Л. Д. Беклемишев, А. А. Оноприенко, О некоторых медленно сходящихся системах преобразований термов, *Матем. сб.*, 2015, том 206, номер 9, 3–20

DOI: http://dx.doi.org/10.4213/sm8519

УДК 510.23+510.58

Л. Д. Беклемишев, А. А. Оноприенко

О некоторых медленно сходящихся системах преобразований термов

Формулируются системы преобразований термов, число шагов работы которых на произвольном входе конечно, но не ограничивается никакой вычислимой функцией, доказуемо тотальной в арифметике Пеано PA. Тем самым, утверждение о сходимости таких систем не доказуемо в PA. Эти системы получаются из независимого комбинаторного утверждения, известного как принцип червя; их также можно рассматривать как вариант хорошо известной игры Геракла и гидры, введенной Дж. Парисом и Л. Кирби.

Библиография: 16 названий.

Ключевые слова: системы преобразований термов, арифметика Пеано, принцип червя.

DOI: 10.4213/sm8519

§ 1. Введение

Первые примеры систем преобразований термов (term rewriting systems) и их важных классов появились в начале XX в. в работах А. Туэ (системы Туэ), М. Шёнфинкеля и Х. Карри (комбинаторная логика), Ж. Эрбрана и К. Гёделя (вычислимость по Эрбрану–Гёделю), А. Чёрча (λ-исчисление), Э. Поста, Ан. А. Маркова и других. В настоящее время системы преобразований термов широко применяются в информатике, в частности в функциональном программировании и в системах компьютерной алгебры. Изучение их свойств, таких, как терминируемость (обрыв любой цепочки преобразований по правилам системы), конфлюэнтность (соединение различных ветвей вычисления, имеющих общее начало), существование и единственность нормальных форм, составляет важную главу теоретической информатики (см. [1], [2]).

В настоящей работе мы рассматриваем системы, которые можно назвать *медленно сходящимися*. Любая цепочка преобразований в таких системах обрывается, т.е. система сходится (терминирует), но число шагов в ней нельзя оценить функцией разумного порядка роста от длины начального терма.

Для любой системы W определим ее функцию сложности $C_W(n)$ как максимальное возможное число шагов цепочки преобразований системы W, начинающейся с терма длины, не превосходящей n. (Если для некоторого n верхней границы на число шагов не существует, то система W не сходится.) Функция сложности рассматриваемых нами медленно сходящихся систем не

Исследование Л. Д. Беклемишева выполнено за счет гранта Российского научного фонда (проект № 14-50-00005) в Математическом институте им. В. А. Стеклова Российской академии наук. Параграфы 1–5 выполнены Л. Д. Беклемишевым, а § 6 – совместно Л. Д. Беклемишевым и А. А. Оноприенко.

является доказуемо тотальной в формальной арифметике Пеано, а полученная нижняя оценка возможного числа шагов превосходит любую функцию так называемой расширенной иерархии Гжегорчика вплоть до ординала ε_0 (см. [3]). В частности, сходимость таких систем не доказуема средствами формальной арифметики.

Существование медленно сходящихся систем преобразований термов следует из некоторых классических результатов математической логики. Хорошо известно, что системы преобразований термов и даже некоторые более специальные их подклассы, такие, как системы Туэ, представляют собой универсальную модель вычислений. В силу того, что множество доказуемых в PA предложений является перечислимым, мы можем рассмотреть вычислимый пересчет $(\varphi_n)_{n\in\mathbb{N}}$ всех вычислимых функций из $\mathbb{N}$ в $\mathbb{N}$, тотальность которых доказуема в PA. Тогда функция

$$\psi(n) = \max_{i,m\leqslant n} \varphi_i(m) + 1$$

удовлетворяет условию $\psi(n) > \varphi_k(n)$ для всех $n \geqslant k$. Поскольку ψ является вычислимой функцией, нетрудно построить конечную систему преобразований термов W, для которой $C_W(n) > \psi(n)$, т.е. C_W превосходит любую доказуемо тотальную в PA вычислимую функцию.

Примеры систем, возникающие при таком построении, опираются на способ кодирования доказательств, применяемый в теореме Гёделя о неполноте, поэтому они достаточно громоздки и их трудно выписать явно. Возникает вопрос: существуют ли естественные, просто формулируемые и легко запоминающиеся примеры таких систем? Ответ на этот вопрос можно получить на основе независимых комбинаторных утверждений, таких, как известный принцип Геракла и гидры (см. [4]).

Первый пример конечной системы преобразований термов, интерпретирующий игру Геракла и гидры, был сформулирован Н. Дершовицем (см. [5]). Эта система была позже скорректирована в работе [6], ее аккуратный анализ оказался довольно трудоемким.

Другая последовательность систем была предложена Э. Тузе (см. [7]). Системы Тузе допускают более простой анализ вопроса о сходимости, но соответствуют лишь ограниченным вариантам игры Геракла и гидры. В частности, сходимость каждой из этих систем доказуема в PA. Сила этих систем приближает PA снизу, но число аргументов функций сигнатуры, участвующих в этих системах, неограниченно возрастает.

Принцип червя (см. [8], [9]) – одно из самых простых независимых комбинаторных утверждений, близкое к утверждению о сходимости некоторой системы преобразований слов. Принцип червя также приводит к естественным примерам медленно сходящихся систем преобразований термов. Мы формулируем три такие системы. Первая система является системой преобразований слов, наиболее близко соответствующей недетерминированному варианту принципа червя. Правила этой системы очень просты, но ее алфавит и набор правил бесконечны (хотя и рекурсивны). Вторая система имеет конечный алфавит, также является легко запоминающейся, но ее набор правил по-прежнему бесконечен. Третья система, полученная из второй, конечна и допускает достаточно простой анализ сходимости, но она не настолько элементарна, как первые две.

Результаты настоящей работы могут быть сформулированы следующим образом.

ТЕОРЕМА 1. *Пусть W – одна из трех систем W_1, W_2 или W_3, описанных ниже. Тогда*:

(i) *W терминирует*;

(ii) *функция сложности $C_W(n)$ не ограничена никакой вычислимой функцией, доказуемо тотальной в* PA;

(iii) *терминируемость W не доказуема в* PA.

Результаты настоящей работы были впервые анонсированы в [10]. Одновременно В. Бухгольц (см. [11]) привел еще одну изящную кодировку игры Геракла и гидры, родственную системе Дершовица, и дал простое доказательство ее терминируемости. Поводом к опубликованию результатов [10] в настоящей работе послужило исследование Х. Цанкля, С. Винклер и А. Мидделдорпа [12], в котором системы W_i использовались в качестве тестовых примеров для разрабатываемых авторами средств компьютерного доказательства терминируемости систем преобразований термов. Им удалось получить доказательство сходимости с использованием компьютера для системы W_3. Также ими было обнаружено забытое в формулировке системы W_3 из [10] правило, которое необходимо для моделирования принципа червя. В настоящей работе мы приводим свое доказательство сходимости системы W_3 в исправленной формулировке.

§ 2. Системы преобразований термов

Сигнатурой мы будем называть множество Σ функциональных символов различной валентности (функциональные символы нулевой валентности – это константы). Помимо символов сигнатуры мы рассматриваем бесконечный алфавит переменных $\{x_0, x_1, \dots\}$, а также вспомогательные символы: скобки и запятую. Множество всех *термов* Tm_Σ сигнатуры Σ определяется индуктивно обычным образом: переменные и константы суть термы, и если $f \in \Sigma$ – функциональный символ валентности $n > 0$ и $t_1, \dots, t_n$ – термы, то выражение $f(t_1, \dots, t_n)$ есть терм. *Замкнутыми термами* мы называем термы, не имеющие вхождений переменных. Термы удобно представлять как конечные деревья, листья которых помечены переменными или константами, а внутренние вершины помечены символами сигнатуры. Валентность этих символов должна соответствовать ветвлению дерева в данной вершине.

Правилом преобразования термов называется выражение $l \to r$, где l и r суть термы. *Системой преобразований термов* сигнатуры Σ мы называем любое множество W таких правил для термов сигнатуры Σ.

Для того чтобы определить понятие вывода в данной системе преобразований термов, нам понадобятся понятия контекста и подстановки. *Контекстом* $t[x]$ мы называем терм, в который выделенная переменная x входит единственный раз. Результат замены переменной x в контексте t на терм s обозначаем $t[s]$.

Подстановка есть отображение $\sigma\colon \mathrm{Tm}_\Sigma \to \mathrm{Tm}_\Sigma$, коммутирующее со всеми символами сигнатуры:

$$(f(t_1, \dots, t_n))^\sigma = f(t_1^\sigma, \dots, t_n^\sigma).$$

Результат применения подстановки σ к терму t обозначаем t^σ. Фактически t^σ есть результат замены всех вхождений переменных терма t некоторыми термами.

Применением правила $l \to r$ мы называем преобразование терма $t[l^\sigma]$ в $t[r^\sigma]$ для некоторой подстановки σ и контекста $t[x]$. Для термов s и t пишем $s \to_W t$, если терм t получен из терма s с помощью применения одного из правил системы W. Если система W, о которой идет речь, понятна из контекста, то индекс W у стрелки будем опускать.

Выводом терма t_n *из терма* t_1 *в системе* W мы называем любую последовательность применений правил этой системы вида

$$t_1 \to_W t_2 \to_W \cdots \to_W t_{n-1} \to_W t_n.$$

Пишем $s \to_W^* t$, если существует вывод терма t из терма s в системе W.

Система преобразований термов W *сходится* (*терминирует*), если для любого начального терма t любая цепочка применений правил системы W, начинающаяся с t, обрывается.

Заметим, что системы преобразований слов, также известные как *односторонние системы подстановок Туэ*, можно рассматривать как особый вид систем преобразований термов в сигнатуре, содержащей лишь одноместные функциональные символы. Каждый из символов соответствует букве алфавита данной системы Туэ. Слову $A = a_1 \dots a_n$ в этом случае соответствует терм $A(x) = a_1(\cdots a_n(x)\cdots)$, где x – фиксированная переменная, а правилу подстановки $A \to B$ в системе Туэ соответствует правило преобразования термов $A(x) \to B(x)$. Понятия вывода в каждом из этих двух формализмов также соответствуют друг другу.

Для доказательства сходимости конкретных систем преобразований термов часто используют какой-нибудь фундированный порядок на термах, который подобран так, чтобы термы убывали при каждом применении правил. Одним из стандартных порядков является так называемый *лексикографический порядок по путям* (lexicographic path ordering; см. [5]).

Пусть на конечном множестве функциональных символов Σ системы W задан строгий линейный порядок $\prec$. Введем отношение порядка $\prec_{\mathrm{lpo}}$ на Tm_Σ (вместе с его рефлексивным замыканием $\preceq_{\mathrm{lpo}}$) индуктивно следующим образом.

Положим $s \succ_{\mathrm{lpo}} t$, если и только если t есть переменная, входящая в s, и $t \neq s$ или же $s = f(s_1, \dots, s_n)$, $t = g(t_1, \dots, t_m)$ и выполнено одно из условий:

1) $s_i \succeq_{\mathrm{lpo}} t$ для некоторого i, $1 \leqslant i \leqslant n$;
2) $f \succ g$ и $s \succ_{\mathrm{lpo}} t_j$ для всех j, $1 \leqslant j \leqslant m$;
3) $f = g$, $s \succ_{\mathrm{lpo}} t_j$ для всех j, $1 \leqslant j \leqslant m$, и для некоторого i, $1 \leqslant i \leqslant n$, имеет место $s_1 = t_1$, …, $s_{i-1} = t_{i-1}$, $s_i \succ_{\mathrm{lpo}} t_i$.

Как хорошо известно, порядок $\prec_{\mathrm{lpo}}$ фундирован. Кроме того, если каждое из правил $l \to r$ системы W удовлетворяет $l \succ_{\mathrm{lpo}} r$, то тем же свойством обладает и любое его применение $t[l^\sigma] \succ_{\mathrm{lpo}} t[r^\sigma]$, поэтому система W терминирует (см. [5], [13], [14]).

§ 3. Принцип червя

Сформулируем независимое комбинаторное утверждение, известное как *принцип червя* (см. [8], [9], [15]). Слова в алфавите натуральных чисел $\mathbb{N}$ будем называть *червями*; множество всех таких слов обозначаем S. Длину слова A обозначим $|A|$; *высотой* червя назовем его максимальную букву. *Размер* червя A есть максимум из его длины и высоты. Первый элемент червя $A = n_0 n_1 \dots n_k$ будем называть его *головой*. Обозначим через S_n множество всех слов в алфавите $\{i\colon i \geqslant n\}$.

Неформально говоря, жизнь любого червя определяется следующим процессом: если голова червя равна 0, то на очередном шаге она отмирает; в противном случае голова уменьшается на единицу, но червь регенерируется в соответствии с простым правилом, описанным ниже. Принцип червя утверждает, что ни один червь не живет бесконечно долго.

Более формально, мы зададим *функцию червя* $A\ m \longmapsto A[m]$, где $A = n_0 n_1 \dots n_k$ – червь и $m \in \mathbb{N}$ – параметр. Для любого начального червя A эта функция определяет последовательность $(A_i)_{i\in\mathbb{N}}$, которую мы называем его *эволюцией*:

$$A_0 := A, \qquad A_{i+1} := A_i[i+1].$$

Таким образом, параметр m на каждом шаге процесса увеличивается на единицу. Правила, определяющие $A[m]$, такие:

1) если $n_0 = 0$, то $A[m] := n_1 \dots n_k$; в этом случае голова червя отмирает;
2) если $n_0 = n+1 > 0$, то найдем максимальное начало $n_0 B$ червя A такое, что $B \in S_{n+1}$ (B может быть пустым); если $A = n_0 BC$, то

$$A[m] := (nB)^{m+1} C.$$

Пример 1. Рассмотрим эволюцию червя $A = 1302$. На первом шаге мы получаем $B = 3$ и $A_1 = A[1] = 030302$. Далее возникает последовательность

$$\begin{aligned} A_0 &= 1302,\\ A_1 &= 030302,\\ A_2 &= 30302,\\ A_3 &= 22220302,\\ A_4 &= (1222)^5 0302,\\ &\dots\dots\dots\dots\dots \end{aligned}$$

Заметим, что для любого слова A имеем $|A[m]| \leqslant |A| \cdot (m+1)$, поэтому длина слов A_i ограничена функцией $|A_i| \leqslant |A| \cdot (i+1)!$.

Принцип червя утверждает, что для любого начального слова A найдется $i \geqslant 0$ такое, что $A_i = \Lambda$.

Обозначим через $D(n)$ максимальное время жизни червей размера, не превосходящего n. Поскольку множество червей размера n конечно, принцип червя равносилен утверждению о том, что функция $D(n)$ определена для любого $n \in \mathbb{N}$. Следующая теорема вытекает из результатов [8], [9].

Теорема 2. (i) *Для любого $A \in S$ найдется такое i, что $A_i = \Lambda$.*
(ii) *Утверждение* (i) *не доказуемо в арифметике Пеано* PA.

(iii) *Функция $D(n)$ мажорирует любую доказуемо тотальную в* PA *вычислимую функцию.*

Доказательство. Для удобства читателя мы приведем доказательство утверждения (i). Доказательство утверждений (ii) и (iii) сложнее (см., например, [15]). Утверждение (ii) непосредственно следует из (iii).

Определим функцию $o\colon S \to \varepsilon_0$ и покажем, что при преобразованиях червя соответствующий ему ординал уменьшается. Обозначим через B^+ результат замены каждой буквы n в слове B на $n+1$, а через B^- – обратную операцию (определенную для слов $B \in S_1$). Значение функции $o(A)$ определяется индукцией по высоте червя A.

Если $A = 0^k$, то $o(A) := k$. Иначе слово A однозначно представляется в виде $A_1 0 A_2 0 \dots 0 A_n$, где в словах A_i нет нулей и не все они пусты. В этом случае полагаем $o(A) := \omega^{o(A_n^-)} + \dots + \omega^{o(A_2^-)} + \omega^{o(A_1^-)}$. Каждый из червей A_i^- имеет высоту, меньшую высоты A, поэтому применимо предположение индукции.

Заметим, что $o(\Lambda) = 0$ и $o(0A) = o(A) + 1$ для любого слова A. Если $B \neq 0^k$, то

$$o(B0A) = o(A) + o(B), \tag{3.1}$$

и если $B \in S_1$ непусто, то $o(B) = \omega^{o(B^-)}$.

Докажем, что $o(A) > o(A[m])$ для любого непустого слова A. Рассуждаем индукцией по высоте A. Для слов высоты 0 утверждение очевидно. Рассмотрим слово $A = A_1 0 A_2 0 \dots 0 A_k$, где $A_i \in S_1$ и не все A_i пусты. Обозначим $C := A_2 0 \dots 0 A_k$; тогда $A = A_1 0 C$, причем слова A_1 и C не могут быть оба пустыми.

Если $A_1 = \Lambda$, то $A = 0C$ и $A[m] = C$, поэтому $o(A) > o(A[m])$.

Пусть $A_1 \neq \Lambda$. Тогда $A[m] = (A_1[m])0C$. Слово $A_1[m]$ имеет вид 0^k лишь в случае $A_1 = 1$. Тогда $o(A) = o(C) + \omega > o(C) + k + 1 = o(A[m])$ и утверждение доказано. Иначе в силу (3.1) мы имеем

$$o(A) = o(C) + o(A_1), \tag{3.2}$$

$$o(A[m]) = o(C) + o(A_1[m]). \tag{3.3}$$

Поэтому нам достаточно установить $o(A_1) > o(A_1[m])$. Рассмотрим следующие два случая.

1) $A_1 = 1B$ для некоторого $B \in S_1$. Тогда $A_1[m] = (0B)^{m+1}$. Имеем

$$o(A_1) = \omega^{o((1B)^-)} = \omega^{o(B^-)+1}.$$

В то же время

$$o(A_1[m]) = \omega^{o(B^-)} \cdot (m+1) + 1 < \omega^{o(B^-)} \cdot \omega = o(A_1).$$

2) $A_1 = (n+1)B$ для некоторого $n > 1$. Тогда по определению функции червя $A_1^-[m] = (A_1[m])^-$. Отсюда по предположению индукции

$$o(A_1) = \omega^{o(A_1^-)} > \omega^{o(A_1^-[m])} = \omega^{o((A_1[m])^-)} = o(A_1[m]),$$

что и требовалось доказать.

§ 4. Система W_1

Эта система преобразований слов представляет собой недетерминированный вариант червя. Пусть $\Sigma = \{a_0, a_1, a_2, \dots\}$ – бесконечный алфавит, S_n – множество всех слов в алфавите $\{a_i \mid i \geqslant n\}$. Система W_1 определяется правилами

$$a_{n+1}A \to (a_nA)^k \quad \text{для всех} \quad A \in S_{n+1}, \quad k \geqslant 1. \tag{$*$}$$

Отметим несколько отличий вычислений в системе W_1 от преобразований червя.

Во-первых, преобразования могут происходить в любом месте, а не только в начале слова. Во-вторых, символ a_0 не удаляется. Поэтому вычисление завершается, если и только если возникает слово из одних букв a_0. В-третьих, число копирований k при применении правил ($*$) выбирается произвольно в каждом преобразовании. В-четвертых, подслово $A \in S_{n+1}$, к которому применяются правила ($*$), вообще говоря, не обязательно должно быть максимально длинным.

Нетрудно видеть, что эволюция любого червя B моделируется некоторой ветвью вычисления системы W_1. Действительно, мы будем применять правила ($*$) каждый раз к самому левому вхождению подслова вида $a_{n+1}A$, где $A \in S_{n+1}$ – максимально длинное, и выбирать число k соответственно номеру шага в эволюции червя B. По индукции легко доказать, что слова, выведенные таким образом в W_1 из B, отличаются от соответствующих червей лишь на некоторый префикс вида a_0^i. Значит, из терминируемости системы W_1 следует, что жизнь любого червя конечна.

Теорема 3. *Система W_1 терминирует.*

Доказательство. Используем то же отображение $o\colon S_0 \longrightarrow \varepsilon_0$ из слов в ординалы, что и ранее, т.е. положим $o(a_0^k) = k$ и $o(A_1a_0A_2a_0\dots a_0A_n) = \omega^{o(A_n^-)} + \dots + \omega^{o(A_1^-)}$, где все A_i принадлежат S_1 и не все они пусты, а B^- получается из $B \in S_1$ заменой всех букв a_{m+1} на a_m.

Лемма 1. *Пусть $A \to_{W_1} B$ – применение одного из правил системы W_1. Тогда $o(A) \geqslant o(B)$. Если же правило применялось к началу слова A, то $o(A) > o(B)$.*

Доказательство. Чтобы избежать лишних вычислений, воспользуемся стандартной интерпретацией слов как модальных формул системы GLP или ее строго позитивного фрагмента RC (см. [8], [16]). Для краткости мы отождествляем слова $a_{i_1}a_{i_2}\dots a_{i_n}$ и формулы системы GLP вида $\langle i_1\rangle\langle i_2\rangle\dots\langle i_n\rangle\top$. Запись $A \vdash B$ означает, что импликация $A \to B$ выводима в GLP.

Известно (см. [8; лемма 12]), что для любых слов $A, B \in S_0$

$$o(A) < o(B) \quad \Longleftrightarrow \quad B \vdash a_0A.$$

Поэтому для доказательства первого утверждения достаточно установить, что из $A \to_{W_1} B$ следует $A \vdash B$. (В этом случае мы имеем $B \nvdash a_0A$ в силу $A \nvdash a_0A$, откуда $o(A) \not< o(B)$.)

Индукцией по k сначала докажем, что $a_{n+1}AC \vdash (a_nA)^kC$ для любого $A \in S_{n+1}$. Базис индукции есть аксиома системы GLP. Предположим, что

утверждение верно для k, и докажем его для $k+1$. Заметим, что $a_{n+1}AC$ влечет $a_{n+1}A \wedge (a_nA)^kC$. По лемме 11, (ii) работы [8] эта формула эквивалентна $a_{n+1}A(a_nA)^kC$, что, очевидно, влечет $a_nA(a_nA)^kC$, что и требовалось доказать.

Отсюда следует, что и для любого слова D выводимо $Da_{n+1}AC \vdash D(a_nA)^kC$, тем самым, первое утверждение леммы доказано. Второе утверждение получается из следующего наблюдения.

Лемма 2. *Если слово B непусто, то $o(A) < o(BA)$.*

Доказательство. Очевидная индукция по длине B показывает, что $BA \vdash a_0A$.

Поскольку $a_{n+1}AC \vdash (a_nA)^kC$ для всех $k \geqslant 1$, мы получаем

$$o(a_{n+1}AC) \geqslant o((a_nA)^{k+1}C) > o((a_nA)^kC).$$

Тем самым, доказано второе утверждение леммы 1.

Предположим, что W_1 не терминирует, и мы имеем бесконечную последовательность преобразований $A_0 \to A_1 \to A_2 \to \cdots$. Используя лемму 2, можно считать, что $o(A_0) = o(A_1) = o(A_2) = \cdots$. Далее, среди всех таких последовательностей можно выбрать одну с минимальным ординалом $o(A_0)$. Так как $o(A_0) = o(A_1) = o(A_2) = \cdots$, то по лемме 1 все сокращения происходят не в начале слов, поэтому существует непустое B – максимальная общая начальная часть всех слов A_i, т.е. для любого $i \geqslant 0$ имеем $A_i = BA'_i$. Это означает, что все преобразования происходят правее B, т.е. имеем вывод $A'_0 \to A'_1 \to A'_2 \to \cdots$, где для любого i $o(A'_i) \leqslant o(A_i) = o(A_0)$. Из максимальности B следует, что для некоторого i преобразование $A'_i \to A'_{i+1}$ применяется к крайнему левому вхождению в A'_i. По лемме 1 имеем $o(A'_i) > o(A'_{i+1})$. Значит, для любого $k > i$ получаем $o(A'_k) < o(A_0)$ и имеем бесконечную цепочку преобразований $A'_{i+1} \to A'_{i+2} \to A'_{i+3} \to \cdots$, в которой $o(A'_{i+1}) < o(A_0)$. Это противоречит минимальности выбора $o(A_0)$.

§ 5. Система W_2

Сигнатура системы W_2 содержит константу 0, одноместный символ f и двуместный символ умножения $\cdot$. Система W_2 определяется следующими правилами:

$$\begin{cases} (x \cdot y) \cdot z \to x \cdot (y \cdot z), \\ f(0 \cdot x) \to (0 \cdot f(x))^m, \quad m \geqslant 1, \\ f(0) \to 0^m, \quad m \geqslant 1. \end{cases}$$

Здесь x^m обозначает терм $\underbrace{x \cdot (\cdots (x \cdot (x \cdot x)) \cdots)}_{m \text{ раз}}$.

Интуитивно f и $\cdot$ можно понимать как функции на словах из алфавита $\Sigma = \{a_0, a_1, a_2, \dots\}$, где 0 есть a_0, $x \cdot y$ обозначает конкатенацию слов x и y, а $f(x)$ есть результат замены всех букв a_i в слове x на a_{i+1}. Таким образом, значением каждого замкнутого терма t является некоторое непустое слово $A \in S_0$.

Слово A может быть найдено по t подсчетом f-глубины нулей. Назовем *f-глубиной* вхождения подтерма s в терм t количество символов f, лежащих

на пути от корня поддерева s к корню дерева t (не считая начала пути). Если n-й слева нуль находится в терме t на f-глубине k, то n-й символ в слове A равен a_k.

Определим отображение непустых слов $A \in S_0$ в замкнутые термы $A^\#$ сигнатуры W_2, при этом значение $A^\#$ будет равно A. Если $A = a_0^n$, то $A^\# := 0^n$. Иначе $A = A_1 a_0 A_2 a_0 \dots a_0 A_n$, где слова A_i принадлежат S_1 и не все из них пусты. Тогда

$$A^\# := f((A_1^-)^\#) \cdot 0 \cdot f((A_2^-)^\#) \cdot 0 \cdots 0 \cdot f((A_n^-)^\#),$$

где сомножитель $f((A_i^-)^\#)$ опускается, если слово A_i пусто. (Предполагается, что все скобки ассоциированы вправо, т.е. $x \cdot y \cdot z$ читаем как $x \cdot (y \cdot z)$.)

Пример 2. $(a_1 a_2 a_0 a_1)^\# = f((a_0 a_1)^\#) \cdot (0 \cdot f(a_0^\#)) = f(0 \cdot f(0)) \cdot (0 \cdot f(0))$.

Как и прежде, через $A[m]$ обозначаем функцию червя.

Лемма 3. *Пусть A непусто и не начинается с a_0. Тогда $A^\# \to_{W_2}^* A[m]^\#$.*

Доказательство. Поскольку A не начинается с a_0, терм $A^\#$ имеет вхождение подтерма вида $f(0 \cdot t)$ или $f(0)$. Выберем самое левое такое вхождение и применим к нему второе или третье правило с соответствующим m, после чего передвинем скобки вправо по первому правилу. Утверждается, что результатом будет $A[m]^\#$.

Действительно, если $C \in S_n$, то $C^\# = f^n(t)$ для некоторого t. Поэтому если слово B начинается с a_n, то $B^\#$ начинается с n символов f: $B^\# = f(f(\cdots f(t) \cdots) \cdots) \cdots$, причем область действия k-го символа f соответствует максимальному началу слова B, принадлежащему S_k. Следовательно, если $a_{n+1}B$ – максимальное начало слова A, принадлежащее S_{n+1}, то возможны два случая.

Если B пусто, то $A^\#$ содержит подтерм $f(0)$. (Это будет $(n+1)$-е вхождение символа f.) Заменим этот подтерм на 0^m по третьему правилу и нормализуем весь терм, передвигая скобки вправо, если это необходимо. Этим нулям предшествуют n символов f. Кроме того, f-глубина всех остальных нулей не изменилась. Следовательно, первый символ a_{n+1} в A заменился на $(a_n)^m$.

Если B непусто, то $A^\#$ содержит подтерм $f(0 \cdot t)$, которому предшествуют n символов f. Здесь $t = C^\#$, где C – результат уменьшения индексов каждой буквы в B на n. Заменим $f(0 \cdot t)$ на $(0 \cdot f(t))^m$ по второму правилу и нормализуем терм по первому правилу, если это необходимо. Это преобразование соответствует замене префикса $a_{n+1}B$ в A на $(a_n B)^m$.

Лемма доказана.

Следствие 1. *Функция сложности $C_{W_2}(n)$ не ограничена никакой доказуемо тотальной в* PA *вычислимой функцией.*

Доказательство. Докажем, что для некоторых примитивно рекурсивных (и поэтому доказуемо тотальных в PA) функций g, h имеет место неравенство $h(n, C_{W_2}(g(n))) \geqslant D(n)$, где $D(n)$ – функция времени жизни червя размера n. Пусть A – червь размера n и функция $g(n)$ дает верхнюю оценку возможной длины терма $A^\#$. Ясно, что g примитивно рекурсивна. Рассмотрим эволюцию червя A и выделим в последовательности A_i подпоследовательность A_{i_k}, состоящую из всех червей, не начинающихся с 0.

Заметим, что переход от A_{i_k} к $A_{i_{k+1}}$ состоит в применении одного правила и последующем удалении всех нулей в начале слова. Поэтому количество таких шагов ограничивается длиной слова A_{i_k+1} и из оценки длины червей мы получаем неравенство

$$i_{k+1} \leqslant i_k + 1 + n \cdot (i_k + 2)!.$$

Отсюда вытекает примитивно рекурсивная оценка i_k как функции k и n (обозначим эту оценку $I(n,k)$). По лемме 3 найдется последовательность термов B_k системы W_2 такая, что

$$B_0 \to^* B_1 \to^* B_2 \to^* \cdots \to^* B_k \to^* \cdots$$

и для каждого k терм B_k имеет вид $0^{m_k} \cdot A^{\#}_{i_k}$ для некоторого $m_k \in \mathbb{N}$. Таким образом, время жизни червя A оценивается величиной

$$i_N + |A_{i_N}| \leqslant i_N + n(i_N + 1)! \leqslant I(n,N) + n(I(n,N)+1)!,$$

где N – длина последовательности преобразований B_k. Обозначим через $h(n,N)$ правую часть неравенства. Поскольку N оценивается величиной $C_{W_2}(g(n))$, отсюда вытекает утверждение следствия.

Лемма 4. *Система W_2 терминирует.*

Доказательство. Мы докажем терминируемость системы W_2, применив лексикографический порядок по путям. Рассмотрим упорядочение символов сигнатуры $0 \prec \cdot \prec f$ и покажем, что все три схемы правил системы W_2 уменьшают терм в смысле порядка $\prec_{\mathrm{lpo}}$.

Действительно, для первого правила получаем $x \cdot y \succ_{\mathrm{lpo}} y$, откуда

$$(x \cdot y) \cdot z \succ_{\mathrm{lpo}} y \cdot z.$$

Поскольку $x \cdot y \succ_{\mathrm{lpo}} x$, мы выводим

$$(x \cdot y) \cdot z \succ_{\mathrm{lpo}} x \cdot (y \cdot z).$$

Утверждение $f(0 \cdot x) \succ_{\mathrm{lpo}} (0 \cdot f(x))^m$, докажем индукцией по m. Для $m = 1$ утверждение следует из того, что $f \succ \cdot$, $f(0 \cdot x) \succ_{\mathrm{lpo}} 0$ и $f(0 \cdot x) \succ_{\mathrm{lpo}} f(x)$. Для $m+1$ применяем условие $f \succ \cdot$, предположение индукции для m и только что доказанное нами утверждение $f(0 \cdot x) \succ_{\mathrm{lpo}} (0 \cdot f(x))$.

Утверждение $f(0) \succ_{\mathrm{lpo}} 0^m$ для любого $m \geqslant 1$ доказывается аналогично.

§ 6. Система W_3

Для того чтобы сформулировать медленно сходящуюся систему преобразований термов с конечным числом правил, мы модифицируем систему W_2. Теперь термы будут кодировать не только самого червя, но и номер шага его эволюции m, т.е. терм, кодирующий пару (m, A), преобразуется в терм, кодирующий пару $(m+1, A[m])$.

Введем новые унарные функциональные символы a, b, c, d, играющие роль маркеров. Пара (m, A) будет представлена как терм $da^m(t)$, где $t = A^{\#}$

(как в W_2), символ d всегда стоит в начале терма. Правила системы W_3 позволяют символу a переместиться внутрь терма t, где и должны происходить преобразования. Каждый из символов a выполняет одно копирование надлежащей части червя. После выполнения копирования каждый из символов a превращается в b. Дальнейшие преобразования возможны по правилам системы W_2, называемым здесь сокращениями, которые приводят к уменьшению ординала соответствующего W_2-терма. После выполнения одного сокращения возникает новый символ c. Символы c способны перемещаться в начало терма, при этом встреченные на пути символы b превращаются в c, а при взаимодействии с d символы c превращаются в a. Отсюда начинается новый цикл вычисления очередного шага эволюции червя.

Особенность системы W_3 состоит в том, что преобразования могут фактически выполняться не совсем в той последовательности, как описано выше. Поэтому доказательство сходимости системы W_3 требует детального анализа.

Правила системы W_3:

1) $(x \cdot y) \cdot z \longrightarrow x \cdot (y \cdot z)$;

2) (копирование)

$$a(f(0 \cdot x)) \longrightarrow b\big(f(0 \cdot x) \cdot (0 \cdot f(x))\big), \qquad a(f(0)) \longrightarrow b(f(0) \cdot 0);$$

3) (редукция)

$$f(0 \cdot x) \longrightarrow c(0 \cdot f(x)), \qquad f(0) \longrightarrow c(0), \qquad 0 \cdot x \longrightarrow c(x);$$

4) (a движется вглубь терма)

$$\begin{aligned} a(f(x)) &\longrightarrow f(a(x)), \qquad & a(x \cdot y) &\longrightarrow a(x) \cdot y, \\ a(b(x)) &\longrightarrow b(a(x)), \qquad & a(c(x)) &\longrightarrow c(a(x)); \end{aligned}$$

5) (c движется наружу и поглощает b)

$$f(c(x)) \longrightarrow c(f(x)), \qquad c(x) \cdot y \longrightarrow c(x \cdot y), \qquad b(c(x)) \longrightarrow c(c(x));$$

6) $d(c(x)) \longrightarrow d(a(x))$.

Сначала покажем, что W_3 моделирует эволюцию любого червя. Для краткости пишем $ab(t)$ вместо $a(b(t))$ и $a^n(t)$ вместо $a(a(\cdots a(t)\cdots))$.

Лемма 5. *Для любого* $n \geqslant 0$:
(i) $a^n f(0 \cdot x) \to^*_{W_3} c^{n+1}(0 \cdot f(x))^{n+1}$;
(ii) $a^n f(0) \to^*_{W_3} c^{n+1}(0^{n+1})$.

Доказательство. Докажем утверждение (i); утверждение (ii) доказывается аналогично, но проще. Рассмотрим вывод, в котором символы a по очереди продвигаются направо к вхождению $f(0 \cdot x)$, превращаются в b и присоединяют справа копии терма $(0 \cdot f(x))$. Затем применяется правило редукции и возникает символ c. Этот символ движется влево, поглощая n символов b, и скобки

перераспределяются вправо:

$$\begin{aligned} a^n f(0\cdot x) &\longrightarrow a^{n-1}b(f(0\cdot x)\cdot(0\cdot f(x))) \longrightarrow^* a^{n-2}b\big(a(f(0\cdot x))\cdot(0\cdot f(x))\big)\\ &\longrightarrow a^{n-2}b\big(b(f(0\cdot x)\cdot(0\cdot f(x)))\cdot(0\cdot f(x))\big)\\ &\longrightarrow^* b\big(\cdots b(b(f(0\cdot x)\cdot(0\cdot f(x)))\cdot(0\cdot f(x))\cdots(0\cdot f(x)))\big)\\ &\longrightarrow b\big(\cdots b(b(c(0\cdot f(x))\cdot(0\cdot f(x)))\cdot(0\cdot f(x))\cdots(0\cdot f(x)))\big)\\ &\longrightarrow^* c^{n+1}(0\cdot f(x))^{n+1}. \end{aligned}$$

Лемма доказана.

Следующая лемма показывает, что W_3 моделирует один шаг эволюции червя.

Лемма 6. *Пусть слово A непусто и отлично от a_0. Тогда для любого $n \geqslant 1$*

$$da^n(A^\#) \longrightarrow^*_{W_3} da^{n+1}(A[n]^\#).$$

Доказательство. Возможны два случая.

Случай 1: слово A начинается с a_0, т.е. $A^\# = 0\cdot t$. В этом случае получаем

$$da^n(0\cdot t) \longrightarrow da^n c(t) \longrightarrow^* dca^n(t) \longrightarrow^* da^{n+1}(t).$$

Случай 2: слово A начинается не с a_0. В этом случае самое левое вхождение 0 в терм $A^\#$ находится в контексте $f(0\cdot t)$ или $f(0)$. Тогда терм $A[n]^\#$ получается из $A^\#$ заменой этих вхождений соответственно на $(0\cdot f(t))^{n+1}$ или на 0^{n+1}. Рассмотрим без ограничения общности первый случай.

Продвинем символы a вправо до первого вхождения $f(0\cdot t)$. По лемме 5 мы можем преобразовать вхождение $a^n f(0\cdot t)$ в терм $c^{n+1}(0\cdot f(t))^{n+1}$. Затем передвинем c^{n+1} в начало терма, получим $dc^{n+1}(A[n]^\#)$. Наконец, из $dc^{n+1}(A[n]^\#)$ получаем $da^{n+1}(A[n]^\#)$.

Следствие 2. *Функция сложности $C_{W_3}(n)$ не ограничена никакой доказуемо тотальной в* PA *вычислимой функцией.*

Доказательство. Рассмотрим эволюцию червя $A = A_0$:

$$A_0 \to A_1 \to A_2 \to \cdots \to A_k \to \cdots.$$

По лемме 6 получаем вывод в системе W_3

$$daA_0^\# \longrightarrow^* da^2A_1^\# \longrightarrow^* da^3A_2^\# \longrightarrow^* \cdots \longrightarrow^* da^{k+1}A_k^\# \longrightarrow^* \cdots.$$

Этот вывод можно продолжать вплоть до предпоследнего шага эволюции червя A (на предпоследнем шаге возникает слово a_0). Каждый шаг эволюции червя моделируется, как минимум, одним применением правил системы W_3. Значит, длина данного вывода в системе W_3 не меньше, чем длина эволюции червя (с точностью до одного шага).

Теорема 4. *Система W_3 терминирует.*

Доказательство. Достаточно доказать, что в системе W_3 не существует бесконечных цепочек преобразований замкнутых термов. В самом деле, из любого вывода в нашей системе можно получить вывод той же длины, заменив все вхождения переменных на 0.

Докажем теперь несколько вспомогательных утверждений.

Лемма 7. *Если терм t не содержит символа d, то не существует бесконечной цепочки преобразований, начинающейся с t.*

Доказательство. Сначала заметим, что правило 6) не используется в выводах из терма t, поскольку буква d не может в них возникнуть. Теперь упорядочим символы исходного алфавита: $0 \prec c \prec \cdot \prec f \prec b \prec a$. Прямая проверка показывает, что все правила системы W_3, кроме правила 6), уменьшают терм в смысле соответствующего лексикографического порядка по путям. Таким образом, любая цепочка преобразований по этим правилам обрывается.

Рассмотрим теперь специальный случай, когда начальный терм имеет вид $d(t)$, где t не содержит символа d и переменных. В этом случае и все остальные термы в цепочке преобразований имеют такой же вид. Самое первое (левое) вхождение нуля в такой терм будем называть *критическим*. Термы будем представлять деревьями, растущими вверх. *Стволом* терма назовем путь от корня до критической вершины этого дерева, т.е. самый левый путь в дереве терма. *Критическими редукциями* (соответственно *копированиями*) назовем преобразования редукции (соответственно копирования), применяемые к критическому вхождению 0. Ключевую роль играет следующая лемма.

Лемма 8. *Всякая бесконечная последовательность преобразований системы W_3 вида $dt_0 \to dt_1 \to dt_2 \to \cdots$, где термы t_i замкнуты и не содержат d, содержит критическую редукцию.*

Доказательство. Допустим, что в бесконечной цепочке преобразований нет критических редукций. Сделаем следующие наблюдения.

1. Через n_a обозначим количество букв a на стволе дерева (и аналогично для b и c). Тогда сумма $n_a + n_b + n_c$ постоянна для всех термов вывода.

Заметим, что преобразования групп 2), 4)–6) таковы, что никакие буквы a, b, c, находящиеся справа от ствола терма, на него не попадут (a движется только вверх, c движется вниз, но только по левой ветви). Те вхождения a, b, c, которые уже были на стволе, остаются на нем или же переименовываются в порядке $a \to b \to c \to a$ без изменения суммы $n_a + n_b + n_c$. Преобразование 1) лишь сокращает один символ умножения на стволе. Преобразования редукции 3) меняют метки на стволе, только если применяются к критическому вхождению; в противном случае возникшая буква c не попадет на ствол.

2. Количество критических копирований в данной цепочке преобразований конечно.

Пусть m_b обозначает количество символов b на стволе, находящихся ниже некоторого символа c. Тогда величина $n_a + m_b + n_c$ не возрастает и, кроме того, убывает на 1 при каждом критическом копировании. Действительно, символы b, возникающие при критических копированиях, находятся выше любых символов c на стволе и вычитаются из суммы.

3. Если в цепочке преобразований нет критических копирований и редукций, то величина n_a не убывает. Поскольку сумма $n_a + n_b + n_c$ постоянна, то n_a постоянна начиная с некоторого момента. Это означает, что в дальнейших преобразованиях правило 6) не применяется. В этом случае мы уже знаем по лемме 7, что цепочка преобразований обрывается.

Определим отображение $\overline{v}$ из множества замкнутых термов системы W_3, не содержащих d, в слова в алфавите $\Sigma = \{a_0, a_1, \dots\}$. Это отображение аналогично интерпретации замкнутых термов системы W_2 как слов, но является его модификацией, учитывающей положение некоторых символов c, входящих в данный терм.

Расширим алфавит Σ бесконечной серией новых символов c_i, т.е. положим $\Sigma' = \Sigma \cup \{c_0, c_1, c_2, \dots\}$. Для слов A в алфавите Σ' определено отображение A^+, повышающее индекс каждой буквы на единицу. Для каждого $n \geqslant 0$ обозначим через S'_n множество всех слов в алфавите $\{a_i, c_i : i \geqslant n\}$. Как и ранее, S_n обозначает множество всех слов в алфавите $\{a_i : i \geqslant n\}$.

Пусть t – замкнутый терм W_3, не содержащий d. Сопоставим терму t слово $v(t)$ в алфавите Σ' индуктивно следующим образом:

$$\begin{gathered} v(0) = a_0, \qquad v(f(t)) = v(t)^+, \qquad v(t \cdot s) = v(t)v(s), \\ v(c(t)) = c_0 v(t), \qquad v(at) = v(bt) = v(t). \end{gathered} \tag{6.1}$$

Пусть теперь $A = A'a_i \in S_0$. Положим $A \circ c_j = A$, если $i \leqslant j$, и $A \circ c_j = Aa_j$, если $i > j$. Также положим $A \circ c_j = \Lambda$, если A пусто. Определим отображение $w\colon S'_0 \to S_0$ следующим образом:

$$w(\Lambda) = \Lambda, \qquad w(Aa_i) = w(A)a_i, \qquad w(Ac_i) = w(A) \circ c_i.$$

Наконец, положим $\overline{v}(t) = w(v(t))$ и $\overline{o}(t) = o(\overline{v}(t))$, где o – ординал слова системы W_1.

Заметим, что для любых слов $A, B \in S'_0$ имеет место $w(Aa_iB) = w(A)w(a_iB)$, что нетрудно установить индукцией по длине B. Мы будем часто пользоваться этим замечанием в доказательстве следующей леммы, не оговаривая это отдельно.

Лемма 9. *Пусть термы t_1, t_2 замкнуты и не содержат d. Если $dt_1 \to_{W_3} dt_2$, то $\overline{o}(dt_2) \leqslant \overline{o}(dt_1)$. Если $t_1 \to_{W_3} t_2$ – критическая редукция, то*

$$\overline{o}(dt_1) < \overline{o}(dt_2).$$

Доказательство. Прежде всего заметим, что любому подтерму s терма t при отображении v соответствует непустое подслово A слова $v(t) = CAD$. Если f-глубина вхождения s в t равна n, то A получается из $v(s)$ n-кратным применением функции $(\,\cdot\,)^+$. Самый правый символ в A соответствует самому правому вхождению 0 в s и, тем самым, есть a_i для некоторого i. Следовательно, слово $\overline{v}(t) = w(CAD)$ имеет вид $w(C)A'D'$ для некоторых слов A' и D', получающихся сокращениями и переименованиями некоторых из символов c_j, входящих в A и D соответственно. (A' и D', вообще говоря, не обязательно должны совпадать с $w(A)$ и $w(D)$.)

Для доказательства первого утверждения, как и в лемме 1, достаточно установить, что $\overline{v}(t_1) \vdash \overline{v}(t_2)$. Рассмотрим применения всех правил системы W_3.

1. Применения правил 1), 4) и 6) не меняют слово $v(t_1)$, а значит, и слово $\overline{v}(t_1)$.

2. *Правило* $af(0) \to b(f(0) \cdot 0)$. Пусть $v(t_1) = Ca_{n+1}D$; тогда $v(t_2) = Ca_{n+1}a_nD$. Индукцией по длине D докажем, что

$$w(a_{n+1}D) \vdash w(a_{n+1}a_nD),$$

откуда вытекает $w(C)w(a_{n+1}D) \vdash w(C)w(a_{n+1}a_nD)$, что и дает требуемое. Базис индукции сводится к очевидному утверждению $a_{n+1} \vdash a_{n+1}a_n$. Рассмотрим шаг индукции.

Если $D = a_iD_1$, то $w(a_{n+1}D) = a_{n+1}w(D)$. Отсюда

$$w(a_{n+1}D) \vdash a_{n+1} \wedge a_nw(D) \vdash a_{n+1}a_nw(D) = w(a_{n+1}a_nD).$$

Если $D = c_iD_1$, то рассмотрим следующие случаи.

Если $i > n$, то по предположению индукции

$$w(a_{n+1}D) = w(a_{n+1}D_1) \vdash w(a_{n+1}a_nD_1) = w(a_{n+1}a_nD).$$

Если $i = n$, то

$$w(a_{n+1}D) = a_{n+1}w(a_nD_1) = a_{n+1}w(a_nc_nD_1) = w(a_{n+1}a_nD).$$

Если $i < n$, то

$$w(a_{n+1}D) = a_{n+1}w(a_iD_1) \vdash a_{n+1} \wedge a_nw(a_iD_1) \vdash a_{n+1}a_nw(a_iD_1) = w(a_{n+1}a_nD).$$

3. *Правило* $a(f(0 \cdot x)) \to b(f(0 \cdot x) \cdot (0 \cdot f(x)))$. В этом случае $v(t_1)$ имеет вид $Ca_{n+1}AD$, где $A \in S'_{n+1}$ заканчивается на a_i для некоторого i, а $v(t_2) = Ca_{n+1}Aa_nAD$. Мы имеем $w(a_{n+1}AD) = a_{n+1}A'D'$ для некоторых $A' \in S_{n+1}$ и D' и $w(a_{n+1}Aa_nAD) = a_{n+1}A'a_nA'D'$ для тех же самых A' и D'. Второе вхождение A' здесь совпадает с первым, поскольку в A нет символов c_j при $j \leqslant n$. Аналогично, D' во втором слове то же, что и в первом, поскольку последний символ в A' тот же a_i, что и в A.

Как и выше в лемме 1, мы получаем

$$a_{n+1}A'D' \vdash a_{n+1}A' \wedge a_nA'D' \vdash a_{n+1}A'a_nA'D',$$

откуда вытекает требуемое.

4. *Правило* $f(0 \cdot x) \to c(0 \cdot f(x))$. В этом случае $v(t_1)$ имеет вид $Ca_{n+1}AD$, где $A \in S'_{n+1}$ и $v(t_2) = Cc_na_nAD$. Тогда

$$w(Ca_{n+1}AD) = w(C)a_{n+1}A'D'$$

для некоторых $A' \in S_{n+1}$ и D'. В свою очередь $w(Cc_na_nAD)$ совпадает или с $w(C)a_na_nA'D'$, если c_n не сокращается, или со словом $w(C)a_nA'D'$ в противном случае.

Мы имеем $a_{n+1}A'D' \vdash a_na_nA'D' \vdash a_nA'D'$, что дает требуемое в любом из этих двух случаев.

Заметим также, что в случае критического применения этого правила слово C пустое и c_n сокращается, поэтому $\overline{v}(t_1) \vdash a_n\overline{v}(t_2)$, а значит, и $\overline{o}(t_1) > \overline{o}(t_2)$.

5. *Правило* $f(0) \to c(0)$. В этом случае $v(t_1)$ имеет вид $Ca_{n+1}D$, а $v(t_2) = Cc_na_nD$. Этот случай аналогичен предыдущему.

6. *Правило* $0 \cdot x \to c(x)$. В этом случае $v(t_1)$ имеет вид Ca_nAD, а $v(t_2) = Cc_nAD$. Пусть $w(C) = C'a_i$. Если $n < i$, то $w(Cc_nAD) = w(Ca_nAD)$, и доказывать нечего. Если $n \geqslant i$, то $w(Cc_nAD) = C'w(a_iA)D'$ и $w(Ca_nAD) = C'w(a_nA)D'$. При этом D' одно и то же в обоих словах, поскольку слово A заканчивается на букву вида a_k, которая не сокращается в $w(a_nA)$. Требуемое утверждение вытекает из следующей леммы.

Лемма 10. *Для любых $n \geqslant i$ и слов A и D' $w(a_nA)D' \vdash w(a_iA)D'$.*

Доказательство. Рассуждаем индукцией по длине A. Если слово A пусто, то утверждение очевидно.

Если $A = a_jA'$, то

$$w(a_nA)D' = a_nw(a_jA')D' \vdash a_iw(a_jA')D' = w(a_iA)D'.$$

Если $A = c_jA'$, то рассмотрим три случая.

Если $j < i$, то

$$w(a_nA)D' = a_nw(a_jA')D' \vdash a_iw(a_jA')D' = w(a_iA)D'.$$

Если $j \geqslant n$, то по предположению индукции

$$w(a_nA)D' = w(a_nA')D' \vdash w(a_iA')D' = w(a_iA)D'.$$

Если $i \leqslant j < n$, то по предположению индукции

$$w(a_nA)D' = a_nw(a_jA')D' \vdash w(a_jA')D' \vdash w(a_iA')D' = w(a_iA)D'.$$

Лемма доказана.

Также заметим, что в случае критического применения этого правила $\overline{v}(t_1) = a_0\overline{v}(t_2)$, поэтому $\overline{o}(t_1) > \overline{o}(t_2)$.

7. *Правило* $f(c(x)) \to c(f(x))$. В этом случае $v(t_1)$ имеет вид $Cc_{n+1}AD$, где $A \in S'_{n+1}$, а $v(t_2) = Cc_nAD$. Пусть $w(C) = C'a_i$; тогда $w(Cc_{n+1}AD)$ имеет вид $C'w(a_ic_{n+1}A)D'$, а $w(Cc_nAD) = C'w(a_ic_nA)D'$. Достаточно доказать

$$w(a_ic_{n+1}A)D' \vdash w(a_ic_nA)D'.$$

Рассмотрим три случая. Если $i \leqslant n$, то

$$w(a_ic_{n+1}A)D' = w(a_iA)D' = w(a_ic_nA)D'.$$

Если $i > n + 1$, то по лемме 10

$$w(a_ic_{n+1}A)D' = a_iw(a_{n+1}A)D' \vdash a_iw(a_nA)D'.$$

Если $i = n + 1$, то по лемме 10

$$w(a_iA)D' \vdash w(a_nA)D',$$

откуда

$$w(a_ic_{n+1}A)D' = w(a_iA)D' \vdash a_i \wedge w(a_nA)D' \vdash a_iw(a_nA)D' = w(a_ic_nA)D'.$$

8. *Правило* $c(x) \cdot y \to c(x \cdot y)$. Оно не изменяет терм $v(t_1)$.

9. *Правило* $b(c(x)) \to c(c(x))$. В этом случае $v(t_1)$ имеет вид Cc_nAD, где $A \in S'_n$, а $v(t_2) = Cc_nc_nAD$. Пусть $w(C) = C'a_i$; тогда $w(Cc_nAD)$ имеет вид $C'w(a_ic_nA)D'$, а $w(Cc_nc_nAD) = C'w(a_ic_nc_nA)D'$. Достаточно доказать

$$w(a_ic_nA)D' \vdash w(a_ic_nc_nA)D'.$$

Если $i > n$, то

$$w(a_i c_n A)D' = w(a_i a_n A)D' \vdash a_i \wedge a_n w(a_n A)D' \vdash a_i a_n w(a_n A)D' = w(a_i c_n c_n A)D'.$$

Если $i \leqslant n$, то

$$w(a_i c_n A)D' = w(a_i A)D' = w(a_i c_n c_n A)D'.$$

Лемма 9 доказана.

Из лемм 8 и 9 мы получаем

Следствие 3. *Пусть d не входит в терм t. Тогда в системе W_3 не существует бесконечной цепочки преобразований, начинающейся с терма dt.*

Замечание 1. Усложнения в определении функций $\overline{v}$ и $\overline{o}$ и в доказательстве леммы 9 связаны с некритическими применениями правила сокращения нуля: $0 \cdot x \to c(x)$. Пример $f(0) \cdot (0 \cdot f(0)) \to f(0) \cdot f(0)$ показывает, что перевод термов системы W_3 в термы системы W_2, забывающий все служебные символы (включая c), может приводить к увеличению ординалов соответствующих слов. В системе W_2 и системе W_1 аналогичное правило сокращения отсутствует, и эта проблема не возникает.

Завершим доказательство теоремы 4. Индукцией по количеству вхождений символов d в произвольный терм s докажем, что не существует бесконечной цепочки преобразований $s = s_0 \to s_1 \to \cdots$. Без ограничения общности считаем s замкнутым термом. Случай, когда s не содержит символ d, был нами уже разобран.

Рассмотрим самое внутреннее вхождение символа d в s, т.е. представим s в виде $s = u[d(t)]$, где t не содержит d. Заметим, что любое преобразование терма s по правилам W_3 либо происходит в подтерме $d(t)$, либо оставляет этот подтерм неизменным. В последнем случае преобразование $s \to s_1$ сводится к преобразованию контекста $u[x] \to u_1$ по тому же правилу и к последующей замене всех вхождений переменной x в u_1 на $d(t)$. (Таких вхождений x в u_1 может быть более одного.) Следовательно, с каждым термом s_i можно связать некоторый терм u_i и конечную последовательность термов вида $dt_{i1}, dt_{i2}, \ldots, dt_{ik_i}$ такую, что s_i есть результат замены последовательных вхождений переменной x в терм u_i на dt_{ij} (термы t_{ij} не содержат d). При этом последовательность термов u_i получается применением тех же правил, что и s_i, за исключением применений, которые производятся в выделенных термах dt_{ij} и не изменяют u_i.

Поскольку $u = u_0$ содержит меньшее число вхождений символа d, чем s, то по предположению индукции начиная с некоторого момента n терм u_i не меняется. Значит, для $i \geqslant n$ каждое преобразование производится в одном из термов $dt_{i1}, dt_{i2}, \ldots, dt_{ik}$ при фиксированном $k = k_n$. По следствию 3 цепочка преобразований в каждом из термов dt_{ij} обрывается, значит, обрывается и вся последовательность в целом.

Авторы выражают благодарность М. Р. Пентусу, нашедшему несколько неточностей в первой версии настоящей работы.

Список литературы

[1] J. W. Klop, “Term rewriting systems”, *Handbook of logic in computer science*, v. 2, Handb. Log. Comput. Sci., **2**, Oxford Univ. Press, New York, 1992, 1–116.

[2] *Term rewriting systems. Terese*, Cambridge Tracts Theoret. Comput. Sci., **55**, eds. M. Bezem, J. W. Klop, R. de Vrijer, Cambridge Univ. Press, Cambridge, 2003, xxii+884 pp.

[3] H. E. Rose, *Subrecursion*: *functions and hierarchies*, Oxford Logic Guides, **9**, The Clarendon Press, Oxford Univ. Press, New York, 1984, xiii+191 pp.

[4] L. Kirby, J. Paris, "Accessible independence results for Peano arithmetic", *Bull. London Math. Soc.*, **14**:4 (1982), 285–293.

[5] N. Dershowitz, J.-P. Jouannaud, "Rewrite systems", *Handbook of theoretical computer science*, v. B, Elsevier, Amsterdam, 1990, 243–320.

[6] N. Dershowitz, G. Moser, "The hydra battle revisited", *Rewriting computation and proof*, Lecture Notes in Comput. Sci., **4600**, Springer, Berlin, 2007, 1–27.

[7] H. Touzet, "Encoding the hydra battle as a rewrite system", *Mathematical foundations of computer science* 1998 (Brno), Lecture Notes in Comput. Sci., **1450**, Springer, Berlin, 1998, 267–276.

[8] L. D. Beklemishev, "The worm principle", *Logic Colloquium*'02, Lect. Notes Log., **27**, Assoc. Symbol. Logic, La Jolla, 2006, 75–95.

[9] M. Hamano, M. Okada, "A relationship among Gentzen's proof-reduction, Kirbi–Paris' hydra game, and Buchholz's hydra game", *Math. Logic Quart.*, **43**:1 (1997), 103–120.

[10] L. D. Beklemishev, "Representing worms as a term rewriting system", Mini-workshop: Logic, combinatorics and independence results, Report No. 52/2006, Mathematisches Forschungsinstitut Oberwolfach, *Oberwolfach Rep.*, **3**:4 (2006), 3093–3095.

[11] W. Buchholz, "Another rewrite system for the standard hydra battle", Mini-workshop: Logic, combinatorics and independence results, Report No. 52/2006, *Oberwolfach Rep.*, **3**:4 (2006), 3099–3101.

[12] H. Zankl, S. Winkler, A. Middeldorp, "Beyond polynomials and Peano arithmetic – automation of elementary and ordinal interpretations", *J. Symbolic Comput.*, **69** (2015), 129–158.

[13] F. Baader, T. Nipkow, *Term rewriting and all that*, Cambridge Univ. Press, Cambridge, 1998, xii+301 pp.

[14] J. H. Gallier, "What's so special about Kruskal's theorem and the ordinal Γ_0? A survey of some results in proof theory", *Ann. Pure Appl. Logic*, **53**:3 (1991), 199–260.

[15] Л. Д. Беклемишев, "Схемы рефлексии и алгебры доказуемости в формальной арифметике", *УМН*, **60**:2(362) (2005), 3–78; англ. пер.: L. D. Beklemishev, "Reflection principles and provability algebras in formal arithmetic", *Russian Math. Surveys*, **60**:2 (2005), 197–268.

[16] L. Beklemishev, "Calibrating provability logic: from modal logic to reflection calculus", Advances in modal logic, **9**, Coll. Publ., London, 2012, 89–94.

Лев Дмитриевич Беклемишев
(Lev D. Beklemishev)
Математический институт им. В. А. Стеклова
Российской академии наук, г. Москва
E-mail: bekl@mi.ras.ru

Анастасия Александровна Оноприенко
(Anastasija A. Onoprienko)
Механико-математический факультет,
Московский государственный университет
имени М. В. Ломоносова
E-mail: ansidiana@yandex.ru

Поступила в редакцию
25.03.2015 и 21.06.2015