Сибирский математический журнал
Ноябрь—декабрь, 2004. Том 45, № 6

УДК 510.52

СЛОЖНОСТЬ ВЫЧИСЛЕНИЙ В АЛГЕБРАИЧЕСКИХ СИСТЕМАХ

А. Н. Рыбалов

Аннотация: В работе [1] впервые были рассмотрены вычислимость и сложность вычислений над полями вещественных и комплексных чисел. Этот подход был обобщен для произвольной алгебраической системы в [2]. В данной статье предлагается подход к теории сложности вычислений над произвольной алгебраической системой, в котором в качестве вычислительной модели взята вычислимость над списочной надстройкой, предложенная в [3]. Изучаются получающиеся классы полиномиальной сложности. Приводятся некоторые NP-полные проблемы.

Ключевые слова: обобщенная вычислимость, сложность вычисления.

1. Основные понятия

Пусть дана алгебраическая система $\mathfrak{A} = \langle A, \sigma \rangle$ с конечной сигнатурой

$$\sigma = \{c_1, \dots, c_k; p_1, \dots, p_m; f_1, \dots, f_l\}.$$

Здесь c_i — константы, p_i — предикаты, f_i — функции сигнатуры. Следуя [3], введем списочную надстройку $\mathrm{HL}(A)$ множества A:

$$L_0 = A, \quad L_{n+1} = L(L_n) \cup L_n, \quad \mathrm{HL}(A) = \bigcup_{n=0}^{\infty} L_n(A),$$

где $L(M)$ — множество всех списков с элементами из M. Расширим сигнатуру σ до сигнатуры

$$\sigma^* = \sigma \cup \{=, \mathrm{cons}^{(2)}, \mathrm{tail}^{(1)}, \mathrm{head}^{(1)}, \mathrm{nil}\},$$

где функции cons — добавление одного списка в конец другого, tail — отбрасывание первого элемента списка, head — взятие первого элемента списка, а константа nil — пустой список. В итоге получаем систему

$$\mathrm{HL}(\mathfrak{A}) = \langle \mathrm{HL}(A), \sigma^* \rangle,$$

которая называется *списочной надстройкой системы* $\mathfrak{A}$. За основную вычислительную модель примем машины с неограниченными регистрами (МНР) над $\mathrm{HL}(\mathfrak{A})$. Детерминированные МНР определены в [3], недетерминированные — в [4]. Напомним, что недетерминированные МНР бывают двух видов: N_1 и N_2. В первых допускаются недетерминированные ветвления, а во вторых еще и подсказки из $\mathrm{HL}(A)$.

Определим функцию размера $\mathrm{size}_{HL} : \mathrm{HL}(A) \to \mathbb{N}$ следующим образом:

$$\mathrm{size}_{HL}(\alpha) = \begin{cases} 1, & \text{если } \alpha = \mathrm{nil} \text{ или } \alpha \in A, \\ \mathrm{size}_{HL}(\alpha_1) + \dots + \mathrm{size}_{HL}(\alpha_n) + 1 & \text{при } \alpha = \langle \alpha_1, \dots, \alpha_n \rangle. \end{cases}$$

По МНР M над $\mathrm{HL}(\mathfrak{A})$ определим функцию $t_M : \mathrm{HL}(A) \to \mathbb{N} \cup \infty$ — *время работы* M. Если на входе $x \in \mathrm{HL}(A)$ машина не останавливается, то полагаем $t_M(x) = \infty$. Иначе пусть $\tau(x) = \{I_1, \dots, I_n\}$ — вычислительный путь M на x. Введем функцию $t : \mathbb{N} \to \mathbb{N}$ по индукции:

$t(0) = 0$,

$t(i+1) = t(i) + 1$, если I_{i+1} — одна из следующих команд:

$$R_m := c, \quad c \in \sigma^* \text{ — константа},$$

$$R_m := f(R_{i_1}, \dots, R_{i_m}), \text{ где } f \in \sigma \text{ — функция},$$

$$\texttt{if } P(R_{i_1}, \dots, R_{i_m}) \texttt{ then goto } l, \text{ где } P \in \sigma \text{ – предикат},$$

$$\texttt{if ? then goto } l,$$

$t(i+1) = t(i) + \mathrm{size}_{HL}(\alpha_k)$, если I_{i+1} – одна из команд

$$R_m := R_k,$$

$$R_m := \mathrm{tail}(R_k),$$

$$R_m := \mathrm{head}(R_k),$$

$$R_k := \texttt{guess},$$

где α_k — содержимое регистра R_k,

$t(i+1) = t(i) + \mathrm{size}_{HL}(\alpha_k) + \mathrm{size}_{HL}(\alpha_l) + 1$, если I_{i+1} – одна из команд

$$\begin{cases} R_m := \mathrm{cons}(R_k, R_l), \\ \texttt{if } R_k = R_l \texttt{ then goto } t, \end{cases}$$

α_k — содержимое R_k, α_l — содержимое R_l.

Положим теперь $t_M(x) = t(n)$, если МНР M детерминированная, и

$$t_M(x) = \min\{t(n) \text{ по всем вычислительным путям на входе } x\},$$

если МНР M недетерминированная.

Будем говорить, что МНР M *полиномиальна*, если существует полином с целыми коэффициентами $p(n)$ такой, что

$$\forall x \in \mathrm{HL}(A) \ M \text{ останавливается на } x \Rightarrow t_M(x) < p(\mathrm{size}_{HL}(x)).$$

Аналогично определяются время работы и полиномиальная МНР для нескольких аргументов, только в оценке времени берется полином от максимума размеров всех аргументов.

В класс P входят все функции $f : \mathrm{HL}(A)^n \to \mathrm{HL}(A)$, которые вычисляются детерминированными полиномиальными МНР. Класс N_1P (соответственно N_2P) образуют все отношения $\varphi \subseteq \mathrm{HL}(A)^n$, вычислимые полиномиальными N_1-МНР (соответственно N_2-МНР).

Множество $\Omega \subseteq \mathrm{HL}(A)^n$ лежит в классе P, если его характеристическая функция

$$\chi_\Omega(x) = \begin{cases} \langle \mathrm{nil} \rangle, & \text{если } x \in \Omega, \\ \mathrm{nil} & \text{иначе} \end{cases}$$

принадлежит P. Множество $\Omega \subseteq \mathrm{HL}(A)^n$ лежит в классе N_1P (соответственно в N_2P), если там находится его частичная характеристическая функция

$$\hat{\chi}_\Omega(x) = \begin{cases} \mathrm{nil}, \text{ если } x \in \Omega, \\ \text{не определена иначе.} \end{cases}$$

Далее под записью $f = \mathrm{O}(g)$ для функций $f, g : \mathbb{N} \to \mathbb{N}$ будет пониматься следующее:

$$\exists C \in \mathbb{N} \ \forall n \in \mathbb{N} \quad f(n) \le Cg(n).$$

2. Вспомогательные утверждения

Установим несколько вспомогательных фактов, которые позволяют ограничивать вид недетерминированных машин, не сужая при этом классов N_1P и N_2P.

Лемма 2.1. *Пусть $\varphi \subseteq \mathrm{HL}(A)^n$ — N_2P-отношение, тогда существует полиномиальная N_2-МНР, вычисляющая это отношение, в которой отсутствуют недетерминированные ветвления.*

Доказательство. Действительно, команду

$$\texttt{N if ? then goto } m$$

можно заменить эквивалентным ей блоком

$$\begin{array}{ll} \texttt{N} & R_{k+1} := \texttt{guess} \\ \texttt{N+1} & \texttt{if } R_{k+1} = \text{ nil } \texttt{then goto } m, \end{array}$$

где R_{k+1} — дополнительный регистр (в старой машине было k регистров). Еще необходимо увеличить на 1 номера последующих команд и метки переходов в соответствующих инструкциях `goto`. Время работы при этом увеличивается на 2, так как для моделирования достаточно подсказок nil и $\langle \text{nil} \rangle$. Заменив таким образом все недетерминированные ветвления, получим МНР, содержащую только команды подсказки, причем если до этого МНР была полиномиальной, то и после переделки она останется полиномиальной. □

Лемма 2.2. *Пусть $\varphi \subseteq \mathrm{HL}(A)^n$ — N_2P-отношение, тогда существует полиномиальная N_2-МНР, вычисляющая φ, в которой все подсказки берутся из множества A.*

Доказательство. Для краткости будем обозначать такие подсказки через $\texttt{guess}_A$. Необходимо смоделировать команду

$$R_i := \texttt{guess}$$

командами

$$\texttt{if ? then goto } m$$

$$R_i := \texttt{guess}_A.$$

Пусть нужно получить подсказку $\alpha \in \mathrm{HL}(A)$. Это делается в несколько этапов. Сначала генерируется носитель α — список глубины 1, составленный из всех праэлементов, входящих в α, в том порядке, в каком они встречаются при просмотре списка слева направо. Для этого используется следующая подпрограмма:

$$\begin{array}{ll} \texttt{N} & R_i := \text{nil} \\ \texttt{N+1} & \texttt{if ? then goto N+5} \\ \texttt{N+2} & R_{k+1} := \texttt{guess}_A \\ \texttt{N+3} & R_i := \text{cons}(R_i, R_{k+1}) \\ \texttt{N+4} & \texttt{gotoN+1}. \end{array}$$

Пробегая носитель слева направо, образуем подсписки высшего уровня, пробегая то, что получилось, образуем подсписки более низкого уровня, и так далее до самого первого уровня. Подпрограмма, которая образует подсписки l-го уровня имеет вид

```
N      R_{k+1} := nil
N+1    R_{k+2} := nil
N+2    if ? then goto N+8
N+3    if ? then goto N+9
N+4    R_{k+3} := head(R_i)
N+5    R_i := tail(R_i)
N+6    R_{k+2} := cons(R_{k+2}, R_{k+3})
N+7    goto N+3
N+8    R_{k+1} := cons(R_{k+1}, R_{k+2})
N+9    if ? then goto N+13
N+10   R_{k+3} := head(R_i)
N+11   R_i := tail(R_i)
N+12   R_{k+1} := cons(R_{k+1}, R_{k+3})
N+13   if ? then goto N+1
N+14   R_i := R_{k+1}.
```

Повторяя эту процедуру нужное количество раз в недетерминированном цикле, можно получить любую подсказку α из $\mathrm{HL}(A)$.

Время моделирования подсказки линейно зависит от глубины α и числа праэлементов в носителе α, но так как глубина и размер носителя меньше $\mathrm{size}_{HL}(\alpha)$, то время будет $\mathrm{O}(\mathrm{size}_{HL}(\alpha)^2)$.

Заменяя подобным образом все инструкции подсказки и соответствующим образом корректируя нумерацию команд и меток переходов, получим нужную полиномиальную МНР. $\square$

Следствие 2.1. *Пусть $\varphi \subseteq \mathrm{HL}(A)^n$ — N_2P-отношение, вычислимое N_2-*МНР*, в которой все подсказки берутся из* $\mathrm{HL}(\varnothing)$. *Тогда существует полиномиальная N_1-*МНР, *которая вычисляет φ.*

Доказательство. Носители элементов из $\mathrm{HL}(\varnothing)$ можно считать списками глубины 1, составленными из единственного элемента nil, поэтому, воспользовавшись леммой 2.2, а затем заменив команды $R_i := \mathtt{guess}_A$ командами $R_i := \mathrm{nil}$, получим нужную полиномиальную N_1-МНР. $\square$

Теперь получим лемму, утверждающую, что в вычислении N_2P-отношений можно обойтись одной подсказкой в самом начале работы.

Лемма 2.3. *Пусть отношение $\varphi \subseteq \mathrm{HL}(A) \times \mathrm{HL}(A)$ лежит в классе N_2P. Тогда существуют полином $p(n)$ и полиномиальная функция $g : \mathrm{HL}(A) \times \mathrm{HL}(A) \to \mathrm{HL}(A)$ такие, что*

$$\varphi(x, y) \Leftrightarrow \exists z \in \mathrm{HL}(A) \ : \ \mathrm{size}_{HL}(z) < p(\mathrm{size}_{HL}(x)) \text{ и } g(z, x) = y.$$

Доказательство. По лемме 2.1 можно без ограничения общности полагать, что отношение φ вычисляется N_2-МНР M без недетерминированных ветвлений. Пусть при этом время работы ограничено полиномом $q(n)$. Строим детерминированную МНР M^* по машине M так: сначала добавим новый регистр, затем сдвинем все регистры на 1, освободив тем самым регистр R_1, заменим в полученной машине все команды

$$R_m := \texttt{guess}$$

блоками команд

$$R_m := \mathrm{head}(R_1)$$
$$R_1 := \mathrm{tail}(R_1).$$

Получилась детерминированная МНР M^* от двух аргументов. Утверждается, что M^* вычисляет нужную нам функцию g. Действительно, пусть $\varphi(x, y)$ и M работает время, ограниченное $q(\mathrm{size}_{HL}(x))$. В процессе работы используются подсказки $\alpha_1, \ldots, \alpha_n \in \mathrm{HL}(A)$, причем

$$\mathrm{size}_{HL}(\alpha_1) + \cdots + \mathrm{size}_{HL}(\alpha_n) < q(\mathrm{size}_{HL}(x)).$$

Поэтому полагаем $p(n) = q(n)$. Возьмем теперь $z = \langle \alpha_1, \ldots, \alpha_n \rangle$, тогда, работая на паре (z, x), машина M^* выдает y. При этом время работы может увеличиться лишь на $\mathrm{O}(p^2(\mathrm{size}_{HL}(x)))$ за счет сложности выполнения операций head и tail. Наоборот, если

$$\exists z \in \mathrm{HL}(A) \quad \mathrm{size}_{HL}(z) < p(\mathrm{size}_{HL}(x)) \text{ и } g(z, x) = y,$$

то существует вычислительный путь в машине M, на котором используются подсказки, соответствующие элементу z. Работая по этому пути, M по входу x дает y, и время ее работы ограничено $q(\mathrm{size}_{HL}(x)) + r(\mathrm{size}_{HL}(x))$, где r — полином, ограничивающий сложность g. □

Аналогичное утверждение имеет место для N_1P-отношений.

Лемма 2.4. *Пусть отношение* $\varphi \subseteq \mathrm{HL}(A) \times \mathrm{HL}(A)$ *лежит в классе* N_1P. *Тогда существуют полином* $p(n)$ *и полиномиальная функция* $g : \mathrm{HL}(A) \times \mathrm{HL}(A) \to \mathrm{HL}(A)$ *такие, что*

$$\varphi(x, y) \Leftrightarrow \exists z \in \mathrm{HL}(\varnothing)\ \mathrm{size}_{HL}(z) < p(\mathrm{size}_{HL}(x)) \text{ и } g(z, x) = y.$$

Доказательство. Действительно, согласно доказательству леммы 2.1 в N_1-МНР, вычисляющей φ, все недетерминированные ветвления можно заменить подсказками $\texttt{guess} \in \mathrm{HL}(\varnothing)$. Затем, используя идею доказательства леммы 2.3, объединяем все подсказки в одну, которая берется в самом начале работы МНР. После всех указанных манипуляций получаем полиномиальную детерминированную МНР, которая вычисляет нужную нам двуместную функцию f. □

В заключение раздела докажем утверждение, которое пригодится в дальнейшем.

Лемма 2.5. *Пусть время работы* МНР M *на входе* a *есть* T. *Тогда размер содержимого любого регистра* M *в любой момент работы не превосходит величины* $\max\{\mathrm{size}_{HL}(a), T\}$.

Доказательство. Пусть $\tau(a) = I_0, \ldots, I_n$ — вычислительный путь M на входе a. Докажем по индукции, что максимум размеров регистров на i-м шаге вычисления не превосходит $\max\{\mathrm{size}_{HL}(a), t(i)\}$, где t — функция времени выполнения i первых шагов из определения функции временной сложности. При

$i = 0$ в первом регистре находится a, а в остальных nil, а $t(0) = 0$. Поэтому необходимая оценка выполняется. Пусть теперь для числа шагов k утверждение выполнено; докажем его для $k+1$. Очевидно, что максимум размеров регистров может увеличиться, только если $k+1$-я команда есть либо $R_l :=$ `guess`, либо $R_l := \mathrm{cons}(R_s, R_t)$, но тогда в первом случае

$$t(i+1) = t(i) + \mathrm{size}_{HL}(R_l) + 1 \geq \max\{\mathrm{size}_{HL}(R_l), t(i)\},$$

во втором

$$\begin{aligned} t(i+1) = t(i) + \mathrm{size}_{HL}(R_s) + \mathrm{size}_{HL}(R_r) + 1 \\ = t(i) + \mathrm{size}_{HL}(R_l) \geq \max\{\mathrm{size}_{HL}(R_l), t(i)\}. \end{aligned}$$

В остальных случаях возрастает лишь $t(i+1)$. Таким образом, для любого $i = 0, \dots, n$ размеры содержимого всех регистров ограничены $t(i)$. В частности, это справедливо для n, а по определению $T = t(n)$. $\square$

3. Натуральные числа

Определим натуральные числа $\mathbb{N}_{HL}$ в списочной надстройке:

$$0_{HL} = \mathrm{nil}, \quad n+1_{HL} = \mathrm{cons}(n_{HL}, \mathrm{nil}) = \langle \underbrace{\mathrm{nil}, \dots, \mathrm{nil}}_{n \text{ раз}} \rangle.$$

Легко заметить, что $\mathrm{size}_{HL}(n_{HL}) = n + 1 \in \mathbb{N}$.

В дальнейшем нам потребуется использовать их при моделировании различных операций, поэтому оценим заранее сложность увеличения натурального числа на 1 и соответственно уменьшения его на 1.

Лемма 3.1. *Имеют место следующие утверждения.*

1. *Функция* $\mathrm{Inc} : \mathbb{N}_{HL} \to \mathbb{N}_{HL}$, *определенная как* $\mathrm{Inc}(n_{HL}) = n+1_{HL}$, *вычислима за линейное время.*

2. *Функция* $\mathrm{Dec} : \mathbb{N}_{HL} \to \mathbb{N}_{HL}$, *определенная как*

$$\mathrm{Dec}(n_{HL}) = \begin{cases} n - 1_{HL}, & \text{если } n_{HL} \neq \mathrm{nil}, \\ \mathrm{nil}, & \text{если } n_{HL} = \mathrm{nil}, \end{cases}$$

вычислима за линейное время.

Доказательство. 1. В самом деле, Inc вычисляет следующая МНР:

$$\texttt{1}\ R_2 := \mathrm{cons}(R_1, \mathrm{nil}),$$

где в регистре R_1 хранится аргумент, а в R_2 — значение функции. Очевидно, что время вычисления на входе n_{HL} есть $\mathrm{size}_{HL}(n_{HL}) + 2$.

2. Функцию Dec вычисляет МНР:

$$\texttt{1}\ R_2 := \mathrm{tail}(R_1),$$

с учетом того, что $\mathrm{tail(nil)} = \mathrm{nil}$. Время вычисления $\mathrm{size}_{HL}(n_{HL})$. $\square$

Далее будем опускать индекс у элементов $\mathbb{N}_{HL}$ и вместо n_{HL} писать просто n, так как из контекста будет понятно, где обычные натуральные числа, а где списки. Также вместо $\mathrm{Inc}(n)$ и $\mathrm{Dec}(n)$ будем писать просто $n+1$ и $n-1$.

4. Полные проблемы

Пусть $S_1, S_2 \subseteq \mathrm{HL}(A)$. Множество S_1 *полиномиально сводится* к множеству S_2 (будем обозначать это через $S_1 \leq_p S_2$), если существует такая полиномиальная вычислимая функция $r : \mathrm{HL}(A) \to \mathrm{HL}(A)$, что

$$\forall x \in \mathrm{HL}(A) \quad x \in S_1 \Leftrightarrow r(x) \in S_2.$$

Множества S_1 и S_2 *полиномиально эквивалентны* ($S_1 \equiv_p S_2$), если $S_1 \leq_p S_2$ и $S_2 \leq_p S_1$. Множество $S \in \mathrm{HL}(A)$ называется *N_1P-полным* (соответственно *N_2P-полным*), если $S \in N_1P$ (соответственно $S \in N_2P$) и любое множество $B \in N_1P$ (N_2P) полиномиально сводится к S.

Введем кодировку термов и бескванторных формул сигнатуры σ^* с параметрами из A. Сначала определим кодировку термов α следующим образом:

$$\begin{aligned}
&\alpha(x_i) = \langle 0, i\rangle,\\
&\alpha(c) = \langle 1, c\rangle,\ c - \text{элемент из } A \text{ или nil},\\
&\alpha(\mathrm{head}(t)) = \langle 2, \alpha(t)\rangle,\ \text{где } t - \text{ терм},\\
&\alpha(\mathrm{tail}(t)) = \langle 3, \alpha(t)\rangle,\ \text{где } t - \text{ терм},\\
&\alpha(\mathrm{cons}(t_1, t_2)) = \langle 4, \alpha(t_1), \alpha(t_2)\rangle,\ \text{где } t_1, t_2 - \text{ термы},\\
&\alpha(F_i(t_1, \ldots, t_n)) = \langle 5, i, \alpha(t_1), \ldots, \alpha(t_n)\rangle.
\end{aligned}$$

Далее кодирование бескванторных формул β осуществляется так:

$$\begin{aligned}
&\beta(t_1 = t_2) = \langle 0, \alpha(t_1), \alpha(t_2)\rangle,\\
&\beta(R_i(t_1, \ldots, t_n)) = \langle 1, i, \alpha(t_1), \ldots, \alpha(t_n)\rangle,\ R_i - \text{ предикат},\\
&\beta(\neg\Phi) = \langle 2, \beta(\Phi)\rangle,\\
&\beta(\Phi_1 \wedge \Phi_2) = \langle 3, \beta(\Phi_1), \beta(\Phi_2)\rangle,\\
&\beta(\Phi_1 \vee \Phi_2) = \langle 4, \beta(\Phi_1), \beta(\Phi_2)\rangle,\\
&\beta(\Phi_1 \to \Phi_2) = \langle 5, \beta(\Phi_1), \beta(\Phi_2)\rangle.
\end{aligned}$$

Лемма 4.1. *Функция*

$$vt(x, y) = \begin{cases} t(a_1, \ldots, a_n), & \text{если } y = \alpha(t) \text{ и } x = \langle a_1, \ldots, a_n\rangle,\\ \text{не определено} & \text{иначе}\end{cases}$$

лежит в классе P.

Доказательство. МНР, вычисляющая функцию vt, действует следующим образом. В цикле производится упрощение кода терма. На каждой итерации цикла делается разбор текущего кода y. Если y есть код переменной, константы сигнатуры σ^* или параметра-праэлемента, то цикл заканчивается и выдаются значение переменной, константы или праэлемента (который берется из кода). Если y является кодом функции от термов, то он имеет вид

$$\langle 2, i, y_1, \ldots, y_n\rangle,$$

где y_j — коды термов более низкого уровня. Теперь выбираем тот среди них, который не является кодом переменной или константы. При отсутствии таких кодов вычисляется значение i-й функции на значениях y_i и совершается переход к следующей итерации. Если y_s — первый код, не являющийся кодом константы или переменной, то вместо исходного кода y пишется

$$\langle\langle 2, i, y_1, \ldots, y_{s-1}, \mathrm{nil}, y_{s+1}, \ldots, y_n\rangle, y_s\rangle.$$

Затем то же самое делается с кодом y_s: удаляется код, не являющийся кодом константы или переменной, и приписывается справа. Так продолжается до тех пор, пока не дойдем до кода подтерма, все подтермы которого — переменные, константы или праэлементы. Тогда этот код заменяется значением терма. Далее происходит обратный процесс. Код собирается справа налево из цепочки кодов (самый правый код подставляется на то место во втором справа, где находится nil, затем второй — в третий, и т. д.). Параллельно производится проверка того, является ли y кодом терма, если нет, то происходит зацикливание.

Заметим, что после выполнения одной такой итерации длина кода терма уменьшается. Это происходит потому, что один из самых нижних подтермов, имеющих вид $F_i(t_1, \dots, t_m)$, где t_j либо константы, переменные, праэлементы, либо уже вычисленные списки, заменяется его значением. Таким образом, после не более чем $\mathrm{size}_{HL}(y)$ упрощений будет вычислено значение терма.

Оценим время выполнения одного упрощения. На каждой итерации упрощения идет работа с одним подтермом, занимающая $\mathrm{O}(\mathrm{size}_{HL}(y) \times \mathrm{size}_{HL}(x))$ единиц времени. Таких подтермов не более $\mathrm{size}_{HL}(y)$, а потому само упрощение происходит за время $\mathrm{O}(\mathrm{size}_{HL}(y)^2 \times \mathrm{size}_{HL}(x))$. Вычисляется значение терма после не более чем $\mathrm{size}_{HL}(y)$ упрощений, поэтому вся работа делается за время

$$\mathrm{O}(\mathrm{size}_{HL}(y)^3 \times \mathrm{size}_{HL}(x)) = \mathrm{O}(\max\{\mathrm{size}_{HL}(y), \mathrm{size}_{HL}(x)\}^4).$$

А это означает, что полученная МНР работает полиномиально. $\square$

Лемма 4.2. *Функция*

$$vf(x, y) = \begin{cases} \Phi(a_1, \dots, a_n), & \text{если } y = \beta(\Phi) \text{ и } x = \langle a_1, \dots, a_n \rangle, \\ \text{не определено} & \text{иначе} \end{cases}$$

лежит в классе P.

Доказательство. Построение полиномиальной МНР, вычисляющей данную функцию, аналогично построению МНР для термов в предыдущей лемме. При этом используются полиномиальная функция vt и то, что логические связки можно смоделировать на МНР. $\square$

Теперь определим следующую проблему:

$$\begin{aligned} SAT_2 = \{\beta(\Phi),\ &\text{где } \Phi \text{ — формула с параметрами такая, что} \\ &\exists a_1, \dots, a_n \in \mathrm{HL}(A)\ \ \mathrm{HL}(\mathfrak{A}) \models \Phi(a_1, \dots, a_n) \\ &\text{и для всех } i \text{ имеет место } \mathrm{size}_{HL}(a_i) \le \mathrm{size}_{HL}(\beta(\Phi))\}. \end{aligned}$$

Лемма 4.3. *$SAT_2 \in N_2P$, $SAT_2 \in N_2P$.*

Доказательство. N_2-МНР для проблемы SAT_2 есть МНР для функции vf, вначале которой добавлена команда подсказки из $\mathrm{HL}(A)$, которая затем подается как первый аргумент функции vf. То, что это полиномиальная МНР, следует из того, что достаточно подсказки размером не больше кода формулы. $\square$

Заметим, что условие ограниченности размеров элементов, на которых истинна формула, в формулировке проблемы SAT_2 существенно. Действительно, если не будет такого ограничения, то элементы, делающие истинной формулу, могут быть экспоненциально большого размера по сравнению с размером кода формулы, а потому указанная проблема не будет лежать в классе N_2P. Для примера рассмотрим следующий набор формул:

$$\Phi_n = (x_n = \mathrm{cons}(x_{n-1}, x_{n-1}) \wedge x_{n-1} = \mathrm{cons}(x_{n-2}, x_{n-2}) \wedge \dots \wedge x_2 = \mathrm{cons}(x_1, x_1)).$$

Легко видеть, что

$$\mathrm{size}_{HL}(\beta(\Phi_n)) = 20 + 3n + \mathrm{size}_{HL}(\beta(\Phi_{n-1})),$$

поэтому

$$\mathrm{size}_{HL}(\beta(\Phi_n)) = 20n + \frac{3}{2}n(n+1).$$

Но значения x_i, которые делают Φ_n истинной, в лучшем случае имеют размеры $\mathrm{size}_{HL}(x_i) = 2^i$, т. е. размеры значений переменных растут экспоненциально, в то время как размеры кодов формул имеют полиномиальный рост.

Для дальнейшего потребуется

Лемма 4.4. *Верны следующие утверждения.*

1. *Существует константа C такая, что для любого $a \in \mathrm{HL}(\varnothing)$ существует терм t от* nil *сигнатуры σ^* такой, что $t = a$ и*

$$\mathrm{size}_{HL}(\alpha(t)) \leq C\,\mathrm{size}_{HL}(a)^2.$$

2. *Существует константа C такая, что для любого $a \in \mathrm{HL}(A)$ существует терм t сигнатуры σ^* от праэлементов и* nil *такой, что $t = a$ и*

$$\mathrm{size}_{HL}(\alpha(t)) \leq C\,\mathrm{size}_{HL}(a)^2.$$

Доказательство. Проведем доказательство для случая 1, случай 2 рассматривается аналогично. Положим $C = 5$. Используем индукцию по глубине списка. Для глубины 1 (элемент nil) утверждение очевидно: список nil представляется термом-константой nil, причем $\mathrm{size}_{HL}(\mathrm{nil}) = 1$, а $\mathrm{size}_{HL}(\alpha(\mathrm{nil})) = 4$. Нужное неравенство имеет вид

$$4 \leq 5 \times 1^2.$$

Теперь допустим, что утверждение верно для всех списков глубины $\leq n$, и докажем его для списков глубины $n + 1$. Пусть $a = \langle a_1, \dots, a_k\rangle$ и найдены термы t_i, $i = 1, \dots, k$, такие, что $t_i = a_i$. Положим тогда

$$t = \mathrm{cons}(\dots \mathrm{cons}(\mathrm{cons}(\mathrm{nil}, t_1), t_2) \dots, t_k).$$

Легко видеть, что $t = a$. Теперь

$$\mathrm{size}_{HL}(\alpha(t)) = 4 + 5k + \sum_{i=1}^{k} \mathrm{size}_{HL}(\alpha(t_i))$$

и

$$\mathrm{size}_{HL}(a) = 1 + \sum_{i=1}^{k} \mathrm{size}_{HL}(a_i).$$

Так как по предположению индукции для t_i верны неравенства

$$\mathrm{size}_{HL}(\alpha(t_i)) \leq 5\,\mathrm{size}_{HL}(a_i)^2,$$

можно написать

$$\mathrm{size}_{HL}(\alpha(t)) = 5k + 4 + \sum_{i=1}^{k} \mathrm{size}_{HL}(\alpha(t_i)) \leq 5k + 4 + 5\sum_{i=1}^{k} \mathrm{size}_{HL}(a_i)^2$$
$$\leq 5\left(1 + \sum_{i=1}^{k} \mathrm{size}_{HL}(a_i)\right)^2 = 5\,\mathrm{size}_{HL}(a)^2. \qquad \square$$

Теорема 4.1. *SAT_2 — N_2P-полная проблема.*

Доказательство. По лемме 4.3 SAT_2 лежит в классе N_2P. Докажем, что любое множество $S \in N_2P$ полиномиально сводится к SAT_2. Пусть M — полиномиальная N_2P-МНР, распознающая S. Полином $p(n)$ — оценка времени ее работы. Нужно по входу $s \in \mathrm{HL}(A)$ построить такую бескванторную формулу Φ сигнатуры σ^* с параметрами-праэлементами, что вычислительный путь M на входе s, который заканчивается остановкой, существует тогда и только тогда, когда существуют $a_i \in \mathrm{HL}(A)$, $i = 1, \dots, l$, такие, что $\Phi(a_1, \dots, a_l)$ истинна. При этом $\beta(\Phi) = r(s)$, где r — нужная нам сводимость.

Пусть МНР M имеет регистры $R_1, \dots, R_n$ (в регистре R_1 — вход). Шагу i работы поставим в соответствие n переменных $x_{i,1}, \dots, x_{i,n}$, которые отвечают за содержимое регистров, переменную y_i, содержащую номер текущей команды, и переменную g_i, отвечающую за подсказки. Для краткости набор переменных $x_{i,1}, \dots, x_{i,n}$ будет обозначаться через $\bar{x}_i$.

Начальной конфигурации будет соответствовать формула

$$\mathrm{Start}(y_1, \bar{x}_1) = (x_{1,1} = t) \wedge (y_1 = 1) \wedge \bigwedge_{l=2}^{n} (x_{1,l} = \mathrm{nil}),$$

где t — терм сигнатуры σ^* от элемента nil и параметров-праэлементов из A, значение которого есть s. Такой терм существует по лемме 4.4.

Пусть $I_1, \dots, I_m$ — набор команд M. Без ограничения общности можно считать, что добавлена новая команда — команда остановки `stop`, которая заменяет все остановочные переходы (т. е. переходы к j-й команде, где $j > m$, заменены переходами к некоторой I_k, уже с $k \le m$ и $I_k =$ `stop`). Тогда переходу от i-й конфигурации к $i+1$-й будет соответствовать формула

$$\mathrm{Step}(y_i, y_{i+1}, g_i, g_{i+1}, \bar{x}_i, \bar{x}_{i+1}) = \bigwedge_{j=1}^{m} ((y_i = j) \to \mathrm{Com}_j(y_i, y_{i+1}, g_i, g_{i+1}, \bar{x}_i, \bar{x}_{i+1})),$$

где формула Com_j зависит от j-й команды МНР следующим образом.

Если I_j есть «$R_k := c$», где c — константа из $\mathrm{HL}(A)$, то

$$\mathrm{Com}_j = (x_{i+1,k} = c) \wedge (y_{i+1} = y_i + 1) \wedge \bigwedge_{l \ne k} (x_{i+1,l} = x_{i,l}).$$

Если I_j — это «$R_k := R_t$», то

$$\mathrm{Com}_j = (x_{i+1,k} = x_{i,t}) \wedge (y_{i+1} = y_i + 1) \wedge \bigwedge_{l \ne k} (x_{i+1,l} = x_{i,l}).$$

Если I_j есть «$R_k := f(R_{j_1}, \dots, R_{j_t})$», где f — функция из σ^*, то

$$\mathrm{Com}_j = (x_{i+1,k} = f(x_{i,j_1}, \dots, x_{i,j_t})) \wedge (y_{i+1} = y_i + 1) \wedge \bigwedge_{l \ne k} (x_{i+1,l} = x_{i,l}).$$

Если I_j — это «`if` $R_k = R_t$ `then` goto q», то

$$\mathrm{Com}_j = ((x_{i,k} = x_{i,t}) \to (y_{i+1} = q)) \wedge \left(\neg(x_{i,k} = x_{i,t}) \to (y_{i+1} = y_i + 1)) \wedge \bigwedge_{l=1}^{n} (x_{i+1,l} = x_{i,l} \right).$$

Если I_j есть «`if` $P(R_{j_1}, \ldots, R_{j_t})$ `then goto` q», где P — предикат из σ^*, то

$$\begin{aligned}\mathrm{Com}_j = (P(x_{i,j_1}, \ldots, x_{i,j_t}) \to (y_{i+1} = q))\\ \wedge \left(\neg P(x_{i,j_1}, \ldots, x_{i,j_t}) \to (y_{i+1} = y_i + 1)) \wedge \bigwedge_{l=1}^{n} (x_{i+1,l} = x_{i,l}\right).\end{aligned}$$

Если I_j есть «`if` ? `then goto` q», то

$$\mathrm{Com}_j = ((y_{i+1} = q) \vee (y_{i+1} = y_i + 1)) \wedge \bigwedge_{l=1}^{n} (x_{i+1,l} = x_{i,l}).$$

Если I_j — это «$R_k :=$ `guess`», то

$$\mathrm{Com}_j = (x_{i+1,k} = g_i) \wedge (y_{i+1} = y_i + 1) \wedge \bigwedge_{l \neq k} (x_{i+1,l} = x_{i,l}).$$

Если I_j — команда остановки «stop», то

$$\mathrm{Com}_j = (y_{i+1} = y_i) \wedge \bigwedge_{l=1}^{n} (x_{i+1,l} = x_{i,l}).$$

Теперь окончательная формула, характеризующая работу МНР на входе s будет иметь вид

$$\begin{aligned}&\mathrm{Act}(\bar{x}_1, \ldots, \bar{x}_T, y_1, \ldots, y_T, g_1, \ldots, g_T)\\ &\quad = \mathrm{Start}(y_1, \bar{x}_1) \wedge \left(\bigwedge_{i=1}^{T-1} \mathrm{Step}(y_i, y_{i+1}, g_i, g_{i+1}, \bar{x}_i, \bar{x}_{i+1})\right) \wedge (x_{T,1} = 1),\end{aligned}$$

где $T = p(\mathrm{size}_{HL}(s))$.

Легко убедиться, что МНР M останавливается на входе s не более чем через $p(\mathrm{size}_{HL}(s))$ шагов и выдает 1 тогда и только тогда, когда существуют такие элементы

$$a_{i,j}, b_i, c_i \in \mathrm{HL}(A), \quad j = 1, \ldots, n,\ i = 1, \ldots, T,$$

что на них истинна формула Act. Именно, $a_{i,j}$ — содержимое j-го регистра на i-м шаге вычисления, b_i — номер команды, выполняемой на i-м шаге, c_i — подсказка на i-м шаге (nil, если на i-м шаге нет подсказки). Размеры этих списков ограничены временем работы МНР T. Действительно, размеры номеров команд не больше m — числа команд МНР, т. е. ограничены константой, не зависящей от размера входа s. Размеры подсказок, которые делаются в процессе работы, меньше T, так как команда подсказки занимает число единиц времени, равное размеру подсказки. Размер содержимого каждого регистра ограничен T по лемме 2.5. В свою очередь, T не больше размера кодировки получившейся формулы. Действительно,

$$\begin{aligned}&\mathrm{size}_{HL}(\beta(\mathrm{Act})) = 4 + \mathrm{size}_{HL}(\beta(\mathrm{Start})) + 4(T-1)\\ &\qquad + \sum_{i=1}^{T-1} \mathrm{size}_{HL}(\beta(\mathrm{Step}_i)) + 19 + T(n-1) > T.\end{aligned}$$

Это и требуется в формулировке проблемы SAT_2. Теперь $r(s) = \beta(\mathrm{Act})$ — нужная сводимость. Покажем, что она полиномиальна. Заметим, что переменных в формуле Act $T(n+2)$ штук. Занумеруем их для единообразия:

$$\begin{aligned} x_{1,1}, x_{1,2}, \dots, x_{T,n} &\to x_1, \dots, x_{Tn}, \\ y_1, \dots, y_T &\to x_{Tn+1}, \dots, x_{T(n+1)}, \\ g_1, \dots, g_T &\to x_{T(n+1)+1}, \dots, x_{T(n+2)}. \end{aligned}$$

Теперь оценим размер кода формулы Act:

$$\mathrm{size}_{HL}(\beta(\mathrm{Act})) = \mathrm{size}_{HL}(\beta(\mathrm{Start})) + 4T + \sum_{i=1}^{T-1} \mathrm{size}_{HL}(\beta(\mathrm{Step}_i))$$
$$+19 + T(n-1) \leq 19 + T(n+3) + \mathrm{size}_{HL}(\beta(\mathrm{Start})) + T \max_{i \in 1,\dots,T} \{\mathrm{size}_{HL}(\beta(\mathrm{Step}_i))\}.$$

Оценим размер кода формулы Start:

$$\begin{aligned} \mathrm{size}_{HL}(\beta(\mathrm{Start})) &= 4 + 6 + \mathrm{size}_{HL}(\alpha(t)) + 2 + Tn + 4 + 13 \\ &+ \sum_{l=2}^{n} \bigl(4 + 2 + l + 7\bigr) = 15 + 13n + Tn + \frac{n(n+1)}{2} + \mathrm{size}_{HL}(\alpha(t)), \end{aligned}$$

где t — терм, представляющий вход s. Согласно лемме 4.4

$$\mathrm{size}_{HL}(\alpha(t)) \leq 5\,\mathrm{size}_{HL}(s)^2.$$

Заметим теперь, что формулы Step_i различаются только нумерацией переменных, а потому максимальный номер достигается на Step_T:

$$\mathrm{size}_{HL}(\beta(\mathrm{Step}_T)) = \sum_{j=1}^{m} (15 + \mathrm{size}_{HL}(\alpha(t_j)) + T(n+1) + \mathrm{size}_{HL}(\beta(\mathrm{Com}_j))),$$

где t_j — терм, представляющий натуральное число j. По лемме 4.4

$$\mathrm{size}_{HL}(\alpha(t_j)) \leq 5\,\mathrm{size}_{HL}(j) = 5(j+1).$$

Поэтому

$$\begin{aligned} \mathrm{size}_{HL}(\beta(\mathrm{Step}_T)) &\leq \sum_{j=1}^{m} (15 + 5(j+1) + T(n+1) + \mathrm{size}_{HL}(\beta(\mathrm{Com}_j))) \\ &= 20m + \frac{5m(m+1)}{2} + Tm(n+1) + \sum_{j=1}^{m} \mathrm{size}_{HL}(\beta(\mathrm{Com}_j)). \end{aligned}$$

По виду формул Com_j легко делаются полиномиальные оценки размеров их кодов.

Проведенные выше вычисления показывают, что размер кода формулы Act оценивается полиномиально от размера входа s. □

Автор благодарит И. В. Ашаева за постановку задачи и полезные замечания в процессе работы над ней, В. Н. Ремесленникова, а также анонимного рецензента за внимательное прочтение текста и предложения по улучшению статьи.

ЛИТЕРАТУРА

1. *Blum L., Shub M., Smale S.* On a theory of computation and complexity over the real numbers: NP-completeness, recursive functions and universal machines // Bull. Amer. Math. Soc. (N.S.) 1989. V. 21, N 1. P. 1–46.
2. *Hemmerling A.* Computability and Complexity over structures // Math. Logic Quart. 1998. V. 44, N 1. P. 1–44.
3. *Ашаев И. В., Беляев В. Я., Мясников А. Г.* Подходы к теории обобщенной вычислимости // Алгебра и логика. 1993. Т. 32, № 4. С. 349–386.
4. *Романов Р. В.* Некоторые проблемы обобщенной вычислимости. Омск: Омск. гос. университет, 1998. 32 с. (Препринт №98–03).

Статья поступила 20 апреля 2004 г.

Рыбалов Александр Николаевич
Омский гос. университет,
кафедра математической логики и логического программирования,
пр. Мира, 55-А, Омск 644077
rybalov@omskreg.ru