{\beta}$.

Тогда для всех достаточно длинных z

$$f\ (w) \leftarrow f\ (w0) \leftarrow f\ (w00) \leftarrow \ldots \leftarrow f\ (w0^{H}) \leftarrow \varphi\ (z),$$

где $f \rightleftharpoons B\ [G]$, $w \rightleftharpoons 0^{\alpha}1z1$, $H \rightleftharpoons h\ (0^{\alpha}1z)$, функцию $B\ [G]$ можно продолжить до общерекурсивной функции, поскольку G общерекурсивна и график функции $\lambda x.\ \Phi\ (B, x)$ общерекурсивен (см. [10], [9]). Таким образом, любая частично рекурсивная словарная функция φ со значениями 0 и 1 продолжима до общерекурсивной функции, что противоречит общеизвестному результату (см. [11]). Следовательно, $P\ (F) \neq P\ (G)$. Лемма доказана.

Далее ограничимся машинами типа Θ. Для доказательства теоремы используются следующие леммы.

ЛЕММА 1. *Существует такая машина U, что для любой машины M найдутся такие натуральные числа α и c, что для всех слов w вида $0^{\alpha}1x$ имеет место $U\ (w) \simeq M\ (w)$;*

$$\text{Time}\ (U, w) \leqslant c \cdot \text{Time}\ (M, w);$$
$$\text{Space}\ (U, w) \leqslant c + \text{Space}\ (M, w).$$

Доказательство. Можно считать, что любая машина имеет максимально возможное число лент, головок и ленточные алфавиты всех машин совпадают. Пусть любая машина задается набором команд вида

$$qa_1a_2 \ldots a_n \rightarrow q'\,a_1'a_2' \ldots a_n'\,\Delta_1\Delta_2 \ldots \Delta_n;$$

где q' — состояние, в которое переходит машина, обозревая в состоянии q символы $a_1, \ldots, a_n$ (всеми головками машины на всех лентах); $a_1', \ldots, a_n'$ — символы, записываемые при этом вместо обозреваемых символов; $\Delta_1, \ldots, \Delta_n \in \{b, 0, 1\}$ (b — пробел); если $\Delta_i = b$, то i-я головка после этого остается на месте; если $\Delta_i = 0$ или $\Delta_i = 1$, то она сдвигается на ячейку влево или вправо соответственно. Каждую такую команду закодируем словом

$$Q1a_1 \ldots 1a_nQ'1a_1' \ldots 1a_n'1\Delta_1 \ldots 1\Delta_n,$$

где Q и Q' — коды номеров q и q' соответственно, т. е. двоичная запись этих чисел, в которой цифра 1 заменена на 01, а цифра 0 — на 00. Кодом машины является слово, состоящее из выписанных подряд кодов всех ее команд.

Машина U работает на слове вида $0^\alpha 1x$ следующим образом. Прежде всего, машина переводит α в систему счисления с основанием, равным числу элементов ленточного алфавита, используя при этом в качестве цифр сами символы алфавита, упорядоченные фиксированным способом. Если получачаемое слово T — код некоторой машины, то на первой ленте образуется слово

$$bbIbbTbb0^\alpha 1x,$$

где I — код номера первого состояния, удлиненный нулями до длины T. Число шагов работы на этом этапе и используемая зона зависят только от α.

Далее первая головка первой ленты в процессе моделирования перемещает с собой вдоль ленты слово вида $bbQbbTbb$, где Q — код номера текущего состояния моделируемой машины. То есть слово xy на первой ленте будет моделироваться словом $xbbQbbTbby$, если первая головка обозревает последний символ x. Моделирование одного шага работы состоит в следующем. Машина запоминает обозреваемые символы, находит в T код соответствующей команды, записывает код нового состояния Q' вместо Q, записывает новые символы вместо обозреваемых, производит требуемые сдвиги головок. Вместе с первой головкой на первой ленте сдвигается и служебное слово $bbQbbTbb$. Если какая-либо головка первой ленты попала на служебное слово, то ее следует провести сквозь него. Такую ситуацию можно распознать следующим способом: первая головка первой ленты на каждом шаге заменяет крайние

пробелы служебного слова на символы 0 и обратно; это изменение регистрируется другими головками, попавшими на слежебное слово (для определенности считаем, что запись символа первой головкой производится в последнюю очередь). На моделирование одного шага работы уходит время, ограниченное величиной, зависящей только от α. Число дополнительных ячеек ленты, используемых при моделировании для хранения служебного слова зависит только от α. Лемма доказана.

ЛЕММА 2. *Для любых частично рекурсивной функции g со значениями* 0, 1 *и машины M существует такая частичная функция h (с той же областью определения, что и g) и машина B, что для любого w вида $y10^{\beta}$*

$$B(w) \simeq \begin{cases} g(y), & \text{если } !h(y) \text{ и } h(y) \leqslant \beta, \\ M(w0) & \text{иначе;} \end{cases}$$

$$\text{Time}(B, w) \simeq \begin{cases} 2 \cdot |y| + 6 \cdot \beta + 7, & \text{если } !h(y) \text{ и } h(y) \leqslant \beta, \\ \text{Time}(M, w0) + 2 \cdot |y| + 6 \cdot \beta + 7 & \text{иначе;} \end{cases}$$

$$\text{Space}(B, w) \simeq \begin{cases} |w| + 2, & \text{если } !h(y) \text{ и } h(y) \leqslant \beta, \\ \text{Space}(M, w0) & \text{иначе.} \end{cases}$$

Доказательство. Машина B работает на слове w вида $y10^{\beta}$ следующим образом.

1. Первой головкой она находит конец входного слова за $|y| + \beta + 2$ шага; затем находит за $\beta + 1$ шаг последнюю единицу входного слова и стирает ее.

2. С помощью второй и первой головок машина копирует слово y, переписывая его символы слева направо на нечетные места, занимаемые заключительными β нулями входного слова. На оставшиеся нечетные места машина записывает пробелы и стирает заключительный нуль, если он еще не стерт. На это уходит $\beta + 2$ шага. Если все слово y не удается скопировать на отведенное место, то копируется только начало y.

3. Первая головка за β шагов возвращается на место, занимавшееся последней единицей входного слова. Вторая головка уже находится там, если слово y скопировано полностью.

4. Первая головка движется вправо вдоль последних β ячеек, занимавшихся входным словом. Если входное подслово y было скопировано полностью, то вторая го-

ловка в это время моделирует на копии слова y работу одноголовочной машины, вычисляющей g на одной ленте, продолжаемой только в одну сторону. При этом она успевает промоделировать $[\beta/2]$ шагов. На это уходит β шагов работы B.

5. Головки возвращаются за $|y| + \beta + 2$ шага в исходное положение. Если вычисление g не было завершено или даже не было начато, то в это время на ленте восстанавливается слово $y10^{\beta}0$ и затем выполняется работа машины M. Если вычисление g было завершено, то первая лента очищается и на ней записывается значение $g(y)$.

Таким образом, машина B, работая на любом слове w вида $y10^{\beta}$, делает $2 \cdot |y| + 6 \cdot \beta + 7$ шагов, используя $|w| + 2$ ячейки, если она успевает вычислить $g(y)$, и делает $\mathrm{Time}(M, w0) + 2 \cdot |y| + 6 \cdot \beta + 7$ шагов, используя $\mathrm{Space}(M, w0)$ ячеек, если она не успевает вычислить $g(y)$. В роли h выступает удвоенное время вычисления функции g. Лемма доказана.

Доказательство теоремы. Проверим условия основной леммы, как для границ времени T_1 и T_2, так и для границ объема памяти S_1 и S_2. Условие (1) основной леммы следует из того, что $T_2/T_1 \to +\infty$ и $S_2 - S_1 \to +\infty$. Из этих же соотношений по лемме 1 следует условие (2) основной леммы. Из того, что

$$\forall_n^{\infty} ((T_1(n+1) + 6 \cdot n) \leqslant T_2(n))$$

и

$$\forall^{\infty}{}_n (S_1(n+1) \leqslant S_2(n)),$$

по лемме 2 следует условие (3) основной леммы. Теорема доказана.

Автор выражает признательность Н. К. Косовскому за внимание к работе. Автор также благодарит участников семинара по конструктивной математике и математической логике ЛОМИ АН СССР за обсуждение работы.

Ленинградский государственный университет

Поступило 29.IV.1977

ЦИТИРОВАННАЯ ЛИТЕРАТУРА

1] Hennie F. C., Stearns R. E., Two-tape simulation of multitape Turing machines, J. of the Association for Computing Machinery, **13**, № 4 (1966), 533—546. (Русский перевод: Проблемы математической логики, М., «Мир», 1970, 194—212.)

[2] Stearns R. E., Hartmanis J., Lewis P. M. II, Hierarchies of memory limited computations, IEEE Conf, Rec. Switch. Circuit Theory and Logic. Design, N. Y., 1965, 179—190. (Русский перевод: Проблемы математической логики, М., «Мир», 1970, 301—319.)
[3] Ibarra O. H., A hierarhy theorem for polynomial-space recognition, J. of Cumputer and System Science, 11, № 1 (1975), 56—67.
[4] Paul W. J., On time hierarchies, Proceedings of 9-th Annual ACM Symposium on Theory of Computing, Boulder, 1977, 218—222.
[5] Seiferas J. I., Fisher M. J., Meyer A. R., Refinements of the nondeterministic time and space hierarchies, Proceedings of 14-th Annual Symposium on Switching and Automata Theory, Iowa, 1973, 130—137.
[6] Seiferas J. I., Nondeterministic time and space complexity classes, Project MAC Technical Report 137, Massachusetts Institute of Technology, 1974.
[7] Seiferas J. I., Techniques for separating space complexity classes, J. of Computer and System Science, 14, № 1 (1977), 73—99.
[8] Seiferas J. I., Relating refined space complexity classes, J. of Computer and System Science, 14, № 1 (1977), 100—129.
[9] Blum M., A machine-independent theory of the complexity of recursive functions, J. of the Association for Computing Machinery, 14, № 2 (1967), 322—336.
[10] Трахтенброт Б. А., Сложность алгоритмов и вычислений, Новосибирск, 1967.
[11] Мальцев А. И., Алгоритмы и рекурсивные функции, М., «Наука», 1965.
[12] Constable R. L., The operator gap, J. of the Association for Computing Machinery, 19, № 1 (1972), 175—183.
[13] Cook S. A., A hierarchy for nondeterministic time complexity, J. of Computer and System Science, 7, № 4, (1973), 343—353.