ТЕОРИЯ АЛГОРИТМОВ И ПРОГРАММ

Сборник научных трудов

Министерство высшего и среднего специального образования
Латвийской ССР
Латвийский ордена Трудового Красного Знамени
государственный университет имени Петра Стучки
Вычислительный центр

ТЕОРИЯ АЛГОРИТМОВ И ПРОГРАММ

СБОРНИК НАУЧНЫХ ТРУДОВ

Латвийский государственный университет им. П.Стучки
Рига 1986

УДК 51.01:518.5

ТЕОРИЯ АЛГОРИТМОВ И ПРОГРАММ

Теория алгоритмов и программ: Сборник научных трудов.- / Отв. ред. Я.М. Барздинь.- Рига: ЛГУ им. П.Стучки, 1986. - 195с.

Сборник посвящен исследованию различных типов вычислительных устройств, таких как альтернирующие и вероятностные машины, а также проблемам индуктивного синтеза программ.

Сборник рассчитан на научных работников, занимающихся или интересующихся теорией алгоритмов и программ, аспирантов и студентов.

Библ. назв. - 44.

Печатается по решению Издательского совета
ЛГУ им. П.Стучки

Т 20205-068у
М 812(II)-86 17.86.1500000000

ВВЕДЕНИЕ

Теория алгоритмов и программ служит основным источником идей и теоретической базой работ по развитию математического обеспечения ЭВМ. В последнее время многие теоретические результаты нашли важные применения. Поэтому не случайно, что по теории алгоритмов и программ ведутся интенсивные исследования как в нашей стране, так и за рубежом.

Данный сборник посвящается различным вопросам этой теории. Часть работ относится к таким хорошо известным разделам теории алгоритмов, как теория сложности вычисления и проблеме эквивалентности многоленточных автоматов. Здесь, в частности, получены новые существенные результаты, касающиеся вычислений на вероятностных машинах.

В сборнике рассмотрены и более нетрадиционные разделы теории алгоритмов и программ, например алгебраическая теория баз данных.

Большая часть сборника посвящена такому совершенно новому направлению, как автоматический синтез программ. Здесь впервые с полным доказательством излагаются результаты по индуктивному синтезу программ, полученные в ВЦ ЛГУ им. П. Стучки и получившие широкую известность как в нашей стране, так и за рубежом.

О ВЕРОЯТНОСТНЫХ И ДЕТЕРМИНИРОВАННЫХ МАШИНАХ ТЬЮРИНГА СО ВХОДОМ И ВЫХОДОМ

Р.В.Фрейвалд

ВЦ ЛГУ им.П.Стучки

В статье доказывается, что вероятностная одноленточная одноголовочная машина Тьюринга со входом и выходом может в реальное время со сколь угодно высокой вероятностью $1-\varepsilon$ ($\varepsilon > 0$) распознавать язык, распознавание которого на детерминированных машинах такого же типа требует почти квадратичное время. Этот результат был опубликован в / 1 /, однако там технически наиболее сложная часть доказательства, а именно доказательство того, что рассматриваемый язык трудно распознаваем на детерминированных машинах, была дана в очень конспективной форме. Было сказано, что используется метод доказательства, примененный в / 2 /. В настоящей статье дается полное изложение доказательства.

Детерминированная одноленточная одноголовочная машина Тьюринга со входом и выходом - это шестерка $\langle X, Y, Z, S, q_1, I \rangle$, где

X - конечный алфавит (буквы на входе), включающий символ $\#$;

Y - конечный алфавит (буквы на рабочей ленте), включающий символ Λ ;

Z - конечный алфавит (буквы на выходе), включающий символ Λ ;

S - конечный алфавит (внутренние состояния);

q_1 - инициальное состояние ($q_1 \in S$);

I - множество инструкций. Каждая инструкция является цепочкой символов из $X \times Y \times S \times \{\rightarrow\} \times S \times Y \times Z \times \{C, R, L\}$. Первые три символа инструкции называются ее левой частью, а последние четыре символа - ее правой частью. Требуется, чтобы для любой цепочки из $X \times Y \times S$ во множестве I нашлась одна и только одна инструкция с такой левой частью.

У этой машины имеется одна бесконечная в обе стороны рабочая лента, по которой передвигается одна головка. В начале работы машина находится в состоянии q_1, на вход поступает первая буква рассматриваемого слова, рабочая лента пуста. Далее, на втором шаге на вход поступает вторая буква слова, на третьем шаге - третья и т.д. После поступления на вход последней буквы слова начинают поступать символы # . Пусть на очередном шаге работы машины состояние равно $q_k \in S$, головка обозревает букву $y \in Y$, и на вход поступает символ $x \in X$. Пусть множество I содержит инструкцию $x\, y\, q_k \to q_{k'}\, y'\, z\, \xi$. Тогда машина на этом шаге переходит в состояние $q_{k'}$, заменяет букву y на y', выдает на выход z и, если $\xi = R$ ($\xi = L$), то головка двигается на одну ячейку вправо (влево) и, если $\xi = C$, остается на месте. Данное слово принимается, если первый отличающийся от Λ символ, выданный машиной на выход после поступления на вход последней буквы входного слова, равен I. Слово отвергается, если этот символ равен 0.

Определение вероятностной машины отличается от определения детерминированной машины тем, что в левой части инструкций появляется еще один символ - выходное значение датчика случайных чисел с конечным алфавитом, который на каждом шаге работы выдает свои выходные значения равновероятно и по схеме Бернулли, т.е. независимо от значений, выданных на других шагах. Требование, чтобы для каждой возможной левой части инструкции (дополненной теперь еще одним символом) во множестве нашлась одна и только одна инструкция с такой левой частью, сохраняется.

Для каждой заканчивающейся реализации работы вероятностной машины можно подсчитать вероятность этой реализации. Вероятность результата у при работе машины $\mathcal{M}$ на x - это сумма вероятностей реализаций работы $\mathcal{M}$ на x, при которых вырабатывается результат у.

Будем говорить, что вероятностная машина $\mathcal{M}$ распознает язык L с вероятностью p ($p > I/2$), если $\mathcal{M}$, при работе на произвольном x_0 с вероятностью не меньшей чем p,

принимает x, если $x \in L$, и отвергает x, если $x \in \bar{L}$.

Вероятностная машина распознает язык L за время $t(x)$ с вероятностью p $(p > 1/2)$, если для любого слова x с вероятностью не меньшей, чем p, выполняется следующее событие: машина, работая на x, останавливается не позже времени $t(x)$ с результатом

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

Рассмотрим язык D, который состоит из всевозможных слов вида

$$x \stackrel{\varepsilon}{=} y\,2\,y\,2\,y\,2 \ldots 2y,$$

где y - произвольное симметричное слово в алфавите $\{0,1\}$, и слово x содержит $\lfloor \log_2 |y| \rfloor$ букв 2, где $|y|$ - длина слова y.

ЛЕММА 1. Для любого $\varepsilon > 0$ существует вероятностная одноленточная одноголовочная машина Тьюринга со входом и выходом, которая распознает язык D в реальное время так, что машина принимает любое слово из D с вероятностью 1 и отвергает любое слово из D с вероятностью $1-\varepsilon$.

ДОКАЗАТЕЛЬСТВО. Конструкция требуемой машины $\mathcal{M}_\varepsilon$ зависит от ε только выбором следующей константы c_ε. Константа c_ε определена так, что она превышает длину всех слов языка D, у которых число символов 2 в слове не больше, чем $2 - 2\log_2 \varepsilon$. Таких слов только конечное число. Машина $\mathcal{M}_\varepsilon$ делит все слова на короткие (не больше чем c_ε букв) и длинные (больше чем c_ε букв). На коротких словах машина выдает верные результаты. Пусть на вход поступает длинное слово

$$y_0\,2\,y_1\,2\,y_2\,2 \ldots 2\,y_n.$$

Тогда $n > 2 - 2\cdot\log_2 \varepsilon$ и $2^{1-n/2} < \varepsilon$. Машина $\mathcal{M}_\varepsilon$ записывает y_0 на ленту со сжатием двух символов в одну ячейку. Потом параллельно выполняются следующие два действия.

I) Проверка условия $n = \lfloor \log_2 |y_0| \rfloor$. (Для этого на специально выделенном "этале" ленты при обработке y_0 записывается массив из $\lfloor |y_0|/2 \rfloor$ единиц. При обработке

каждого следующего подслова y_i ($i \in \{1,2,\dots,n\}$) зачеркиваются 1-я, 3-я, 5-я,... из оставшихся единиц этого массива. Равенство $n = [\log_2 |y_0|]$ выполнено тогда и только тогда, когда последняя единица зачеркивается при обработке y_n.)

2) Сравнение каждого очередного y_i либо с y_0, либо с обращением y_0, либо свертывание y_i "змейкой" (см. рис. 1 и 2), проверка на симметричность и одновременно проверка на совпадение со всеми предыдущими y_j, которые обрабатывались в таком режиме. (Точнее, если до сих пор не обнаружилось отличие длин подслов $y_0, y_1, y_2, \dots$, то в начале обработки следующего y_i головка находится либо на начале записи y_0, либо на конце; в первом случае y_i обязательно сравнивается с y_0, во втором случае с вероятностью 1/2 y_i сравнивается с обращением y_0 и с вероятностью 1/2 сравнивается на симметричность и на совпадение со всеми y_i, которые тоже свертывались "змейкой".)

$y_i(n+1)$	$y_i(n+2)$	$y_i(n+3)$	...	$y_i(2n-1)$	$y_i(2n)$
$y_i(n)$	$y_i(n-1)$	$y_i(n-2)$	...	$y_i(2)$	$y_i(1)$
$*y_0(1)y_0(2)$	$y_0(3)y_0(4)$	$y_0(5)y_0(6)$	...	$y_0(2n-3)y_0(2n-2)$	$y_0(2n-1)y_0(2n)*$

Рис. 1.

—	$y_i(n+2)$	$y_i(n+3)$	...	$y_i(2n-1)$	$y_i(2n)$	$y_i(2n+1)$
$y_i(n+1)$	$y_i(n)$	$y_i(n-1)$	...	$y_i(3)$	$y_i(2)$	$y_i(1)$
$*y_0(1)y_0(2)$	$y_0(3)y_0(4)$	$y_0(5)y_0(6)$	...	$y_0(2n-3)y_0(2n-2)$	$y_0(2n-1)y_0(2n)$	$y_0(2n+1)*$

Рис. 2.

Если $x \in D$, то $\mathcal{M}_2$ достоверно выдает правильный ответ "принадлежит". Если $x \in \overline{D}$, и слово содержит два подслова y_i и y_j различной длины или число букв 2 отлично от $[\log_2 |y_0|]$, то $\mathcal{M}_2$ также достоверно выдает правильный ответ "не принадлежит". Остается рассмотреть слу-

чай, когда $x \in \bar{D}$, все подслова y_i имеют одинаковую длину ℓ и число букв 2 равно $[\log_2 \ell]$.

Машина $\mathcal{M}_\varepsilon$ обращается к датчику случайных чисел лишь в начале обработки некоторых из подслов $y_1, y_2, \dots, y_n$ (да и то не всех). Назовем способом обработки данного слова x (зависящим от работы датчика случайных чисел) слово $\alpha_1, \alpha_2, \dots, \alpha_n$, где α_i равно 0, если y_i сравнивается с y_0, α_i равно 1, если y_i сравнивается с обращением y_0, и $\alpha_i = 2$, если y_i свертывается "змейкой".

Пусть q – число нулей в $\alpha_1\alpha_2 \dots \alpha_n$. Тогда вероятность способа $\alpha_1\alpha_2 \dots \alpha_n$, очевидно, равна 2^{q-n}. Так как в любом $\alpha_1\alpha_2 \dots \alpha_n$ число нулей не превышает $1+n/2$, то вероятность одного фиксированного способа не больше чем $2^{1-n/2}$.

Покажем теперь, что если на некотором слове x машина $\mathcal{M}_\varepsilon$ выдает результат "принадлежит" при двух различных способах обработки, то $x \in D$. Из этого будет вытекать утверждение леммы, так как $2^{1-n/2} < \varepsilon$.

Пусть $\alpha_1\alpha_2 \dots \alpha_n$ и $\beta_1\beta_2 \dots \beta_n$ – два различных способа обработки. Пусть i – первое такое j, что α_j и β_j отличаются. Так как $\alpha_i \neq 0$ и $\beta_i \neq 0$, то без ущерба для общности считаем, что $\alpha_i = 1$ и $\beta_i = 2$. Так как $\beta_i = 2$, то y_i симметрично. Так как $\alpha_i = 1$, то y_i совпадает с обращением y_0, а следовательно, и с самим y_0. Так как $\beta_i = 2$, то с y_i совпадают все y_j при тех j, что $\beta_j = 2$. Так как y_i совпадает как с y_0, так и с обращением y_0, то y_i совпадает также со всеми y_k при тех k, что $\beta_k = 0$ или $\beta_k = 1$. Следовательно, все y_m ($m \in \{1, 2, \dots, n\}$) совпадают и симметричны, т.е. $x \in D$. Лемма доказана.

Нераспознаваемость языка D в реальное время на детерминированных одноленточных одноголовочных машинах Тьюринга со входом и выходом доказывается методикой, известной под названием "техника следов". Рассмотрим ряд понятий, подготавливающих применение этой методики.

Без ущерба для общности можно считать, что наши машины имеют ленту бесконечную только в одну сторону (вправо). Известно, что это ограничение не увеличивает существенно время работы машины.

i-ю слева ячейку ленты ($i=0,1,2,\ldots$) обозначим через $\mathcal{Q}_i$. Предположим, что в начале работы лента пуста, а головка находится в ячейке $\mathcal{Q}_0$. Точкой i на ленте назовем границу между ячейками $\mathcal{Q}_i$ и $\mathcal{Q}_{i+1}$.

При $i<j$ через $\langle i,j\rangle$ обозначим зону, состоящую из всех $j-i+1$ ячеек, расположенных между точками $i-1$, j. При $j=\infty$ через $\langle i,j\rangle$ обозначим бесконечную зону, расположенную правее точки $i-1$. Помимо общей нумерации ячеек ленты на каждой зоне может рассматриваться своя внутренняя нумерация ячеек (также слева направо).

Конфигурацией зоны $\langle i,j\rangle$ (отнесенной к такту τ) называется функция, указывающая для каждого натурального n, не большего длины зоны: 1) какая буква вписана в ячейку с внутренним номером n зоны $\langle i,j\rangle$, и если эта ячейка обозревается головкой, то 2) в каком состоянии; 3) входную букву в момент τ; 4) выходную букву в момент τ.

Конфигурацией (отнесенной к такту τ) машины называется конфигурация зоны $\langle 0,\infty\rangle$.

Пусть при переработке машиной слова x головка в первый раз переходит точку d в состоянии $q(1)$, затем в состоянии $q(2)$ и т.д. Тогда мы будем говорить, что слово x имеет в точке d след $q(1)q(2)\ldots$ Длиной следа будем называть длину слова $q(1)q(2)\ldots$, т.е. число переходов этой точки. Если переходов нет, то говорим, что след пуст.

Определим процедуру разделения слова x на куски относительно точки d. Рассмотрим работу машины при подаче на вход слова x. Ту часть слова x, которая поступает на вход между i-м и $(i+1)$-м переходом головкой точки d, назовем i-м куском слова x ($i\geqslant 0$). Так как в начале работы головка находилась на $\mathcal{Q}_0$, нетрудно заметить, что буквы из кусков с четными номерами появились на входе, когда головка находилась левее точки d, а буквы из кусков с нечетными номерами, когда головка была правее.

Рассмотрим следующую конструкцию. Разделим слово x_1 на куски относительно точки d_1, и слово x_2 - на куски относительно точки d_2. Если количество кусков в обоих случаях совпадает, то возьмем куски с четными номерами из x_1, куски

с нечетными номерами из x_2 и соединим их в порядке возрастания номеров. Слово, полученное в результате такой конструкции, обозначим через $S(x_1, x_2, d_1, d_2)$.

ЛЕММА 2. Пусть машина работает на словах x_1 и x_2 так, что след слова x_1 в точке d_1 совпадает со следом слова x_2 в точке d_2. Тогда машина на слове $S(x_1, x_2, d_1, d_2)$ работает так, что конфигурация зоны $\langle 0, d_1 \rangle$ (отнесенная к такту, когда на вход подается последняя буква слова) совпадает с конфигурацией зоны $\langle 0, d_1 \rangle$ при работе машины на x_1, а конфигурация зоны $\langle d_1+1, \infty \rangle$ работы на $S(x_1, x_2, d_1, d_2)$ совпадает с конфигурацией зоны $\langle d_2+1, \infty \rangle$ работы машины на x_2.

Пусть y - произвольное симметричное слово в алфавите $\{0,1\}$, а слово $x = y2y2\ldots2y$ принадлежит D. Введем такую функцию t, что $|x| = t(|y|)$. Подадим слово x на вход машины. Совокупность ячеек, где головка находилась хотя бы один раз, пока на вход подавалась первая половина первого вхождения подслова y в слово x, назовем исходной зоной слова x.

$\sqrt{t(|x|)}$ ячеек направо от исходной зоны назовем контрольной зоной этого слова (см. рис. 3). Работа машины занимает не более, чем $t(|x|)$ шагов. Следовательно, для каждого слова в контрольной зоне найдется по крайней мере одна такая точка, что головка перешла ее не более $\frac{t(|x|)}{\sqrt{t(|x|)}} = \sqrt{t(|x|)}$ раз. Зафиксируем для каждого слова x одну такую точку. Назовем ее контрольной точкой слова x.

Пусть число состояний машины равно a. Все слова данной длины $|x|$ имеют в своих контрольных точках следы, длины которых не превосходит $\sqrt{t(|x|)}$. Поэтому слова длины $|x|$ могут иметь в своих контрольных точках не более

$$\sqrt{t(|x|)} \cdot a^{1+\sqrt{t(|x|)}} \le a^{2\sqrt{t(|x|)}} = 2^{2\cdot\log_2 a \cdot \sqrt{t(|x|)}}$$

различных следов.

Пусть M - некоторое множество слов длины $\lambda(2\lambda)$ из языка D, α - некоторый след, s_1, s_2, s_3, s_4 - некоторые числа, не превосходящие $\lambda(2\lambda)$.

Обозначим через $M(\alpha, s_1, s_2, s_3, s_4)$ подмножество всех тех слов множества M, для которых:

а) след в контрольной точке этого слова равен α;

б) суммарное количество букв первого вхождения слова y в слово x в кусках с четными номерами (при делении на куски относительно контрольной точки) равно s_1;

в) перед самым длинным из кусков с четными номерами, относящимся к первому вхождению слова y в слово x (если таких несколько, то - перед первым из них; если в некоторый кусок попадают и буквы из дальнейших вхождений слова y, то они не учитываются), в слове имеется ровно s_2 букв;

г) среди первых s_2 букв слова ровно s_3 принадлежат кускам с четными номерами;

д) первое вхождение буквы 2 в слово x встречается в куске с номером s_4.

Число слов в любом множестве T будем обозначать через $|T|$.

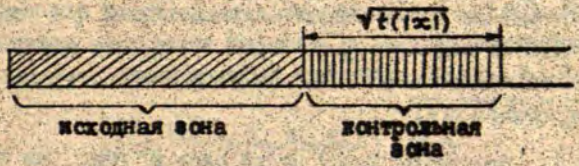


Рис. 3.

ЛЕММА 3 ($\alpha, s_1, s_2, s_3, s_4$) можно зафиксировать так, что

$$\left|M(\alpha, s_1, s_2, s_3, s_4)\right| \geq \frac{|M|}{2^{c_0 \cdot \sqrt{t(z(2\lambda))}}},$$

где C_0 - некоторая величина, зависящая лишь от параметров машины.

ДОКАЗАТЕЛЬСТВО. В контрольных точках слов множества M может быть не более чем $2^{2 \cdot \log_2 a \cdot \sqrt{t(z(2\lambda))}}$ различных следов. Каждый из параметров s_1, s_2, s_3 не превышает 2λ, а параметр s_4 не превышает $z(2\lambda)$. Следовательно, имеется не более чем $2^{2 \cdot \log_2 a \cdot \sqrt{t(z(2\lambda))}} \cdot (2\lambda)^3 \cdot z(2\lambda) \leq 2^{c_0 \cdot \sqrt{t(z(2\lambda))}}$ пятерок $(\alpha, s_1, s_2, s_3, s_4)$. Каждое слово из M входит в не-

которое из $M(\alpha, s_1, s_2, s_3, s_4)$, отсюда и вытекает утверждение леммы.

Зафиксируем теперь одну детерминированную одноленточную одноголовочную машину со входом и выходом $\mathcal{M}$, распознающую язык D.

ЛЕММА 4. Если на вход машины $\mathcal{M}$ поданы разные слова языка D, то до момента подачи на вход первой буквы 2 их конфигурации различны.

ДОКАЗАТЕЛЬСТВО. Действительно, пусть $\mathcal{M}$ при подаче двух различных начал слов языка D переходит в одинаковые конфигурации. Это значит, что машина не различает эти начала y' и y''. Пусть x_0 - такое слово в алфавите $\{0,1,2\}$, что $y'2\,x_0 \in D$. (Такое x_0 определяется однозначно.) Машина $\mathcal{M}$ принимает слово $y'2\,x_0$. Но тогда $\mathcal{M}$ принимает и слово $y''2\,x_0 \notin D$. Противоречие. Лемма доказана.

Пусть M_{c_1} - множество, содержащее те и только те слова x, которые принадлежат D, длина которых равна $z(2\lambda)$ и у которых во время подачи на вход второй половины первого вхождения подслова y в слово x головка машины отходит правее контрольной точки на расстояние $\psi(x) \geqslant c_1 \cdot \sqrt{t(z(2\lambda))}$.

ЛЕММА 5. При любом $c_1 > 0$ число слов в M_{c_1} не превосходит $2^{\lambda-(c_1-c_0)\cdot\sqrt{t(z(2\lambda))}}$.

ДОКАЗАТЕЛЬСТВО. Покажем сперва, что при любой фиксации $(\alpha, s_1, s_2, s_3, s_4)$ у всех слов множества $M(\alpha, s_1, s_2, s_3, s_4)$ совпадают подслова длины $c_1 \cdot \sqrt{t(z(2\lambda))}$, начинающиеся (s_2+1)-ой буквой (ниже будем называть их С-подсловами).

Пусть $x_1, x_2 \in M_{c_1}(\alpha, s_1, s_2, s_3, s_4)$. Обозначим реализации работы $\mathcal{M}$ на x_1 и x_2 через ρ_1, ρ_2. Так как у x_1 и x_2 совпадают следы в их контрольных точках, можно построить составленное слово $x' = S(x_1, x_2; \rho_1, \rho_2;$ контр.точка$_1$, контр.точка$_2)$. Слово x' обладает следующими свойствами:

I) $x' \in D$. (Согласно лемме 2 машина на x' выдает результат "принадлежит", ибо на каждом из x_1 и x_2 она выдает такой результат.)

2) Начало слова x' до первого вхождения буквы 2 имеет такую же длину 2λ, как и соответствующие начала слов x_1 и x_2 (первое вхождение буквы 2 встречается у слов x_1 и x_2 в кусках с одинаковым номером, и суммарное количество букв из указанного начала в кусках с четными номерами у этих слов совпадает). Это показывает, что центр симметрии начала слова x' не сместился.

3) У слов x' и x_1 совпадают C-подслова (первая половина начала слова x' до первого вхождения буквы 2 совпадает с первой половиной начала слова x_1, так как во время подачи букв первой половины слова x_1 головка находилась в исходной зоне, т.е. левее контрольной точки. Из симметричности начал слов x' и x_1 следует, что и вторые половины начал слов совпадают, в том числе совпадают и C-подслова).

4) C-подслова совпадают также у x' и x_2. (C-подслова по определению принадлежит кускам с четными номерами, притом в словах x_1 и x_2 перед C-подсловами находится одинаковое число букв из кусков с четными номерами.)

Следовательно, C-подслова совпадают также у x_1 и x_2, а значит и у всех слов из $M_{C_1}(\alpha, s_1, s_2, s_3, s_4)$. Из симметричности начал всех слов $M_{C_1}(\alpha, s_1, s_2, s_3, s_4)$ и совпадения C-подслов следует, что при любой фиксации $(\alpha, s_1, s_2, s_3, s_4)$

$$\left| M_{C_1}(\alpha, s_1, s_2, s_3, s_4) \right| \leq 2^{\lambda - c_1 \cdot \sqrt{t(\lambda(2\lambda))}}.$$

По лемме 3 можно так зафиксировать $(\alpha, s_1, s_2, s_3, s_4)$, что

$$\left| M_{C_1}(\alpha, s_1, s_2, s_3, s_4) \right| \geq \frac{|M_{C_1}|}{2^{c_0 \cdot \sqrt{t(\lambda(2\lambda))}}}.$$

Из двух последних неравенств вытекает утверждение леммы.

СЛЕДСТВИЕ. Можно так зафиксировать $C_1^{(0)}$, что

$$|M_{C_1^{(0)}}| = \theta\left(2^{\lambda - \sqrt{t(\lambda(2\lambda))}}\right).$$

Глубиной ячейки исходной зоны назовем ее расстояние от правого конца исходной зоны (см. рис. 4).

Через M^- обозначим множество всех слов x языка D, длина которых равна $z(2\lambda)$, и для которых максимальная глубина ячеек, посещенных головкой во время подачи второй половины первого вхождения подслова y в слово x, $\xi(x)$ меньше чем $\lambda/3\log_2 b$ (b – число букв в алфавите ленты, включая пустой символ).

ЛЕММА 6. $(\exists C_3>0)(\exists\lambda_0)(\forall\lambda>\lambda_0)\left[|M^-| \geqslant 2^{\lambda-\sqrt{t(z(2\lambda))}} \longrightarrow \right.$

$\left. \longrightarrow t(z(2\lambda)) > c_3\cdot\lambda^2 \right]$.

исходная зона

Рис. 4.

ДОКАЗАТЕЛЬСТВО. Пограничной зоной будем называть зону, занимающую $\frac{\lambda}{3\log_2 b}$ самых правых ячеек исходной зоны и еще $(1+c_1^{(0)})\sqrt{t(z(2\lambda))}$ ячеек правее ее (см.рис.5). Очевидно, в пограничной зоне возможно не более $2^{\lambda+o(\lambda)+(1+c_1^{(0)})\cdot\sqrt{t(z(2\lambda))}}$ различных конфигураций.

Рассмотрим множество $M' = M^- \setminus M_{c_1^{(0)}}$.

$$|M'| = |M^- \setminus M_{c_1^{(0)}}| \geqslant 2^{\lambda-\sqrt{t(z(2\lambda))}} - o\left(2^{\lambda-\sqrt{t(z(2\lambda))}}\right). \quad (I)$$

Пусть K_1 и K_2 – две конфигурации пограничной зоны. Обозначим через $M'(K_1, K_2)$ множество всех слов из M',

у которых конфигурация пограничной зоны в момент λ равна K_1, а в момент 2λ равна K_2.

Каждое слово из M' попадает в некоторое $M'(K_1, K_2)$, поэтому можно зафиксировать K_1 и K_2 так, что

$$\left|M'(K_1,K_2)\right| \geqslant \frac{|M'|}{2^{\frac{1}{3}\lambda + o(\lambda) + 2\cdot(1+c_1^{(\infty)})\sqrt{t(z(2\lambda))}}}.$$

Однако, если бы $M'(K_1, K_2)$ содержало два различных слова x_1 и x_2, то слово, составленное из начала слова x_1 (до момента λ) и конца слова x_2 (начиная с момента λ), имело бы в момент 2λ такую же конфигурацию, как x_1, что противоречит лемме 4.

Поэтому $|M'(K_1, K_2)| \leqslant 1$

и $$|M'| \leqslant 2^{\frac{1}{3}\lambda + o(\lambda) + 2\cdot(1+c_1^{(\infty)})\sqrt{t(z(2\lambda))}}.$$

Сравнение этого неравенства с (I) дает

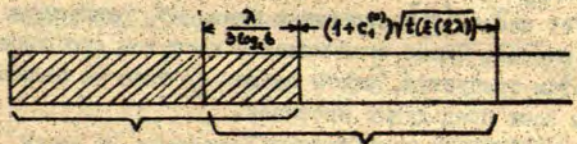


Рис. 5.

$$t(z(2\lambda)) \geqslant \frac{\lambda}{6\,(1+c_1^{(\infty)})} - o(\lambda).$$

Лемма доказана.

Наша цель - доказать, что время работы любой детерминированной одноленточной одноголовочной машины Тьюринга со входом и выходом, распознающей язык D, для бесконечно многих слов x не меньше, чем $\mathrm{const}\cdot|x|^2 / \log|x|$.

Если для некоторой такой машины для бесконечно многих λ имеет место $|M^-| \geq 2^{\lambda - \sqrt{t(z(2\lambda))}}$, то нужное утверждение вытекает из леммы 5. Остается доказать утверждение для тех машин, у которых $(\exists\lambda_0)(\forall\lambda>\lambda_0)(|M^-| < 2^{\lambda - \sqrt{t(z(2\lambda))}})$.

Далее мы рассмотрим только слова специального вида и усилим леммы 5 и 6, формулируя их применительно к этим словам. Доказательства проводятся аналогично, только в подсчетах учитывается, что все рассматриваемые множества содержат существенно меньше слов.

Специальным словом с параметрами $d; c; \delta_1, \delta_2, \dots, \delta_n$ назовем следующее слово из нулей и единиц:

Первые d букв $y_1, y_2 \dots y_d$ произвольны - они составляют первый массив свободных букв. Следующие δ_1 букв называются связанными и подчиняются отношению $y_{d+i} = y_{d-i+1}$ $(i=1,2,\dots,\delta_1)$. После массива связанных букв следуют $\left[\frac{d}{c}\right]$ свободных, т.е. произвольных, потом δ_2 связанных, т.е. подчиняющихся $y_{e+i} = y_{e-i+1}$ $(i = 1,2,\dots,\delta_2;\ e = d + \delta_1 + \left[\frac{d}{c}\right])$. Потом следует снова $\left[\frac{d}{c}\right]$ свободных букв, δ_3 связанных и т. д. (см. рис.6). Слово оканчивается массивом свободных букв. При этом требуется, чтобы любое δ_j было не больше общей длины всех предыдущих массивов.

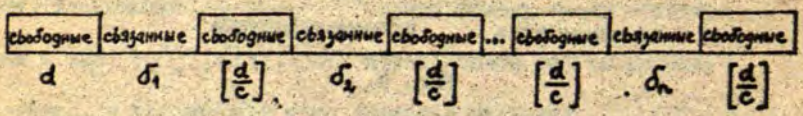


Рис.6.

Пусть w - специальное слово с параметрами $d; c; \delta_1, \delta_2, \dots, \delta_n$. Приведенной длиной слова w называется число свободных букв в нем, т.е. $d + n\cdot\left[\frac{d}{c}\right]$.

Обозначим через M множество слов языка D, у которых первая половина каждого подслова y - специальное слово, причем параметры $d; c; \delta_1, \delta_2, \dots, \delta_n$ совпадают у всех слов

множества M. Приведенную длину $d + n \cdot [\frac{d}{c}]$ этих слов обозначим через ℓ. Длину подслов y этих слов обозначим через 2λ, а длину самих слов – через $t(2\lambda)$. Число слов в M равно 2^{ℓ}.

Обозначим через M_{c_1} подмножество M, содержащее те и только те слова, у которых во время подачи второй половины подслова y головка нашей фиксированной машины отходит правее контрольной точки на расстояние $\psi(w) \geq c_1 \cdot \sqrt{t(t(2\lambda))}$.

ЛЕММА 7. При любом $c_1 > 0$ число слов в M_{c_1} не превосходит $2^{\ell - (c_1 \cdot 2^{-n} - c_0) \cdot \sqrt{t(t(2\lambda))}}$.

ДОКАЗАТЕЛЬСТВО отличается от доказательства леммы 5 в основном лишь тем, что там совпадение C-подслов у всех слов некоторого множества означало совпадение такого же количества (свободных) букв в первой половине подслов y. Здесь надо учитывать, что одна свободная буква может соответствовать сразу $\leq 2^n$ буквам C-подслов.

СЛЕДСТВИЕ. Можно так зафиксировать $C_1^{(n)}$, что

$$|M_{C_1^{(n)}}| = o\left(2^{\ell - \sqrt{t(t(2\lambda))}}\right).$$

Через M^- обозначим подмножество множества M, состоящее из всех слов, для которых максимальная глубина $\xi(w)$ ячеек, посещенных головкой во время подачи второй половины первого вхождения подслова y в рассматриваемое слово, меньше чем $\ell / 3 \cdot \log_2 6$.

ЛЕММА 8. $(\exists c_3 > 0)(\forall n)(\forall c)(\exists d_0)(\forall d > d_0)(\forall \delta_1, \delta_2, \ldots, \delta_n)$

$$\left[|M^-| \geq 2^{\ell - \sqrt{t(t(2\lambda))}} \longrightarrow t(t(2\lambda)) > c_3 \cdot \lambda^2\right].$$

Выше были рассмотрены специальные слова, у которых параметр n принимает значения 1,2,3,... В дальнейшем, чтобы считать лемму 5 частным случаем леммы 7 и лемму 6 частным случаем леммы 8, под специальным словом с параметром $n=0$ будем понимать произвольное слово (в алфавите $\{0,1\}$) длины d.

Зеркальное отображение любого слова w будем обозначать через $mi(w)$. Слово, образованное первыми δ буквами слова $mi(w)$, будем обозначать через $mi(w)_\delta$.

ЛЕММА 9. $[(\forall n)(\forall c)(\exists \tilde{d})(\forall d > \tilde{d})(\forall \delta_1, \delta_2, \dots, \delta_n)$

$$\left(|M^-| < 2^{\ell - \sqrt{t(z(2d))}}\right)] \to (\exists c_4 > 0)(\forall \tau_0)(\exists \tau > \tau_0)(t(\tau) > c_4 \cdot \tau^3 / \log_2 \tau).$$

ДОКАЗАТЕЛЬСТВО. Пусть дано:

$$(\forall n)(\forall c)(\exists \tilde{d})(\forall d > \tilde{d})(\forall \delta_1, \delta_2, \dots, \delta_n)(|M^-| < 2^{\ell - \sqrt{t(z(2d))}}) \quad (2)$$

Возьмем $c = 3 \cdot \log_2 6$ и рассмотрим конечное число различных $n = 0, 1, 2, \dots, 6 \cdot (\log_2 6)^2$. Пусть d_0 настолько велико, что при указанных n и c для всех $d \geq d_0$ формула (2) истинна. Обозначим через R_0 множество всех слов длины d_0 в алфавите $\{0, I\}$.

Пусть R_0' будет подмножеством всех тех слов w из R_0, что

$$\xi(w \cdot mi(w)) \geq \frac{d_0}{3 \cdot \log_2 6} \; ; \; \psi(w \cdot mi(w)) < c_1^0 \cdot \sqrt{t(z(2d_0))}.$$

По лемме 5 и формуле (2)

$$|R_0'| \geq 2^{d_0} - 2 \cdot 2^{d_0 - \sqrt{t(z(2d_0))}}.$$

Отнесем к $R_0'(\delta)$ $(\delta = 1, 2, \dots, d_0)$ все те слова $w \in R_0'$, что в результате подачи которых на вход слова $w \cdot mi(w)_\delta$ головка заходит в исходную зону слова $w \cdot mi(w)$ (т.е. в зону, где головка находилась, пока на вход подавалось слово w) на глубину не меньшую чем $\frac{d_0}{3 \cdot \log_2 6}$.

Зафиксируем некоторое δ_1 так, чтобы

$$|R_0'(\delta_1)| \geq \frac{2^{d_0} - 2 \cdot 2^{d_0 - \sqrt{t(z(2d_0))}}}{d_0} \geq 2^{d_0 - 2\log_2 d_0}.$$

R_1 определим как множество всех слов вида $w \cdot mi(w)_{\delta_1} \cdot v$, где $w \in R_0'(\delta_1)$, а v – произвольное слово (в алфавите $\{0,1\}$) длины $\frac{d_0}{3 \cdot \log_2 b}$. Для R_1 справедливы следующие утверждения:

1) все слова из R_1 – специальные слова с одинаковыми параметрами $d_0; 3\log_2 b; \delta_1$ (и, следовательно, с одинаковой длиной $d_1 = d_0 + \delta_1 + \frac{d_0}{3 \cdot \log_2 b}$);

2) $|R_1| \geq 2^{d_0 + \frac{d_0}{3 \cdot \log_2 b} - 2\log_2 d_0}$;

3) пока подается любое слово из R_1, головка побывает не более чем на $m_1 = d_0 + c_1^{(0)} \sqrt{t(z(2d_0))}$ ячейках ленты.

Аналогичным способом из R_1 получим R_2. Для этого определяем R_1' как множество всех тех слов $w \in R_1$, что

$$\xi(w \cdot mi(w)) \geq \frac{d_0}{3 \cdot \log_2 b}, \quad \psi(w \cdot mi(w)) < c_1^{(1)} \cdot \sqrt{t(z(2d_1))}.$$

По лемме 7 и формуле (2)

$$|R_1'| \geq 2^{d_0 + \frac{d_0}{3 \cdot \log_2 b} - 2\log_2 d_0} - 2 \cdot 2^{d_0 + \frac{d_0}{3 \cdot \log_2 b} - \sqrt{t(z(2d_1))}}.$$

Далее, к $R_1'(\delta)$ отнесем все такие слова $w \in R_1'$, что в результате подачи на вход слова $w \cdot mi(w)_\delta$ головка заходит в исходную зону слова $w \cdot mi(w)$ на глубину, не меньшую, чем $\frac{d_0}{3 \cdot \log_2 b}$. Зафиксируем некоторое δ_2 так, чтобы

$$|R_1'(\delta_2)| \geq \frac{|R_1'|}{d_0 + \delta_1 + \frac{d_0}{3 \cdot \log_2 b}} > 2^{d_0 + \frac{d_0}{3 \cdot \log_2 b} - 4\log_2 d_0}.$$

Наконец, R_2 определим как множество всех слов вида $w \cdot mi(w)_{\delta_2} \cdot v$, где $w \in R_1'(\delta_2)$, а v – произвольное слово (в алфавите $\{0,1\}$) длины $\frac{d_0}{3 \cdot \log_2 b}$. При этом:

1) все слова из R_2 – специальные слова с одинаковыми параметрами d_0; $3 \cdot \log_2 b$; δ_1, δ_2 (и, следовательно, с одинаковой длиной $d_2 = d_1 + \delta_2 + \frac{d_0}{3 \cdot \log_2 b}$);

2) $R_2 \geq 2^{d_0 + 2 \cdot \frac{d_0}{3 \cdot \log_2 b} - 4 \cdot \log_2 d_0}$;

3) пока подается любое слово из R_2, головка побывает не более чем на

$$d_0 + c_1^{(0)} \cdot \sqrt{t(2(2d_0))} + c_1^{(1)} \sqrt{t(2(2d_1))} \leq$$

$$\leq d_0 + 2 \cdot \max_{i \in \{1,2\}} c_1^{(i)} \cdot \max_{j \in \{1,2\}} \sqrt{t(2(2d_j))} \underset{def}{=} m_2$$

ячейках ленты.

Аналогично построим множества $R_3, R_4, \ldots, R_N$, где $N = 6 \cdot (\log_2 b)^2$, попутно определяя $\delta_3, \delta_4, \ldots, \delta_N$ и $d_3, d_4, \ldots, d_N$, а также пользуясь $c_1^{(3)}, c_1^{(4)}, \ldots, c_1^{(N)}$.

R_N обладает аналогичными свойствами:

1) все слова из R_N – специальные слова с одинаковыми параметрами d_0; $3 \log_2 b$; $\delta_1, \delta_2, \ldots, \delta_N$ (и, следовательно, с одинаковой длиной $d_N = d_{N-1} + \delta_N + \frac{d_0}{3 \cdot \log_2 b}$);

2) $|R_N| \geq 2^{d_0 + N \cdot \frac{d_0}{3 \cdot \log_2 b} - 2N \cdot \log_2 d_0}$;

3) пока подается любое слово из R_N, головка побывает не более чем на

$$m_N = d_0 + N \cdot \max_{i \in \{1,2,\ldots,N\}} c_1^{(i)} \cdot \max_{j \in \{1,2,\ldots,N\}} \sqrt{t(2(2d_j))}$$

ячейках ленты.

Пусть

$$\max_{i\in\{1,2,\dots,N\}} c_1^{(i)} = c_1^{(i_0)}$$

и

$$\max_{j\in\{1,2,\dots,N\}} t(z(2d_j)) = t(z(2d_{j_0})).$$

Так как конфигурации машины после подачи различных начал слов в алфавите $\{0,1\}$ по лемме 4 различны, то число $\tau(m_N)$ конфигураций длины m_N не меньше числа слов в R_N :

$$2^{2(d_0+N\cdot c_1^{(i_0)}\cdot\sqrt{t(z(2d_{j_0}))})\cdot\log_2 6} > \tau(m_N) \geq |R_N| \geq 2^{d_0+N\cdot\frac{d_0}{3\log_2 6}-2N\cdot\log_2 d_0}.$$

Отсюда

$$\sqrt{t(z(2d_{j_0}))} > \frac{d_0}{12\cdot c_1^{(i_0)}\cdot(\log_2 6)^3} - \frac{\log_2 d_0}{c_1^{(i_0)}\cdot\log_2 6}.$$

Так как d_0 можно было взять сколь угодно большим, так как $d_0 < d_1 < \dots < d_N < d_0\cdot 2^{1+N}$

и $z(2d_{j_0}) \leq (2d_{j_0}+1)\cdot(1+[\log_2 d_{j_0}])$,

то заключаем, что при некотором $c_4 > 0$ у нашей машины $t(\tau) > c_4\cdot\tau^2/\log\tau$ для бесконечного числа различных τ. Лемма доказана.

Из лемм 1, 8 и 9 непосредственно вытекает следующая теорема.

ТЕОРЕМА 10. (1) Для любого $\varepsilon > 0$ существует вероятностная одноленточная одноголовочная машина Тьюринга со входом и выходом, которая распознает язык D в реальное время так, что машина принимает любое слово из D с вероятностью 1 и отвергает любое слово из $\bar{D}$ с вероятностью $1-\varepsilon$. (2) Для любой детерминированной одноленточной одноголовочной машины Тьюринга со входом и выходом, распознающей язык D, существует такая константа $c > 0$, что для бесконечно многих различных n существует слово длины n,

на котором машина работает больше времени, чем $c \cdot n^2 / \log_2 n$. (3) Существует детерминированная одноленточная одноголовочная машина Тьюринга со входом и выходом, распознающая язык D за время $const \cdot n^2 / \log_2 n$.

ЛИТЕРАТУРА

1. Фрейвалд Р.В. Распознавание языков с высокой вероятностью на различных классах автоматов // Доклады АН СССР. - 1978. - Т.239, №1. - С.60-62.
2. Фрейвалд Р.В. Сложность распознавания симметрии на машинах Тьюринга с входом // Алгебра и Логика: Семинар. 1965 - Т.4 - Вып.1 - С.47-58.
3. Бухараев Р.Г. Основы теории вероятностных автоматов. М.- 1985. - 284с.
4. Барздинь Я.М. Сложность распознавания симметрии на машинах Тьюринга // Проблемы кибернетики. - М.-1965.- Вып.15.- С.245-248.
5. Трахтенброт Б.А. Тьюринговы вычисления с логарифмическим замедлением // Алгебра и Логика: Семинар.- 1964. - Т.3 - Вып.4. - С.33-48.
6. Рабин М.О. Вероятностные автоматы // Кибернетический сборник.- М.:Мир.-1964.-Вып.9-С.123-141.

АНАЛОГ ТЕОРЕМЫ ПАРИХА ДЛЯ ЯЗЫКОВ, ПРИНИМАЕМЫХ МНОГОЛЕНТОЧНЫМ ОДНОСТОРОННИМ КОНЕЧНЫМ НЕДЕТЕРМИНИРОВАННЫМ АВТОМАТОМ

Д.Г.Гейдманис

ВЦ ЛГУ им.П.Стучки

I. Введение

Мы опишем языки, принимаемые n-ленточным односторонним конечным недетерминированным автоматом (сокращенно: n-ленточный 1-КНА), путём подсчёта символов в словах языка.

Опишем n-ленточный 1-КНА. Подробное определение таких детерминированных автоматов можно найти в / 1 /. На каждой из двух лент автомата имеется по одной головке. Головки могут стоять на месте или двигаться слева направо. Входной информацией автомата является упорядоченная пара слов, написанных на лентах. После конца слова на каждой ленте написан специальный маркер # . Работа автомата задаётся программой, состоящей из команд. Команда по внутреннему состоянию в очередной момент работы и по буквам, обозреваемым головками (эту тройку будем называть левой частью команды), определяет следующее внутреннее состояние и движение той и другой головки (эту тройку будем называть правой частью команды). В левой части команды совокупность букв, обозреваемых головками, назовём условием команды. Во множестве внутренних состояний выделены начальное состояние и два заключительных состояния (принимающее и отвергающее). Пара слов принимается (отвергается), если в результате его обработки автомат приходит в принимающее (отвергающее) заключительное состояние.

n-ленточный 1-КНА отличается от детерминированного тем, что программа может содержать команды с одинаковыми

левыми частями и различными правыми частями.

Реализацией работы автомата на данной паре входных слов называется последовательность выполняемых команд. Недетерминированный автомат принимает данную пару слов, если существует реализация работы, оканчивающаяся в принимающем заключительном состоянии.

2. Определения

Пусть имеется недетерминированный автомат $\mathcal{A}$. Обозначим множество состояний автомата $\mathcal{A}$ через $\{q_1, q_2, \ldots, q_n\}$, а множество команд программы автомата - $\{I_1, I_2, \ldots, I_m\}$. Программу можно изобразить в виде ориентированного меченного графа. В графе G автомата $\mathcal{A}$ имеется ровно столько вершин, сколько состояний в автомате, и ровно столько ориентированных дуг, сколько команд в программе автомата. Каждая вершина помечена одним из состояний автомата, каждая дуга - командой программы. В графе автомата из вершины, помеченной состоянием q_i, в вершину, помеченную состоянием q_j, ведет дуга, помеченная командой I_k из программы автомата $\mathcal{A}$, тогда и только тогда, когда команда I_k переводит автомат из состояния q_i в состояние q_j.

Введем естественное понятие порождения цепочек команд.

Определение 2.1. Граф автомата порождает данную конечную цепочку команд, если в графе существует цепь, начинающаяся в вершине, помеченной начальным состоянием автомата, что 1) цепь проходит по стольким дугам, сколько в данной цепочке команд; 2) i -тая команда цепочки совпадает с меткой на i -той дуге цепи.

Пусть кроме графа G недетерминированного автомата $\mathcal{A}$ имеется еще один граф G_1, у которого количество вершин и дуг может отличаться от n и m, соответственно.

Определение 2.2. Будем называть граф G_1 сопоставимым с автоматом $\mathcal{A}$, если G_1 является графом

(вообще говоря другого) автомата и если метки на вершинах графа G_1 – это элементы множества состояний автомата $\mathcal{A}$, а метки на дугах графа G_1 – это команды из программы автомата $\mathcal{A}$.

Очевидно, что граф G автомата $\mathcal{A}$ сопоставим с автоматом $\mathcal{A}$ также, как сопоставим любой подграф графа G. Любой граф сопоставимый с автоматом $\mathcal{A}$ можно получить из конечного числа подграфов графа G путем совмещения некоторых вершин.

Определение 2.3. Будем говорить, что граф G_1, сопоставимый с недетерминированным автоматом $\mathcal{A}$, моделирует граф G автомата $\mathcal{A}$, если граф G_1 порождает только те (но не обязательно все) цепочки команд, что порождает граф G.

Определение 2.4. Граф G_1, сопоставляемый с недетерминированным автоматом $\mathcal{A}$, будем называть строго эквивалентным графу G автомата $\mathcal{A}$, если граф G_1 порождает те и только те цепочки команд, что порождает граф G.

Иначе говоря, граф G_1 строго эквивалентен графу G, если эти графы моделируют друг друга.

Перед тем как давать определения некоторых видов графов, скажем, что простой цепью будем считать цепь без повторяющихся вершин, а простым циклом – замкнутую простую цепь.

Если в графе G нужно особенно выделить вершину A среди других вершин, то будем писать (G, A) и говорить о графе с выделенной вершиной.

Пусть G_1 и G_2 – графы, сопоставимые с автоматом $\mathcal{A}$, а A_1 и A_2 – некоторые неузловые вершины графа G_1 и G_2 соответственно.

Определение 2.5. Назовём графы с выделенными вершинами (G_1, A_1) и (G_2, A_2) изоморфными, если существует взаимно однозначное соответствие $\mathcal{M}$ между вершинами графов G_1 и G_2 такое, что I) вершина A_1

соответствует вершине A_2 (это обозначим $A_1 \wr A_2$), 2) из вершины B_1 в вершину C_1 в графе G_1 идет дуга (B_1, C_1) тогда и только тогда, когда в графе G_2 также идет дуга (B_2, C_2) из вершины B_2 в вершину C_2 такая, что выполняется $B_1 \wr B_2$ и $C_1 \wr C_2$, и метки на дугах (B_1, C_1) и (B_2, C_2) одинаковые.

Иначе говоря, графы изоморфны, если совмещая выделенные вершины, можно потом наложить друг на друга также весь граф, и при этом метки на дугах совпадают. Если выделенные вершины обоих графов помечены одним и тем же начальным состоянием, то изоморфность этих графов очевидно влечет также строгую эквивалентность.

Определение 2.6. Кактусом назовем ориентированный связный граф, каждая дуга которого принадлежит ровно одному простому циклу.

Будем говорить, что кактус лежит на вершине графа, если при ее удалении из графа получается по крайней мере две связные компоненты, одна из которых - кактус без удалённой вершины. Вершину, на которой лежит кактус, назовём узловой вершиной.

Теперь кактус можно определить также рекурсивно: 1) простой цикл - это кактус; 2) граф, состоящий из кактуса, на любой вершине которого лежит кактус, - это тоже кактус.

Определение 2.7. Кактусным деревом данного кактуса назовем дерево, получаемое при взаимно однозначном отображении простых циклов кактуса в множество вершин дерева, в котором две вершины инцидентны общему ребру тогда и только тогда, когда прообразы вершин дерева - простые циклы кактуса имеют общую вершину

Определение 2.8. Деревом Хусими назовём ориентированный связный граф, состоящий из дерева с корнем, дуги которого ориентированы от корня к листьям (назовём его несущим деревом), на любой вершине которого лежит любое количество кактусов.

Пусть A и B - вершины дерева Хусими. Обозначим через

АВ простую цепь в дереве Хусими, соединяющую вершину А с вершиной В. Докажем, что если цепь АВ существует, то она единственная.

Во-первых, если А и В – вершины одного кактуса, то очевидно, цепь, соединяющая А и В, существует. Если, рассуждая от противного, от А к В идут две разные простые цепи, то получается, что в цепи, соединяющей В с А, существует дуга, принадлежащая двум разным простым циклам. Это противоречит определению кактуса. Во-вторых, если А и В – вершины несущего дерева, то очевидно, существует единственная цепь АВ. В-третьих, если А и В – вершины двух разных кактусов, то цепь можно разделить на три составные части: 1) соединяющую А с несущим деревом, 2) по несущему дереву до другого кактуса, 3) по другому кактусу от несущего дерева до вершины В. Но, как выяснилось выше, каждая из этих частей – простая единственная цепь

Понятие дерева Хусими для неориентированных графов ввели Уленбек и Риддел после появления статьи Хусими в связи с описанием структуры химических молекул. Кроме того исследованием занялись Харари и Нормен / 2 /, но они затрагивали главным образом вопросы, касающиеся подсчета топологически различных графов при фиксированном числе вершин и др. условиях.

В нашем случае дерево Хусими появляется как граф недетерминированного автомата. Мы ставим себе задачей показать, что все, что могут сделать недетерминированные автоматы вообще, то же самое могут сделать и недетерминированные автоматы, графы которых – деревья Хусими.

Ниже определим понятия линейного и полулинейного подмножества множества $\mathbb{N}^n$. Пусть $\mathbb{N}$ – множество неотрицательных целых чисел. Для каждого $n \geq 1$ положим $\mathbb{N}^n = \mathbb{N} \times \mathbb{N} \times \ldots \times \mathbb{N}$ (n раз). Введем некоторые операции. Пусть $v \in \mathbb{N}^n$, $L_1 \subset \mathbb{N}^n$, $L_2 \subset \mathbb{N}^n$. Тогда $v + L_1$ – это множество всех векторов, представимых в виде $v + u_1$, где $u_1 \in L_1$; $L_1 + L_2$ – множество векторов в виде $u_1 + u_2$, где $u_1 \in L_1$ и $u_2 \in L_2$;

v^* - множество векторов в виде $k \cdot v$, где $k \in \mathbb{N}$; $L_1 \cup L_2$ и $L_1 \cap L_2$ - это обычное объединение и пересечение множеств.

Определение 2.9. Множество $L \subset \mathbb{N}^n$ называется линейным, если $L = w + v_1^* + v_2^* + \ldots + v_m$, где $w \in \mathbb{N}^n$ и $v_i \in \mathbb{N}^n$ $(i = 1, 2, \ldots, m)$.

Определение 2.10. Подмножество множества $\mathbb{N}^n$ называется полулинейным, если оно является объединением конечного числа линейных множеств.

3. Описание посредством полулинейных множеств языков, принимаемых многоленточным I-КНА.

Результат Парика см. /3/ с помощью полулинейных множеств описывает контекстно - свободные языки или в терминах автоматов - языки, принимаемые одноленточным односторонним недетерминированным автоматом с магазинной памятью. Мы же дадим похожее описание многоленточных языков, но для конечных автоматов.

Рассмотрим n -ленточный I-КНА.

Без потери общности можем предположить, что слова на всех лентах написаны в одном и том же алфавите $\Sigma = \{a_1, a_2, \ldots, a_\ell\}$. Пусть $\bar{z} = (z_1, z_2, \ldots, z_n) \in (\Sigma^*)^n$. Будем обозначать через ψ^i отображение множества Σ^* на i -той ленте $(i = 1, 2, \ldots, n)$ во множество $\mathbb{N}^\ell$, определяемое следующим образом: $\psi^i(z) = (\#_{a_1}(z_i), \#_{a_2}(z_i), \ldots, \#_{a_\ell}(z_i))$, где $\#_{a_j}(z_i)$ - число вхождений символа a_j в цепочку z_i.

ЛЕММА 3.1. Если I-КНА, граф которого-дерево Хусими, принимает n -ленточный язык M $(M \subset (\Sigma^*)^n)$, где для каждой ленты алфавит Σ состоит из ℓ символов, то множества $\psi^1(M), \psi^2(M), \ldots, \psi^n(M)$ - полулинейные подмножества множества $\mathbb{N}^\ell$.

В доказательстве по цепочке команд, порождаемой графом автомата, восстановим n -ленточное слово $\bar{z}$,

которое принимается этой цепочкой команд. Исследуя все принимающие цепочки команд, порождаемые графом автомата, мы получим множество слов, т.е. язык M, принимаемый автоматом. Но о каждом слове z нас интересуют только значения функций $\psi^1(z)$, $\psi^2(z),\dots,\psi^n(z)$, а о языке M - множества $\psi^1(M),\psi^2(M),\dots,\psi^n(M)$.

ДОКАЗАТЕЛЬСТВО. Покажем, что $\psi^n(M)$ полулинейное подмножество множества $\mathbb{N}^l$, рассматривая одно фиксированное i $(i=1,2,\dots,n)$.

Пусть v - это цепь по графику автомата. Обозначим $\vec{v}$ (и назовём вектором кратности символов на i-той ленте цепи v) вектор $(n_1,n_2,\dots,n_l)\in\mathbb{N}^l$ такой, что выполняя цепочку команд, порождаемую цепью v, головка i-той ленты передвигается вправо $\sum_{j=1}^{l} n_j$ раз и ровно n_j раз при этом головка i-той ленты обозревает символ a_j $(j=1,2,\dots,l)$.

Аналогично $\vec{AB}$ обозначим вектор кратности символов на i-той ленте простой цепи AB из вершины A в вершину B, и $\vec{d}$ -вектор кратности символов i-той ленты простого цикла d.

Обозначим буквой 0 вершину несущего дерева Хусими, помеченную начальным состоянием автомата. Пусть B вершина дерева Хусими, помеченная принимающим заключительным состоянием. Исследуем произвольную цепь от 0 к вершине B. Она должна содержать простую цепь 0B, которая в узловых вершинах может пополняться замкнутой цепью по кактусу, лежащему на этой вершине.

Чтобы цепочка команд, порождаемая цепью от 0 к B была бы выполнимой на какой-то паре слов (такую цепь назовём непротиворечивой) необходимо и достаточно, чтобы команды этой цепочки не содержали противоречий с односторонностью автомата, т.е. I) если в цепочке впервые встретилась команда, содержащая условие: головка i-той ленты обозревает маркер $\#$ $(i=1,2,\dots,n)$, то предыдущая команда цепочки должна предусматривать перемещение головки i-той ленты вправо, а все последующие команды

должны обеспечивать положение головки на месте, 2) если данная команда цепочки предусматривает, что головка i-той ленты обозревает другой символ, а не такой как в предыдущей команде, то предыдущая команда должна предусматривать передвижение головки i-той ленты вправо

Если к непротиворечивости цепи прибавить, что последняя дуга цепи помечена командой, переводящей автомат в заключительное принимающее состояние, и все головки при этом обозревают маркер $\#$, то выполнив команды цепочки, порождаемые данной цепью, автомат пару слов принимает. Иначе не существует такой пары слов, на которой выполнив эту цепочку команд, автомат принял бы её.

Пусть вершина B помечена принимающим заключительным состоянием. Обозначим M(B) множество слов, принимаемых в вершине B цепочками команд, порождаемых цепями, начинающимися в вершине O. Пусть цепь OB по дереву Хусими содержит узловые вершины $A_1, A_2, \dots, A_k$. Обозначим G_i кактус (или букет кактусов), лежащих на узловой вершине A_i. Тогда множество $\Psi^{\mathcal{L}}(M(B))$ в точности равно сумме

$$\overrightarrow{OA_1} + \mathcal{L}(G_1) + \overrightarrow{A_1A_2} + \mathcal{L}(G_2) + \dots + \overrightarrow{A_{k-1}A_k} + \mathcal{L}(G_k) + \overrightarrow{A_kB} = \overrightarrow{OB} + \sum_{i=1}^{k} \mathcal{L}(G_i), \qquad (*)$$

где $\mathcal{L}(G_i)$ множество векторов кратности символов i-той ленты по всем непротиворечивым цепям в G_i, начинающимся и кончающимся в вершине A_i.

То, что $\mathcal{L}(G_i)$ - полулинейное подмножество множества N^e, можно показать, рассуждая по следующей схеме. Пусть на вершине A в цепи OB лежит букет G, состоящий из кактусов $H_1, H_2, \dots, H_t$ и пусть K_i - кактусное дерево кактуса H_i $(i = 1,2,\dots,t)$, прообраз корневой вершины которого - это простой цикл, содержащий вершину A. Пусть входящая в вершину A и выходящая из A дуга цепи OB помечена, соответственно, командой $I_{вх}$ и $I_{вых}$, а входящая в A и выходящая из A дуга кактуса H_i помечена соответственно командой $I_{i\,вх}$ и $I_{i\,вых}$ $(i = 1,2,\dots,t)$.

Определим команду I_1 предшествующую команде I_2 (обозначим $I_1 \leqslant I_2$), если условие команды I_2 можно

получить из условия команды I_1 путем замены по крайней мере одного символа из алфавита Σ на маркер #, или если эти условия совпадают.

Тогда $\mathcal{L}(G)$ будет объединением сумм

$$\mathcal{L}(K'_{i_1}, I_{1вх}, I_{i_2 вых}) + \mathcal{L}(K'_{i_2}, I_{i_2 вх}, I_{i_3 вых}) + \ldots$$
$$\ldots + \mathcal{L}(K'_{i_{t-1}}, I_{i_{t-2} вх}, I_{i_t вых}) + \mathcal{L}(K'_{i_t}, I_{i_{t-1} вх}, I_{вых}) \quad (\text{ж})$$

по всем комплектам $(K'_{i_1}, K'_{i_2}, \ldots, K'_{i_t})$ таким, что
1) $(i_1, i_2, \ldots, i_t)$ - перестановка кортежа $(1, 2, \ldots, t)$,
2) K'_i - это связное поддерево дерева K_i $(i = 1, 2, \ldots, t)$, содержащее корневую вершину (K'_i может быть пустым поддеревом $\varnothing$);
3) для любых команд $I_1 \leq I_2 \leq I_3 \leq I_4$ в случае пустых поддеревьев определено: $\mathcal{L}(\varnothing, I_1, I_3) + \mathcal{L}(K'_i, I_2, I_4) = \mathcal{L}(K'_i, I_1, I_4)$ и $\mathcal{L}(K'_i, I_1, I_3) + \mathcal{L}(\varnothing, I_2, I_4) = \mathcal{L}(K'_i, I_1, I_4)$, а также определено $\mathcal{L}(\varnothing, I_{вх}, I_{вых}) = \{\alpha^l\}$.

В сумме (ж) через $\mathcal{L}(K'_i, I_1, I_2)$ обозначено множество векторов кратности символов r-той ленты по всем таким замкнутым цепям С по кактусу H_i, идущим через вершину A_i, где цепь С по крайней мере один раз обходит дуги всех циклов - прообразов вершин поддерева K'_i и идет только по этим простым циклам, и если перед первой дугой цепи С присоединить дугу, помеченную командой I_1, и после последней - дугу с меткой I_2, то получается непротиворечивая цепь. Если такой цепи С не существует, то $\mathcal{L}(K'_i, I_1, I_2)$ определим как пустое множество и всю сумму (ж) - тоже как пустое множество. Множество $\mathcal{L}(K'_i, I_1, I_2)$ можно выразить следующим образом:
$\mathcal{L}(K'_i, I_1, I_2) = \Sigma_1 \alpha + \Sigma_2 \alpha^*$, где сумма Σ_1 идет по всем простым циклам α - прообразам вершин поддерева K'_i, а сумма Σ_2 идет только по тем простым циклам α -прообразам вершин поддерева K'_i, у которых все дуги цикла помечены командами с одним и тем же числом маркеров # в условии, и дуги которых образуют непротиворечивую цепь.

Очевидно, что $\mathcal{L}(K_i', \bar{I}_1, \bar{I}_2)$ – полулинейное подмножество множества $\mathbb{N}^\ell$, значит сумма (ж) тоже полулинейное подмножество $\mathbb{N}^\ell$.

Итак сумма вида (ж) – полулинейное подмножество множества $\mathbb{N}^\ell$. Объединив множества $\psi^z(M(B))$ по всем вершинам B в дереве Хусими, которые помечены принимающим заключительным состоянием, получим множество $\psi^z(M)$, которое как объединение полулинейных подмножеств – также полулинейное подмножество множества $\mathbb{N}^\ell$. □

Следующая лемма верна для любого вида недетерминированного автомата, поскольку использует только закономерности меченного графа.

ЛЕММА 3.2. Для каждого недетерминированного автомата $\mathcal{A}$ существует граф $H(\mathcal{A})$, сопоставимый с автоматом $\mathcal{A}$, такой что 1) граф $H(\mathcal{A})$ строго эквивалентен графу автомата $\mathcal{A}$; 2) граф $H(\mathcal{A})$ – это дерево Хусими.

Схема доказательства леммы. Обозначим P множество всех конечных цепей по графу автомата $\mathcal{A}$. Опишем алгоритм, который работает на любой цепи $\mathcal{D}$ из P и создает дерево Хусими $X(\mathcal{D})$, которое моделирует граф автомата $\mathcal{A}$ и в частности порождает цепочку $\mathcal{D}$ команд – меток цепи $\mathcal{D}$. Обозначим X множество всех деревьев Хусими, получаемых алгоритмом из цепей множества P. Покажем, что из-за особенностей алгоритма, X – это конечное множество деревьев Хусими, которые в совокупности порождают все конечные цепочки команд, которые порождает граф автомата $\mathcal{A}$. Совмещая вершину, помеченную начальным состоянием автомата $\mathcal{A}$, объединим все элементы X в одно дерево Хусими, которое строго эквивалентно графу автомата $\mathcal{A}$.

ДОКАЗАТЕЛЬСТВО. Приведем алгоритм, который по данной цепи дуг $\mathcal{D} = (\mathcal{D}_1, \mathcal{D}_2, \ldots, \mathcal{D}_m)$ по графу автомата $\mathcal{A}$ создает дерево Хусими $X(\mathcal{D})$. Без ограничения к общности предположим, что граф автомата $\mathcal{A}$ не имеет петель. Обозначим первую вершину цепи $\mathcal{D}$ символом Q_0, а последующие – $Q_1, \ldots, Q_m$ такие, что из вершины Q_{i-1}, к

вершине Q_i идет дуга $\mathfrak{D}_i$ $(i=1,2,\dots,m)$. Факт, что вершина Q_i помечена состоянием q_j автомата $\mathcal{A}$, обозначим равенством $\bar{Q}_i = q_j$, а что дуга $\mathfrak{D}_i$ помечена командой I_j из программы автомата $\mathcal{A}$ – равенством $\bar{\mathfrak{D}}_i = I_j$.

Для любого подмножества $S \subset \{Q_0, Q_1, \dots, Q_m\}$ обозначим $\bar{S} = \{\bar{Q}_i \mid Q_i \in S\}$ множество меток, которыми помечены вершины множества S. Пусть в S действует отношение эквивалентности $\equiv$, разбивающее множество S на классы эквивалентности. Любую вершину из S будем использовать как представитель класса эквивалентности, к которому она принадлежит. Обозначим $G_{\mathfrak{D}}(S)$ граф с вершинами из фактормножества $S/\equiv$, такой, что вершинам в множестве S из одного класса эквивалентности соответствует ровно одна вершина в графе $G_{\mathfrak{D}}(S)$, и с дугами из цепи $\mathfrak{D}$, что если S содержит вершины Q_{i-1} и Q_i, то в графе $G_{\mathfrak{D}}(S)$ из Q_{i-1} к Q_i идет дуга с меткой $\bar{\mathfrak{D}}_i$. Обозначим $\bar{G}_{\mathfrak{D}}(S)$ подграф графа автомата $\mathcal{A}$ с множеством вершин $\bar{S}$ и, если в $G_{\mathfrak{D}}(S)$ из Q_{i-1} к Q_i идет дуга с меткой $\bar{\mathfrak{D}}_i$, то в графе $\bar{G}_{\mathfrak{D}}(S)$ из вершины $\bar{Q}_{i-1}$ к $\bar{Q}_i$ идет дуга с меткой $\bar{\mathfrak{D}}_i$.

Работа алгоритма заключается в вычислении множества вершин S и отношения эквивалентности $\equiv$ в нем, и в конце работы определяем $G_{\mathfrak{D}}(S)$ как искомый граф $X(\mathfrak{D})$. Для описания алгоритма множество S разбито на два непересекающиеся подмножества: рабочие вершины S_p, которые во время работы алгоритма могут быть с помощью эквивалентности $\equiv$ определены совпадающими с очередной обрабатываемой вершиной, и нерабочие вершины $S_н$ $(S = S_p \cup S_н)$. Будем считать, что изменяя подмножества S_p и $S_н$ соответственно меняется также множество S и обратно.

Основное действие в алгоритме – при обработке очередной вершины Q_i цепи D – искать среди рабочих вершин такую, которая помечена тем же состоянием что очередная и при ее нахождении очередная и рабочая вершина объявляются совпа-

дающими и создается новый простой цикл.

Во всех действиях важны только метки вершин - состояния автомата , а названия вершин $Q_0, Q_1, Q_2, \ldots$ нужны только для техники описания алгоритма.

А Л Г О Р И Т М

$S_P := \{Q_0\}$; $S_H := \emptyset$;
do $i = 1$ **to** m **by** 1;
  if $\bar{Q}_i \notin \bar{S}_P$
  then $S_P := S_P \cup \{Q_i\}$;
  else /* $\bar{Q}_i \in \bar{S}_P$. Пусть имеется в точности k+1 вершина */
    /* $(k \geq 0)$ $Q_{j_0}, Q_{j_1}, \ldots, Q_{j_k} \in S_P (0 \leq j_0 < j_1 < \ldots < j_k < i)$, что */
    /* $\bar{Q}_{j_0} = \bar{Q}_{j_1} = \ldots = \bar{Q}_{j_k} = \bar{Q}_i$ */
    $S_P := S_P \cup \{Q_i\}$; $Q_i = Q_{j_k}$;
    /* $G_{Q_i}\{Q_j \in S_P \mid j_k \leq j < i\}$ - это новый простой цикл */
    /* $G_{Q_i}\{Q_j \in S \mid j_u \leq j \leq i\}$ - это новый кактус, лежащий */
    /* на вершине Q_i $(Q_{j_0} \equiv Q_{j_1} \equiv \ldots \equiv Q_{j_k} \equiv Q_i)$ */
    if ($k > 0$ и существует $u \in \{1, 2, \ldots, k\}$, что графы с выделенными вершинами $(G_{Q_i}\{Q_j \in S \mid j_{u-1} \leq j < j_u\}, Q_i\}$ и $(G_{Q_i}\{Q_j \in S \mid j_k \leq j < i\}, Q_i)$ изоморфны)
    then $S := S \setminus \{Q_j \in S \mid j_k < j < i\}$;
    else $S_H := S_H \cup \{Q_j \in S_P \mid j_k < j < i\}$; $S_P := S_P \setminus \{Q_j \in S_P \mid j_k < j < i\}$;
  fi **fi**
od

В алгоритме использован цикл **do - to - by - od**, оператор ветвления **if - then - else - fi**, присваива-

ния $:=$ и оператор $\equiv$, относящий вершины к одному классу эквивалентности. Комментарий заключен в скобках /* и */.

Обозначим $S(i)$, $S_\rho(i)$ и $S_\mu(i)$ значение множества, соответственно, S, S_ρ и S_μ после обработки вершины Q_i. Обозначим $|S|$ объем или число элементов множества S и $|\mathcal{A}|$ - количество состояний автомата $\mathcal{A}$.

Сформулируем некоторые утверждения, исполняющиеся при работе алгоритма.

УТВЕРЖДЕНИЕ I. Для каждого $0 \le t_1 < t_2 \le m$ из $Q_t \in S(t_2)$ следует, что $S(t_1) \subset S(t_2)$, т.е. что $G_g(S(t_1))$ подграф графа $G_g(S(t_2))$.

Вершины из S удаляются единственным оператором $S := S \setminus \{Q_j \in S \mid j_k < j < i\}$; Значит, рассуждая от противного, при каком-то i $(t_1 < i < t_2)$ удалив вершину Q_{t_0} $(t_0 < t_1)$ удалилась бы также вершина Q_{t_1}. Противоречие!

УТВЕРЖДЕНИЕ 2. Вершины одного класса эквивалентности в множестве S помечены одной и той же меткой.

Это очевидно следует из того, что единственный оператор $Q_{j_k} \equiv Q_i$; выполняется только тогда, когда $\bar{Q}_{j_k} = \bar{Q}_i$.

УТВЕРЖДЕНИЕ 3. Граф $G_g(S_\rho(i))$ - это простая цепь из вершины Q_0 к вершине Q_i.

При $i = 0$ утверждение выполняется тривиально, ведь $S_\rho(0) = \{Q_0\}$. Пусть утверждение 3 верно при $0 \le i \le t$. Сравним множество $\bar{S}_\rho(t+1)$ с множеством $\bar{S}_\rho(t)$. Если объем множества $\bar{S}_\rho(t+1)$ уменьшился, то при i=t+1 уменьшение происходит оператором $S_\rho := S_\rho \setminus \{Q_j \in S_\rho \mid j_k < j < i\}$; или оператором $S := S \setminus \{Q_j \in S \mid j_k < j < i\}$;, которые на множество S_ρ действуют идентично. Положив $t_1 = j_k$ и $t_2 = t$ из утверждения I следует, что $S_\rho(j_k) \subset S_\rho(t)$. Значит из $Q_{t+1} \equiv Q_{j_k}$ и из того, что вершины $Q_j \in S_\rho(j_k < j < t+1)$

из S_P удалены, следует равенство $S_P(j_k) = S_P(t+1)$. Верно индуктивное предположение для $i = j_k$. Если объем $|\bar{S}_P(t+1)|$ увеличился, то множество $S_P(t+1)$ увеличилось на вершину Q_{t+1}, где $\bar{Q}_{t+1} \notin \bar{S}_P(t)$. Значит в S_P появился новый класс эквивалентности и цепь $G_Q(S_P(t+1))$ – это простая цепь, удлиненная на одну дугу до вершины Q_{t+1}. Если допустить, что $|\bar{S}_P(t+1)| = |\bar{S}_P(t)|$, то должно быть $Q_{t+1} \equiv Q_t$ (значит $\bar{Q}_{t+1} = \bar{Q}_t$), что невозможно вследствие допущения, что в графе автомата $\mathcal{A}$ петлей нет.

Индукцией по i покажем, что $G_Q(S(i))$ – это дерево Хусими. При $i = 0$ множество $S(0) = \{Q_0\}$, а граф из одной вершины – это тривиальное дерево Хусими. Предположим, что при i $(0 \le i \le t)$ граф $G_Q(S(i))$ – дерево Хусими, состоящее из несущего дерева, – простой цепи $Q_0 Q_i$, на узловых вершинах которой лежат кактусы. Из утверждения 2 следует: простая цепь $Q_0 Q_i$ – это граф $G_Q(S_P(i))$. Положим $i = t+1$. Если $\bar{Q}_{t+1} \notin \bar{S}_P(t)$, то $Q_t \in S_P(t+1)$. Из утверждения I следует, что граф $G_Q(S(t+1))$ – это граф $G_Q(S(t))$, простая цепь $Q_0 Q_t$ которого удлинена на одну дугу из вершины Q_t к Q_{t+1}. Если $\bar{Q}_{t+1} \in \bar{S}_P(t)$, то находится вершина $Q_{j_k} \in S_P(t)$, что $\bar{Q}_{j_k} = \bar{Q}_{t+1}$ и определяется $Q_{j_k} \equiv Q_{t+1}$, и либо вершины $\{Q_j \in S \mid j_k < j < t+1\}$ удаляются из множества S, тогда $G_Q(S(j_k)) = G_Q(S(t+1))$, либо эти вершины в множестве $S(t+1)$ остаются и в том числе вершина $Q_t \in S(t+1)$. Тогда из утверждения I следует, что граф $G_Q(S(t+1))$ – это граф $G_Q(S(t))$, к которому присоединена дуга из вершины Q_t к вершине $Q_{t+1} \equiv Q_{j_k}$. Заметим, что вершина Q_{t+1} является представителем класса эквивалентности, включающего все такие вершины $Q_j \in S_P(t)$, что $\bar{Q}_j = \bar{Q}_{t+1}$, и в том числе $Q_{j_k} \equiv Q_{t+1}$. В соответствии с утверждением 3 множество вершин цепи $Q_0 Q_t$ – это $S_P(t)$. Из части $Q_{j_k} Q_t$ цепи $Q_0 Q_t$ и вышеупомянутой дуги из Q_t к Q_{t+1} образуется простой цикл, поэтому на

вершине Q_{t+1} лежит кактус. Вершины $\{Q_j \in S_p \mid j_k < j < t+1\}$ переносятся из S_p в $S_н$, поэтому множество $S_p(t+1)$ равно множеству вершин цепи $Q_0 Q_{t+1}$, при этом цепи $Q_0 Q_{t+1}$ и $Q_0 Q_{j_k}$ равны.

Покажем, что дерево Хусими $X(\mathfrak{D})$ порождает цепочку команд $\bar{\mathfrak{D}} = (\bar{\mathfrak{D}}_1, \bar{\mathfrak{D}}_2, \ldots, \bar{\mathfrak{D}}_m)$. Пусть во время работы алгоритма на цепи $\mathfrak{D}$ множество S уменьшается τ раз при значениях $i = i_j : 0 < i_1 < i_2 < \ldots < i_\tau \le m$ и пусть Δ_j $(j = 1,2,\ldots,\tau)$ множество, включающее при $i = i_j$ удаленные вершины и вершину Q_{i_j}. Вершины удалялись при условии, что на вершине Q_{i_j} уже лежит кактус, изоморфный кактусу $G_{\mathfrak{D}}(\Delta_j)$. Обозначим $E_0 = \mathfrak{D}$ и для каждого $j = 1,2,\ldots,\tau$ из цепи E_{j-1} создадим новую цепь E_j, замещая каждую дугу в E_{j-1}, идущую по кактусу $G_{\mathfrak{D}}(\Delta_j)$, на соответствующую дугу изоморфного кактуса. Значит $\bar{E}_j = \bar{E}_{j-1}$ и дуги цепи E_j до i_j-той дуги включительно идут по графу $G_{\mathfrak{D}}(S(i_j))$. Граф $G_{\mathfrak{D}}(S(i_\tau))$ подграф графа $G_{\mathfrak{D}}(S(m))$, который содержит также все дуги $\mathfrak{D}_j$ $(j > i_\tau)$ из цепи $\mathfrak{D}$, поэтому цепь E_τ $(\bar{E}_\tau = \ldots = \bar{E}_1 = \bar{E}_0 = \bar{\mathfrak{D}})$ идет по дугам графа $X(\mathfrak{D})$.

То, что граф $X(\mathfrak{D}) = G_{\mathfrak{D}}(S(m))$ моделирует граф автомата $\mathcal{A}$ очевидно следует из того, что граф $G_{\mathfrak{D}}(S(m))$ моделирует подграф $\bar{G}_{\mathfrak{D}}(S(m))$ графа автомата $\mathcal{A}$.

Определим множество X, которое состоит из графов $X(\mathfrak{D})$ получаемых алгоритмом, по всем конечным цепям $\mathfrak{D}$ по графу автомата $\mathcal{A}$, исходящим из вершины, помеченной начальным состоянием. Покажем, что X конечное множество, если одинаковыми считать изоморфные графы с выделенной вершиной Q_0. Для этого достаточно показать, что разных кактусов, получаемых алгоритмом, конечное число.

Уровнем простого цикла в кактусе с выделенной вершиной A, где A принадлежит ровно одному простому циклу, назовем количество дуг простой цепи по кактусному дереву

кактуса из корневой вершины, прообраз которой - простой цикл содержит A, к вершине, прообраз которой - данный простой цикл.

Обозначим H_k $(k \geq 0)$ множество кактусов с выделенной вершиной, сопоставимых с автоматом $\mathcal{A}$, что 1) кактусы из H_k состоят из простых циклов по графу автомата $\mathcal{A}$; 2) если для каждого кактуса G из H_k и узловой вершины B в нем рассмотрим множество подкактусов кактуса G, лежащих на вершине B, то среди них нет двух изоморфных кактусов с выделенной вершиной B; 3) простые циклы кактуса находятся не далее k-го уровня.

Таким образом, множество H_0 состоит из всевозможных пар (G, A), где G - простой цикл по графу автомата $\mathcal{A}$ и A-вершина этого цикла. Элемент множества H_k $(k \geq 1)$ мы получим из любого элемента множества H_0 - пары (G_0, A_0), к любой вершине A_1 в цикле G_0 $(A_1 \neq A_0)$ присоединением любого количества (включая ноль) разных кактусов G с выделенной вершиной A, где пара (G, A) - элемент множества H_{k-1} и вершины A и A_1 помечены одним и тем же состоянием автомата $\mathcal{A}$, и присоединение происходит путем совмещения вершины A_1 кактуса G_0 с вершиной A кактуса G.

Покажем, что для каждого дерева Хусими $X(\mathcal{D})$ из X любой кактус в $X(\mathcal{D})$ - это элемент множества H_k при $k = |\mathcal{A}|$. Во-первых, множество простых циклов в графах из H_k ограничено. Пусть $S_1 \subset S(m)$ множество вершин какого-то простого цикла в графе $X(\mathcal{D})$. Из утверждения 2 следует, что простому циклу $G_{\mathcal{D}}(S_1)$ в графе $X(\mathcal{D})$ соответствует простой цикл $\bar{G}_{\mathcal{D}}(S_1)$ в графе автомата $\mathcal{A}$, идущего по вершинам из множества $\bar{S}_1$. Во-вторых, заметим, что количество вершин цепи $Q_0 Q_i$, равное числу классов эквивалентности в множестве $S_p(i)$ (утверждение 3), совпадает с объемом множества $\bar{S}_p(i)$. Но $\bar{S}_p$ подмножество множества состояний автомата $\mathcal{A}$ и $|\bar{S}_p| \leq |\mathcal{A}|$. Из утверждения I следует, что для каждого i:

$Q_i \in S(m)$ количество вершин цепи $Q_0 Q_i$ не превосходит $|\mathcal{A}|$.

Множество разных деревьев Хусими, получаемых из цепей по графу автомата $\mathcal{A}$, конечно, что следует из ограниченности несущего дерева - цепи $Q_0 Q_m$ и из того, что на каждой вершине несущего дерева может находиться не более чем ограниченное число кактусов из множества $H_{|\mathcal{A}|}$. □

Используя результаты лемм мы можем доказать следующую теорему.

ТЕОРЕМА. Для каждого языка M, принимаемого n-ленточным I-КНА, множества $\psi^1(M), \psi^2(M), \ldots, \psi^n(M)$ являются полулинейными.

ДОКАЗАТЕЛЬСТВО. Пусть $\mathcal{A}$ -это I-КНА, который принимает язык M. Из леммы 3.2 следует, что автомату $\mathcal{A}$ соответствует строго эквивалентный автомат $\mathcal{A}'$, граф которого - это дерево Хусими. Из леммы 3.I следует, что у принимаемого автоматом $\mathcal{A}'$ языка M' множества $\psi^1(M), \psi^2(M), \ldots, \psi^n(M)$ - это полулинейные множества. □

Литература

I. Рабин М.О., Скотт Д. Конечные автоматы и задачи их разрешения // Кибернетический сборник.- М.- I962.- Вып.4.- С.58-9I.

2. Harary F., Norman R.Z. The dissimilarity characteristic of Husimi trees // Ann. of Math.(2)- Vol. 59 - 1953. - P.134-141.

3. Гинзбург С. Математическая теория контекстно-свободных языков.-М. - I970 - 326с.

4. Husimi K. Note on Mayers' theory of cluster integrals // - Journ.Chem.Phys. - Vol.18 - 1950 - P.682-684.

О РАСПОЗНАВАНИИ ω-ЯЗЫКОВ АВТОМАТАМИ С МАГАЗИННОЙ ПАМЯТЬЮ

Д.Я.Тайминя
ЛГУ им.П.Стучки

Известна следующая теорема (см. например [I]):

ТЕОРЕМА I. [I] Если язык над алфавитом из одной буквы распознаваем односторонним автоматом с магазинной памятью, то этот язык является регулярным, т.е. распознаваем конечным автоматом.

Цель настоящей статьи - доказать аналогичную теорему для ω-языков (т.е. языков, состоящих из слов бесконечной длины).

Конечным детерминированным автоматом A называется пятерка $(\Sigma, Q, \delta, q_0, F)$, где Σ - входной алфавит (непустое конечное множество), Q - конечное, непустое множество внутренних состояний, δ -функция переходов (отображение из $Q \times \Sigma$ в Q), $q_0 \in Q$ - начальное состояние, $F \subseteq Q$ - множество принимающих состояний. Говорят, что автомат A принимает слово, если, окончив работу над словом x, автомат попадает в принимающее состояние $q_a \in F$. Говорят, что автомат A отвергает слово x, если, окончив работу над словом x, автомат попадает в отвергающее состояние $q_r \notin F$. Говорят, что автомат A распознаёт язык L, если A принимает все слова из L и только эти слова.

Функционирование автомата A происходит строго детерминированно: в каждом такте следующее состояние полностью определяется воспринимаемым символом и текущим состоянием. Если допустить возможность выбора следующего состояния, т.е. функция переходов δ неоднозначна, то имеем дело с так называемым недетерминированным автоматом A'. Говорят, что недетерминированный автомат A' принимает слово x, если существует по крайней мере одна после-

довательность допустимых выборов состояний, при которой считывание слова завершается в принимающем состоянии $q_a \in F$. Автомат A' отвергает слово, если такой последовательности допустимых наборов состояний не существует.

Конечный детерминированный ω-автомат $\mathcal{A}$ так же как и автомат A имеет конечный входной алфавит Σ, множество внутренних состояний Q, функцию переходов δ, начальное состояние q_0. Множество всех подмножеств состояний ω-автомата $\mathcal{A}$ обозначим через $\mathfrak{F}$. В $\mathfrak{F}$ выделено подмножество $\mathfrak{G} \subseteq \mathfrak{F}$, называемое множеством принимающих подмножеств.

Пусть при работе ω-автомата $\mathcal{A}$ над некоторым ω-словом x некоторые внутренние состояния $\{q_{i_1}, q_{i_2}, \dots, q_{i_r}\}$ повторяются бесконечно много раз, а остальные - не более чем конечное число раз. Тогда множество $\mathcal{T} = \{q_{i_1}, q_{i_2}, \dots, q_{i_r}\}$ называется предельным множеством состояний ω-автомата $\mathcal{A}$ при работе над x. Говорят, что $\mathcal{A}$ принимает x, если предельное множество $\mathcal{T}$ принадлежит $\mathfrak{G}$. Говорят, что $\mathcal{A}$ распознаёт ω-язык $\mathcal{L}$, если $\mathcal{A}$ принимает все ω-слова из $\mathcal{L}$ и только эти ω-слова.

Детерминированным автоматом с магазинной памятью называется семерка $M = (Q, \Sigma, \Gamma, \delta, z_0, q_0, F)$, где Q - множество внутренних состояний (конечное, непустое), Σ - входной алфавит (конечное, непустое множество), Γ -множество символов магазинной памяти (конечное, непустое), δ - отображение множества $Q \times (\Sigma \cup \{\lambda\}) \times \Gamma$ в множество всех конечных подмножеств множества $Q \times \Gamma^*$, $q_0 \in Q$ - начальное состояние, z_0 - маркер магазинной памяти, $F \subseteq Q$ - множество финальных состояний. Множество финальных состояний разделяется на принимающие и отвергающие состояния. Автомат заканчивает работу, если он попадает в финальное состояние. Автомат с магазинной памятью принимает слово x, если финальное состояние является принимающим, и отвергает слово x, если фи-

нальное состояние является отвергающим состоянием.

Автомат, находясь в каком-либо состоянии, читает букву из входной ленты или λ и самый правый символ магазинной памяти является $z \in \Gamma$. Элементарным шагом работы автомата является следующее: автомат переходит в другое внутреннее состояние, стирает считанную букву или λ, стирает символ z на ленте магазинной памяти и записывает произвольную последовательность (быть может, пустую) на ленте магазинной памяти, начиная с ячейки, в которой был записан символ z.

Если функция переходов δ неоднозначна, то имеем дело с так называемым недетерминированным автоматом с магазинной памятью. Недетерминированный автомат с магазинной памятью принимает слово x, если существует такая последовательность допустимых выборов внутренних состояний, что автомат считав слово x попадает в принимающее состояние, и отвергает слово x в противном случае.

Детерминированный ω-автомат с магазинной памятью отличается от такого же автомата для обработки конечных слов лишь тем, что вместо множества F финальных состояний для автомата определено множество $\mathcal{F}$ принимающих подмножеств ($\mathcal{F} \subseteq \mathcal{P}$, где $\mathcal{P}$ - множество всех подмножеств внутренних состояний ω-автомата с магазинной памятью).

При попытке сформулировать аналог теоремы I, возникает следующее затруднение. Слов в алфавите из одной буквы бесконечно много. Они друг от друга отличаются длиной слова. В однобуквенном алфавите возможно только одно ω-слово. Поэтому прямой аналог упомянутой теоремы малосодержателен. Будет доказана следующая

ТЕОРЕМА 2. Пусть ω-язык L над алфавитом $\{0,1\}$ таков, что буква I в ω-словах из L появляется не более одного раза. Тогда из распознаваемости L недетерминированным автоматом с магазинной памятью следует распознаваемость L конечным автоматом.

Перед доказательством этой теоремы введем несколько обозначений и докажем две леммы. Рассмотрим произвольный язык L конечных слов над конечным алфавитом Σ. Зафиксируем букву $a \notin \Sigma$, обозначим $\Sigma' = \Sigma \cup \{a\}$. Образуем ω-язык L_ω над алфавитом Σ': $L_\omega = \{ x a^\omega \mid x \in L \}$.

ЛЕММА 3. Язык L (конечных слов) распознаваем детерминированным автоматом с магазинной памятью тогда и только тогда, когда язык L_ω (ω-слов) распознаваем детерминированным ω-автоматом с магазинной памятью.

ДОКАЗАТЕЛЬСТВО. Легко видеть, что из распознаваемости автоматом с магазинной памятью языка L следует и распознаваемость языка L_ω и автоматом с магазинной памятью.

Докажем, что из распознаваемости языка L_ω ω-автоматом с магазинной памятью следует распознаваемость языка L автоматом с магазинной памятью.

По ω-автомату с магазинной памятью M_ω, распознающему язык L_ω, построим автомат с магазинной памятью M, распознающий язык L.

Для моделирования нам понадобится дополнительная информация о каждой тройке $\mathfrak{D}(q_i, x(\ell), z_j)$, где q_i - внутреннее состояние автомата, $x(\ell)$ - обозреваемая буква на входной ленте, z_j - обозреваемая буква в магазинной памяти, дано следующее: а) если $x(\ell) \neq a$ и автомат M_ω при выходе из тройки $\mathfrak{D}$ будет работать бесконечно долго, больше не читая букв из входного слова, то M_ω будет принимать или отвергать входное слово; б) если $x(\ell) = a$, то возможны два случая: 1) автомат M_ω больше не будет заходить левее z_j на ленте магазинной памяти, в этом случае известно, будет ли M_ω входное слово принимать или отвергать, 2) если автомат M_ω при дальнейшей работе будет заходить левее z_j в магазинной памяти, то известно, в каком внутреннем состоянии q' это будет.

Эта дополнительная информация конечная, не зависящая от входного слова, и она задана равномерной таблицей для всех входных слов.

Рассмотрим автомат M, который будет моделировать работу автомата M_ω. Автомат M, читая начальный фрагмент x ω-слова xa^ω (где $x \in \Sigma^*$), выполняет те же самые действия, что и автомат M_ω, учитывая в каждый момент времени дополнительную информацию о соответствующей тройке $\mathfrak{A}$. Если из дополнительной информации следует, что автомат M_ω продолжит работу бесконечно долго, не читая новую букву из входной ленты (случай a) то моделирующий автомат на этом месте останавливается и выдает такой же результат, как и автомат M_ω, т.е. принимает или отвергает входное слово.

Если из дополнительной информации следует, что автомат M_ω продолжает считывать буквы с входной ленты, то продолжаем моделирование, пока M_ω не прочтет эту букву. Когда с входной ленты автомат M_ω первый раз считывает букву a, переходим к заключительному этапу моделирования. Если автомат M_ω продолжает считывание букв a на входной ленте, то наш автомат M работает без чтения новых символов на входной ленте и использует дополнительную информацию о соответствующей тройке $\mathfrak{A}$. Если из дополнительной информации следует, что, продолжая работу бесконечно долго, левее соответствующего символа z_j из тройки $\mathfrak{A}$, автомат M_ω на ленте магазинной памяти заходить не будет, то моделирующий автомат выдает такой же результат, как и автомат M_ω, т.е. принимает или отвергает входное слово. Если автомат M_ω будет возвращаться к символу z_j, то будем продолжать моделировать автомат M_ω после момента возврата. Если автомат M_ω будет заходить левее символа z_j в состоянии q', то моделирующий автомат использует дополнительную информацию о новой тройке (q', a, z).

Так моделирующий автомат либо прочтет всю запись на ленте магазинной памяти, либо достигнет какого-то символа,

левее которого продвигаться не будет. В любом случае после конечного числа шагов автомат M останавливается и выдает результат. Лемма доказана.

ЛЕММА 4. Язык L (конечных слов) принимается недетерминированным автоматом с магазинной памятью тогда и только тогда, когда язык L_ω (ω-слов) принимается недетерминированным ω-автоматом с магазинной памятью.

ДОКАЗАТЕЛЬСТВО отличается от доказательства леммы 3 лишь следующим. При определении дополнительной информации о тройках $\delta(q_i; x(e), x_j)$ вместо реакции ω-автомата указывается множество всех возможных реакций. При моделировании данного ω-автомата недетерминированный автомат M выбирает одну их этих реакций.

ДОКАЗАТЕЛЬСТВО ТЕОРЕМЫ 2 вытекает из леммы 4 и теоремы I.

ПРЕДЛОЖЕНИЕ 5. Существует такой ω-язык в алфавите $\{0,1\}$, каждое ω-слово которого содержит не более двух символов 1, и который распознаваем детерминированным ω-автоматом с магазинной памятью, но не распознаваем конечным ω-автоматом.

ДОКАЗАТЕЛЬСТВО. Легко видеть, что этим свойством обладает язык $\{0^n10^n10^\infty\}$.

ЛИТЕРАТУРА

I. Гинзбург С. Математическая теория контекстно-свободных языков. – М.– 1970.– С.123.

2. Гладкий А.В. Формальные грамматики и языки. – М.– 1973 – 368с.

3. Трахтенброт Б.А., Барздинь Я.М. Конечные автоматы: Поведение и синтез. – М.– 1970 – 400с.

О ЧИСЛЕ СОСТОЯНИЙ КОНЕЧНЫХ АВТОМАТОВ, РАСПОЗНАЮЩИХ РАВЕНСТВО ω-СЛОВ

Д.Я.Тайминя
ЛГУ им.П.Стучки

В настоящей статье речь пойдет об оценке необходимого и достаточного числа состояний конечных автоматов, распознающих равенство ω-слов, т.е. бесконечно длинных слов, состоящих из букв конечного алфавита.

Конечным автоматом A называется пятерка $(\Sigma, Q, \delta, q_0, F)$, где Σ - входной алфавит (непустое, конечное множество), Q - конечное, непустое множество внутренних состояний, δ - функция переходов (отображение $Q \times \Sigma$ в Q), q_0 - начальное состояние, $F \subseteq Q$ - множество финальных состояний. Множество всех входных слов, каждое из которых переводит q_0 в какое-либо финальное состояние $q \in F$, называется языком, представляемым автоматом A.

Конечный ω-автомат $\mathcal{A}$ так же как и автомат A имеет конечный входной алфавит Σ, множество внутренних состояний Q, функцию переходов δ, начальное состояние q_0.

Множество всех подмножеств состояний ω-автомата $\mathcal{A}$ обозначим через $\mathcal{F}$. В $\mathcal{F}$ выделено подмножество $\mathcal{G} \subseteq \mathcal{F}$, называемое множеством финальных подмножеств.

Пусть при работе ω-автомата $\mathcal{A}$ над некоторым ω-словом x некоторые внутренние состояния $\{q_{i_1}, q_{i_2}, \dots, q_{i_r}\}$ повторяются бесконечно много раз, а остальные - не более, чем конечное число раз. Тогда множество $\mathcal{T} = \{q_{i_1}, q_{i_2}, \dots, q_{i_r}\}$ называется предельным множеством состояний ω-автомата $\mathcal{A}$ при работе над x. Говорят, что $\mathcal{A}$ принимает x, если предельное множество $\mathcal{T}$ принадлежит $\mathcal{G}$. Говорят, что $\mathcal{A}$ распознает язык $\mathcal{L}$, если $\mathcal{A}$ принимает все ω-слова из $\mathcal{L}$ и только эти ω-слова.

Будем сравнивать число состояний конечных автоматов и конечных ω-автоматов при распознавании равенства слов. Может показаться, что число состояний в обоих случаях должно быть приблизительно одинаковым, но однако это не так - от-

личие может быть экспоненциальное. Точнее для каждого $k \in \mathbb{N}$ рассмотрим язык L_k: $L_k = \{y 2 y \mid y \in \{0,1\}^k\}$ и ω-язык M_k:

$M_k = \{y_1 2 y_2 2 y_3 2 \dots 2 y_n 2 \dots \mid (\forall^\infty n)(y_n = y_{n+1}) \& (\forall n)(y_n \in \{0,1\}^k)\}$.

ПРЕДЛОЖЕНИЕ I. Для распознавания языка L_k конечным автоматом необходимо не менее 2^k состояний.

ДОКАЗАТЕЛЬСТВО очевидно.

ТЕОРЕМА 2. Для распознавания языка M_k конечным ω-автоматом достаточно $2k+2$ состояний.

ДОКАЗАТЕЛЬСТВО. Доказательство теоремы опирается на то, что обычный конечный автомат имеет принимающие и отвергающие состояния, а для ω-автомата определяются множества принимающих и отвергающих состояний.

Наш ω-автомат имеет состояния $q_1^0, q_2^0, \dots, q_k^0, q_1^1, q_2^1, \dots, q_k^1, q_a, q_r$. Автомат $\mathcal{A}$ работает следующим образом: когда начинается новый фрагмент y_i, тогда после прочтения первой буквы фрагмента (обозначим ее $y_i(1)$) автомат $\mathcal{A}$ переходит в состояние $q_1^{y_i(1)}$, после прочтения второй буквы фрагмента автомат переходит в состояние $q_2^{y_i(2)}$ и т.д. После прочтения k-той буквы автомат переходит в состояние $q_k^{y_i(k)}$. Если в какой-либо из состояний q_i $(1 \leq i \leq k-1)$ автомат прочитает 2, то автомат переходит в состояние q_r. Если в состоянии q_k^0 или q_k^1 автомат прочитает 0 или 1, тогда автомат переходит в состояние q_r. Если в состоянии q_k^0 или q_k^1 автомат прочитает 2, то автомат переходит в состояние q_a. Если в состоянии q_a или q_r автомат прочитает 2, тогда автомат переходит в состояние q_r. Если в состоянии q_a автомат прочитает 0 или 1, то автомат дальше работает как в начальном состоянии q_0. Если в состоянии q_r автомат прочитает все равно что, то автомат остается в этом же состоянии. Принимающими будут все те множества состояний автомата $\mathcal{A}$, которые содержат q_a, не содержат q_r и из каждой пары состояний q_i^0, q_i^1 содержат ровно одно состояние.

Допустим, что ω-слово $y_1 2 y_2 2 \dots 2 y_n 2 \dots$ принадлежит языку M_k, т.е. все фрагменты y_i имеют длину

k и, начиная с некоторого места n, все y_i, $i > n$ равны.

Рассмотрим, как автомат $\mathcal{A}$ будет принимать это слово.

Начиная с некоторого фрагмента y_n, внутренние состояния $q_1^{y(1)}, q_2^{y(2)}, \ldots, q_k^{y(k)}, q_a$ повторяются бесконечно много раз (по одному разу в каждом фрагменте y_i), внутренние состояния $q_1^{\overline{y(1)}}, q_2^{\overline{y(2)}}, \ldots, q_k^{\overline{y(k)}}, q_r$ не появляются ни одного раза. Такое множество бесконечно много раз повторяющихся состояний по определению является принимающим.

Допустим, что ω - слово $y_1 2 y_2 2 \ldots 2 y_n 2 \ldots$ не принадлежит языку M_k. В этом случае есть несколько возможностей. Во-первых, может быть, что длина какого-то фрагмента y_i отлична от k. В этом случае ω-автомат при чтении ω-слова попадает в состояние q_r, из которого больше никогда не будет выходить. Но каждое множество состояний, содержащее q_r, является отвергающим.

Во-вторых, возможно, что все подслова y_i имеют длину k, но неверно утверждение, что начиная с некоторого n все y_n равны. Это означает, что существует такое j, $1 \leq j \leq k$, что в подсловах y_i, $i > n$ бесконечно много раз встречаются как $y_i(j) = 0$, так и $y_i(j) = 1$. Из этого следует, что бесконечно много раз появляются внутренние состояния q_j^0 и q_j^1, но все множества внутренних состояний, которые содержат эти состояния одновременно, являются отвергающими. Теорема доказана.

Сравнение предложения I и теоремы 2 показывает, что число состояний еще не достаточно характеризует сложность автомата. Еще надо учитывать сложность множества принимающих состояний. Полученный эффект как раз опирается на то, что само множество состояний может быть даже очень сложным по сравнению с числом состояний.

С другой стороны, сравнение полученных двух результатов показывает, что доказательство нижних оценок числа состояний ω-автомата отнюдь не тривиальная задача, которая бы автоматически решалась применением техники доказательства нижних оценок числа состояний обыкновенных конечных автоматов. Докажем нижнюю оценку числа состояний конечных

ω -автоматов, распознающих язык M_k.

ТЕОРЕМА 3. Каждый конечный ω -автомат, который распознает язык M_k, имеет не менее, чем k состояний.

ДОКАЗАТЕЛЬСТВО. Допустим от противного, что существует такое k и такой конечный ω -автомат, который распознает язык M_k с менее, чем k состояниями. Рассмотрим входное слово $y_1 2 y_2 2 \dots 2 y_n 2 \dots$, у которого все $y_i = 0^k$. Это слово принадлежит языку M_k. Автомат принимает слово. Это означает, что начиная с некоторого места, все те состояния, которые ω -автомат принимает конечное число раз, автомат уже больше не будет принимать. Это подслово обозначим через y_s. Так как подслов y_i бесконечно много, но число состояний конечно, то существует бесконечная последовательность этих подслов $\{y_{i_j}\}$ (без ущерба для общности можно считать, что все $i_j > s$), чтение которых наш фиксированный автомат всегда начинает в одном и том же входном состоянии. Так как число состояний ω -автомата меньше k, то ω -автомат, читая подслово y_{i_1} , будет принимать некоторое состояние повторно, т.е. существует внутреннее состояние q_m, в которое ω -автомат приходит после чтения буквы $y_{i_1}(\ell')$ и $y_{i_1}(\ell'')$ $(\ell'' > \ell')$.

Заметим, что если мы удлиним любое из y_{i_j} на $\ell'' - \ell'$ нулей, то наш ω -автомат не заметит подмены и закончит чтение подслова y_{i_j} в том же внутреннем состоянии.

Наряду с ω -словом $y_1 2 y_2 2 \dots 2 y_n 2 \dots$ рассмотрим и другое ω -слово, в котором $y_{i_1}, y_{i_3}, y_{i_5}, \dots$ удлинены на $\ell'' - \ell'$ нулей. Это ω -слово не принадлежит ω -языку M_k, так как не выполняется условие о длине фрагментов y_i. С другой стороны, наш ω -автомат попадает в то же самое множество бесконечно много раз повторяющихся состояний как и при чтении первого ω -слова. А это значит, что ω -автомат принимает и второе ω -слово, не принадлежащее языку M_k. Это приводит к противоречию. Теорема доказана.

ЛИТЕРАТУРА

I. Трахтенброт Б.А., Барздинь Я.М. Конечные автоматы: Поведение и синтез . - М.: Наука, 1970.- 400с.

ОЦЕНКА ДЛИНЫ СЛОВА ПРИ МОДЕЛИРОВАНИИ КОНЕЧНЫХ ДЕТЕРМИНИРОВАННЫХ АВТОМАТОВ

Я.А.Булс
ЛГУ им. П.Стучки

$\varphi: A \to B$ - отображение множества A в множестве B ; $(\varphi_1,\dots,\varphi_n): (A_1,\dots,A_n) \to (B_1,\dots,B_n) - \varphi_1: A_1 \to B_1, \dots, \varphi_n: A_n \to B_n$; в частности, если $A_1 = \dots = A_n = A$, то пишем $(\varphi_1,\dots,\varphi_n): A \to (B_1,\dots,B_n)$.

φ^{-1} - отображение обратное φ ; $\varphi|A'$ - ограничение φ на A' ; $|A|$ - мощность множества A .

Под понятием "автомат" везде понимается упорядоченная пятерка $\langle A,B,Q,\Gamma,\vartheta\rangle$, где A,B,Q - произвольные конечные непустые множества, а $(\Gamma,\vartheta): Q\times A \to (Q,B)$. Ради удобства под $\mathcal{A}$ повсюду понимается автомат $\langle A,B,Q,\Gamma,\vartheta\rangle$, под $\mathcal{B}$ - $\langle X,Y,Z,\Delta,\lambda\rangle$.

Логические связки - это $\vee$ (или), $\wedge$ (и) , $\Longrightarrow$ (влечёт), $\Longleftrightarrow$ (тогда и только тогда, когда), $\forall$ (для всех), $\exists$ (существует).

Символ $\rightleftharpoons$ заменяет словесный оборот "есть по определению"; так, если φ, ψ - отображения, то

$\varphi\psi(x) \rightleftharpoons \varphi(\psi(x))$, $\rho_\varphi \rightleftharpoons \{y \mid \exists x\ \varphi(x)=y\}$.

Слова "множество" и "алфавит" употребляются как синонимы. A^* - множество всех слов алфавита A, включая и пустое слово.

$\overline{k,n} \rightleftharpoons \{k,k+1,\dots,n\}$; но если i - индекс, то запись $i=\overline{k,n}$ означает, что i пробегает все значения множества $\overline{k,n}$.

$\ell(u)$ - число символов (букв) в слове u ;

$(u=u_1\dots u_n \wedge \forall i=\overline{1,n}\ \ell(u_i)=1) \Longrightarrow$
$(pr_k u \rightleftharpoons u_k \wedge \overline{pr}_k u \rightleftharpoons u_1\dots u_k)$.

$$u_i\in\overline{0,1} \Longrightarrow \bar{u}_i \rightleftharpoons \begin{cases} 0, & \text{если } u_i=1; \\ 1, & \text{если } u_i=0. \end{cases}$$

$(u=u_0\dots u_k \wedge \forall i=\overline{0,n}\ (u_i\in\overline{0,1})) \Longrightarrow$
$$d(u) \rightleftharpoons \sum_{i=0}^{n} u_i 2^{n-i}.$$

0^n - слово из n нулей.

$\lceil x \rceil$ - наибольшее целое число, меньшее или равное x.

Знак $\square$ в тексте отмечает начало доказательства, а знак $\boxtimes$ - его окончание.

О п р е д е л е н и е. Будем говорить, что $\mathcal{A}$ моделирует $\mathcal{L}$ посредством (φ, f, χ) - в символической записи $\mathcal{A} \geqslant \mathcal{L}(\varphi, f, \chi)$, если:

(i) $(\varphi, f, \chi): (X^*, B^*, Z) \to (A^*, Y, Q)$,

(ii) $\varphi(ux) = \varphi(u)\varphi(x)$,

(iii) $\chi(\Delta(z,u), x) = f\vartheta(\Gamma(x(z), \varphi(u)), \varphi(x))$

для всех $x \in X$, $z \in Z$, $u \in X^*$.

Условимся, что $\Delta(z,u) \rightleftharpoons z$, если u - пустое слово.

Отметим, что условие (ii) фактически позволяет определить φ на множестве X. В таком случае (ii) служит правилом доопределения φ на X^*.

Далее, если $\mathcal{A} \geqslant \mathcal{L}(\varphi, f, \chi)$ и $g: B^* \to Y$ такое, что $g|\vartheta(Q, \varphi(X)) = f|\vartheta(Q, \varphi(X))$, то $\mathcal{A} \geqslant \mathcal{L}(\varphi, g, \chi)$. Имея ввиду, что Q, X - конечные множества, получаем - $\vartheta(Q, \varphi(X))$ - конечное множество. Это показывает, что определение моделирования финитно. Иными словами - отображение f необходимо определить только на $\vartheta(Q, \varphi(X))$, т.е. на конечном множестве, для остальных слов алфавита B отображение доопределяется произвольно.

О п р е д е л е н и е. Условимся говорить, что $\mathcal{A}$ моделирует $\mathcal{L}$ (в символической записи $\mathcal{A} \geqslant \mathcal{L}$), если существуют такие φ, f, χ, что $\mathcal{A} \geqslant \mathcal{L}(\varphi, f, \chi)$. Если кроме того $\max\limits_{x \in X} l\varphi(x) = l^*$, то будем говорить, что $\mathcal{A}$ моделирует $\mathcal{L}$ со словами длины l^* (символически - $\mathcal{A} \geqslant^{l^*} \mathcal{L}$).

$\mathcal{A}$ называется (η, θ) -автоматом, если $|A| = \eta > 1$ и $|B| = \theta > 1$. Класс всех (η, θ) -автоматов обозначается через $\mathfrak{A}^{\eta}_{\theta}$.

В [1] рассмотрен следующий частный случай моделирования.

О п р е д е л е н и е. Говорят, что $\mathcal{A}$ t-моделирует $\mathcal{B}$ (символически - $\mathcal{A} \triangleright \mathcal{B}$), если существуют такие $(\varphi, \psi, \chi): (X, Y, Z) \to (A^*, B^*, Q)$, что:

(i) ψ - инъекция;

(ii) $\chi\Delta(z, x) = \Gamma(\chi(z), \varphi(x))$,

$\psi\lambda(z, x) = \vartheta(\chi(z), \varphi(x))$

для всех $(z, x) \in Z \times X$.

Если кроме того $\max\limits_{x \in X} l\varphi(x) = l^*$, то говорят, что $\mathcal{A}$ t-моделирует $\mathcal{B}$ со словами длиной l^* (символически - $\mathcal{A} \triangleright^{l^*} \mathcal{B}$).

Доказано [1], что существует такой (ζ, ξ)-автомат $\mathcal{B}$, для которого справедливо следующее предложение; если $\mathcal{A} \triangleright \mathcal{B}$ и $\mathcal{A}$ является (η, θ)-автоматом, то

$$l^* \geq \lceil \log_\eta \zeta \rceil + \lceil \log_\theta \xi \rceil - 1. \qquad (1)$$

Естественно встает вопрос:

Справедлива ли оценка (1) в общем случае?

Предложение 1 дает отрицательный ответ, зато предложение 2 показывает, что для малых ζ, ξ даже в общем случае оценка (1) верна.

Из предложения 6 видно, если изменить понятие моделирования, то удается улучшить оценку (1).

Введена мера сложности для класса $\mathfrak{M}^n_\theta$, даны первые оценки сложности. Предложения 3, 4, 5 дают точные значения сложностей для классов $\mathfrak{M}^5_3$, $\mathfrak{M}^2_3$, $\mathfrak{M}^3_3$ соответственно.

Интерпретация понятия моделирования

Введенное понятие моделирования согласовано со схемой: кодирующее устройство - моделирующий автомат - декодирующее устройство (рис. 1). Поясним подробнее схему $\mathcal{G}_{\gamma_1}$ - $\mathcal{A}$ - $\mathcal{G}_{\gamma_2}$. Если на вход кодирующего устройства $\mathcal{G}_{\gamma_1}$ подается буква $x \in X$, то $\mathcal{G}_{\gamma_1}$ перерабатывает x в слово $\varphi(x) \in A^*$. Далее, устройство (автомат) $\mathcal{A}$ слово $\varphi(x)$ переводит в $w \in B^*$, и, наконец, декодирующее устройство $\mathcal{G}_{\gamma_2}$ выдает символ

$\varphi(w) \in Y$. Кроме того, если автомат $\mathcal{B}$, находясь в состоянии z, букву x перерабатывает в y, то устройство $\mathcal{K}_{\varphi_1} - \mathcal{A} - \mathcal{K}_{\varphi_2}$ букву x тоже перерабатывает в y, если только $\mathcal{A}$ находится в состоянии $\chi(z)$. Отметим, что работа устройств $\mathcal{K}_{\varphi_1}$, $\mathcal{K}_{\varphi_2}$ не зависит от состояния $\chi(z)$, в котором установлено устройство $\mathcal{A}$.

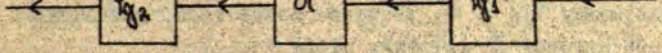


Рис. 1. $\mathcal{K}_{\varphi_1}$ - кодирующее устройство; $\mathcal{A}$ - моделирующий автомат; $\mathcal{K}_{\varphi_2}$ - декодирующее устройство.

Пусть $\forall i = \overline{1,n}$ $x_i \in X$ и автомат $\mathcal{B}$, находясь в состоянии z, перерабатывает слово $x_1 \dots x_n$ в слово $y_1 \dots y_n$. Тогда устройство $\mathcal{K}_{\varphi_1} - \mathcal{A} - \mathcal{K}_{\varphi_2}$ после установления $\mathcal{A}$ в состояние $\chi(z)$ преобразует также слово $x_1 \dots x_n$ в слово $y_1 \dots y_n$.

Прямая связь между $\mathcal{K}_{\varphi_1}$ и $\mathcal{K}_{\varphi_2}$ (рис. 1) указывает на необходимость информации для устройства $\mathcal{K}_{\varphi_2}$ о длине слов $\varphi(x_i)$. Это иллюстрирует следующий пример: $A = B = \{0\}$, $Q = \{q\}$, $\Gamma(q,0) = q$, $\delta(q,0) = 0$; $X = \{x_1, x_2\}$, $Y = \{y_1, y_2\}$, $Z = \{z\}$, $\Delta(z,x_1) = \Delta(z,x_2) = z$, $\lambda(z,x_1) = y_1$, $\lambda(z,x_2) = y_2$.

Определяя $\varphi(x_1) = \psi(y_1) = 0$, $\psi(x_2) = \psi(y_2) = 00$, $\chi(z) = q$ получаем, что $\mathcal{A} \rhd \mathcal{B}$.

Слово 00 расшифровывается двояко:
$00 = \psi(y_2)$ и $00 = \psi(y_1)\psi(y_1)$.

По своему замыслу устройство $\mathcal{K}_{\varphi_1} - \mathcal{A} - \mathcal{K}_{\varphi_2}$ должно t-моделировать $\mathcal{B}$, но после подачи буквы x_2 на вход $\mathcal{K}_{\varphi_1}$, если не имеется прямой связи между $\mathcal{K}_{\varphi_1}$ и $\mathcal{K}_{\varphi_2}$, устройство $\mathcal{K}_{\varphi_2}$ не в состоянии работать.

Итак, если дано слово $x_1 \dots x_n$, то всегда имеется возможность сконструировать слово $\varphi(x_1) \dots \varphi(x_n)$. Автомат $\mathcal{A}$ слово $\varphi(x_1) \dots \varphi(x_n)$ также всегда перерабатывает в слово $w_1 \dots w_k$, но далее это слово однозначно переработать в

$f(w_1)\dots f(w_n)$, если не известны длины слов $\varphi(x_1),\dots,\varphi(x_n)$, не всегда удается. Ситуацию "не спасает" и задание числа n. Так, в данном примере $000 = \psi(y_1)\psi(y_2) = \psi(y_2)\psi(y_1)$.

Отметим следующий результат [1]:

$$\forall \mathcal{B} \in \mathfrak{K}_\xi^S \; \exists \mathcal{A} \in \mathfrak{K}_\theta^n \; (\mathcal{A} \triangleright \mathcal{B} \wedge l^* = \lceil \log_\eta S \rceil + \lceil \log_\theta \xi \rceil - 1) \qquad (1').$$

Значит, существуют такие отображения φ, ψ, χ, что $\mathcal{A} \triangleright \mathcal{B}(\varphi, \psi, \chi)$. Доказывая в [1] этот факт, отображение φ строится так, чтобы $\forall x \in X \; l\varphi(x) = b^*$, т.е. в схеме $\mathfrak{N}_{\eta_1}$-$\mathcal{A}$-$\mathfrak{N}_{\eta_2}$ прямая связь между $\mathfrak{N}_{\eta_1}$ и $\mathfrak{N}_{\eta_2}$ излишня. Более того, доказывая (1'), построен автомат $\mathcal{B}$ [1] так, чтобы

$$\forall \mathcal{A} \triangleright \mathcal{B}(\varphi, \psi, \chi) \quad \forall x \in X \quad l\varphi(x) = \text{const}.$$

Мы видим, что существуют такие автоматы $\mathcal{B}$, для которых в схеме $\mathfrak{N}_{\eta_1}$-$\mathcal{A}$-$\mathfrak{N}_{\eta_2}$ (рис. 1) прямая связь между $\mathfrak{N}_{\eta_1}$ и $\mathfrak{N}_{\eta_2}$ при t-моделировании не нужна.

Далее, рассматривая доказательство предложения 1, увидим, что там тоже прямая связь между $\mathfrak{N}_{\eta_1}$ и $\mathfrak{N}_{\eta_2}$ не нужна.

Оценка длины слова .

П р е д л о ж е н и е 1.

$$\forall \mathcal{B} \in \mathfrak{K}_{2^{n}+1}^{k^{n}+1} \; \exists \mathcal{A} \in \mathfrak{K}_{2}^{k} \; (k>1 \wedge n>1 \Rightarrow \mathcal{A}^{k+n} \triangleright \mathcal{B}).$$

□ Ограничимся конструкцией отображений φ, ξ и определением подходящего автомата $\mathcal{A}$.

Пусть $X = \{x_i \mid i = \overline{0,2^k}\}$, $Y = \{y_i \mid \overline{0,2^n}\}$, $Z = \{z_i \mid i = \overline{0,m}\}$, $A = B = \overline{0,1}$.

$\varphi(x_i) \leftrightarrows \alpha_0^i \dots \alpha_k^i 0^{n-1}$, где $\alpha_s^i \in \overline{0,1}$ и определены, исходя из равенства $i = \sum_{s=0}^{k} \alpha_s^i 2^s$.

Вся дальнейшая трудность состоит в том, что

$\overline{\rho z}_k \varphi(x_0) = \overline{\rho z}_k \varphi(x_{2^n})$.

$$\xi(v) \leftrightarrows \begin{cases} c_i^{-1}(v), & \text{если } v \in \varrho_{c_i}; \\ y_0 & \text{в остальных случаях;} \end{cases}$$

где $c_1 : Y \setminus \{y_0\} \to B^*$, $c_2 : Y \setminus \{y_1\} \to B^*$, $c_3 : Y \to B^*$;

и $c_1(y_i) \leftrightarrows 0^k \rho_n^i \dots \rho_1^i$,

$$c_2(y_i) \leftrightarrows \begin{cases} 0^{k-1} 1 0^n, & \text{если } i = 0; \\ 0^{k-1} 1 \rho_n^i \dots \rho_1^i, & \text{если } i \neq 0; \end{cases}$$

$c_3(y_i) \Leftrightarrow 0^{k-2}1\gamma_n^i \dots \gamma_0^i$; $\beta_s^i, \gamma_s^i \in \overline{0,1}$ и определены из равенств

$$i-1 = \sum_{s=1}^{n} \beta_s^i 2^{s-1}, \quad i = \sum_{s=0}^{n} \gamma_s^i 2^s.$$

$$Q \Leftrightarrow \{q_{sj}^i \mid i = \overline{0,m} \wedge s+1 = \overline{1,k+n} \wedge j+1 = \overline{1,2^s}\}.$$

$$\Gamma(q_{sj}^i, t) \Leftrightarrow \begin{cases} q_{s+1\,2j+t}^i, & \text{если } s \in \overline{0,k+n-2}; \\ q_{00}^{\sigma}, & \text{если } \exists x (d_x\varphi(x) = 2j+t \wedge \Delta(z_i, x) = z_\sigma); \\ q_{00} & \text{в остальных случаях.} \end{cases}$$

Определение отображения ϑ.

(i) Если $\exists \sigma \in M_1(d\overline{pz}_{s+1}\varphi(x_\sigma) = 2j+t)$, где M_1 – множество нечетных чисел, то

$$\vartheta(q_{sj}^i, t) \Leftrightarrow pz_{s+1}\, c_3\lambda(z_i, x_\sigma).$$

(ii) Пусть $\exists \sigma \in M_0(d\overline{pz}_{s+1}\varphi(x_\sigma) = 2j+t)$, где M_0 – множество четных чисел.

(a) $((\lambda(z_i, x_0) \neq y_0 \wedge \lambda(z_i, x_{2^n}) \neq y_0) \vee$
$(\lambda(z_i, x_0) \neq y_1 \wedge \lambda(z_i, x_{2^n}) \neq y_1)) \Rightarrow$

$$\vartheta(q_{sj}^i, t) \Leftrightarrow \begin{cases} pz_{s+1}c_1\lambda(z_i, x_\sigma), & \text{если } \lambda(z_i, x_\sigma) \neq y_0; \\ pz_{s+1}c_2\lambda(z_i, x_\sigma), & \text{если } \lambda(z_i, x_\sigma) = y_0. \end{cases}$$

(б) $((\lambda(z_i, x_0) = y_0 \wedge \lambda(z_i, x_{2^n}) = y_1) \vee$
$(\lambda(z_i, x_0) = y_1 \wedge \lambda(z_i, x_{2^n}) = y_0)) \Rightarrow$
$\vartheta(q_{sj}^i, t) = pz_{s+1}\, c_3\, \lambda(z_i, x_\sigma)$.

(iii) $\vartheta(q_{sj}^i, t) \neq 0$ во всех остальных случаях. □

Предложение 2.

$\exists \mathcal{A} \in \mathcal{K}_5^3[(\mathcal{A} \overset{t}{\geq} \mathcal{A} \wedge \mathcal{A} \in \mathcal{K}_2^2) \Rightarrow t > 3]$.

□ Пусть $X \Leftrightarrow \{x_j \mid j = \overline{0,2}\}$; $Y \Leftrightarrow \{y_j \mid j = \overline{0,4}\}$;
$Z \Leftrightarrow \{z_j \mid j = \overline{0,124}\}$, где $z_j \Leftrightarrow y_{i_1}y_{i_2}y_{i_3} \Leftrightarrow$
$\sum_{s=1}^{3} i_s 5^{3-s} = j$; $\Delta(z_i, x_j) \Leftrightarrow z_i$; $\lambda(z_i, x_j) \quad pz_{j+1}z_i$.

Отсюда следует, что

$$\forall x\, \forall y\, \exists z\ \lambda(z, x) = y. \qquad (2)$$

Предположим, что $\mathcal{A} > \mathcal{J}r(\varphi, \xi, x) \wedge \exists x l\varphi(x) = 2$. Тогда, имея ввиду (2), $\forall i = \overline{0,4}\ \exists w^i[l(w^i) = 2 \wedge \xi(w^i) = y_i]$. Последнее условие противоречит факту, что ξ – отображение; ведь в алфавите $\overline{0,1}$ существуют только 4 разные слова длины 2.

Предположим, что $\forall x\ l\varphi(x)=3$; тогда
$\exists x\,\exists x'\ (x\neq x' \wedge pr_1\varphi(x)=pr_1\varphi(x'))$.

Имея ввиду (2), получаем:

$$\forall y\,\forall y'\,\exists w\,\exists w'\ (\xi(w)=y \wedge \xi(w')=y' \wedge pr_1 w = pr_1 w'), \qquad (3)$$

ведь автомат работает так, что одинаковые буквы, находясь в одном и том же состоянии, перерабатывает в один и тот же результат.

Пусть $W_0=\{w_0^i \mid i\in\overline{1,4} \wedge pr_1 w_0^i=0\}$ и $W_1=\{w_1^i \mid i\in\overline{1,4} \wedge pr_1 w_1^i=1\}$ – множества, содержащие только слова алфавита $\overline{0,1}$ длины 3. Пусть $\xi(w_j^i)=y_j^i\in Y$ и Y_j содержит все y_j^i. Тогда $Y=Y_0\cup Y_1$, так как $\forall x\ l\varphi(x)=3$, $\lambda(z,l\varphi(x))=l\lambda(z,\varphi(x))$ и выполняется (2). Но $|Y_j|\leqslant 4<|Y|$, потому найдутся такие $y^1\in Y\setminus Y_j$, что $y^0\neq y^1$.

Отсюда видно, что пара y^0, y^1 условию (3) не удовлетворяет. Значит неверно, что $\forall x\ l\varphi(x)=3$. ☐

Из предложения 1 заключаем, что $\forall \mathcal{A}\in \mathfrak{M}_3^5\ \exists \mathcal{A}\in \mathfrak{M}_2^2$ $\mathcal{A}\geqslant \mathcal{B}$. Но предложение 2 показывает, что аналогичный результат для автоматов класса $\mathfrak{M}_5^3$ неверен.

Отсюда, если $\forall \mathcal{B}\in \mathfrak{M}_3^5\ \exists \mathcal{A}\in \mathfrak{M}_2^2\ \mathcal{A}\geqslant \mathcal{B}$, то это не означает, что 5 и ξ можно менять местами. Более того, даже такой ослабленный результат не следует:

$$\forall \mathcal{B}\in \mathfrak{M}_5^3\ \exists \mathcal{A}[\mathcal{A}\geqslant \mathcal{B} \wedge (\mathcal{A}\in \mathfrak{M}_2^2 \vee \mathcal{A}\in \mathfrak{M}_2^\theta)].$$

В этом смысле параметры 5 и ξ не входят симметрично.

Сложность класс $\mathfrak{M}_3^5$.

Определение. Под суммарной сложностью автомата $\mathcal{B}$ понимаем число

$$n(\mathcal{B})=\min_{\mathcal{A}}\min_{\varphi}\sum_{x\in X} l\varphi(x),$$

где $\mathcal{A}\in \mathfrak{M}_{n_2}^{n_1} \wedge \mathcal{A}>\mathcal{B}(\varphi,\xi,x)$.

Число $n_5^\xi=\max_{\mathcal{B}\in \mathfrak{M}_5^\xi} n(\mathcal{B})$ будем называть суммарной сложностью класса $\mathfrak{M}_5^\xi$.

Корректность определения вытекает из нижеследующих рассуждений.

Так как

$$\forall 5\,\forall \xi\,\exists \mathcal{B}\in \mathfrak{M}_5^\xi\ \forall x\,\forall y\,\exists z\ \lambda(z,x)=y, \qquad (4)$$

то $n^5_\xi > 5\lceil \log_{d_2} \xi \rceil$.

Из условия (1') следует, что

$n^5_\xi \leq 5(\lceil \log_{d_2} 5 \rceil + \lceil \log_{d_2} \xi \rceil) - 5$.

Предложение 3. $n^5_3 = 14$.

□ Сперва построим такой автомат $\mathfrak{A} \in \mathfrak{M}^5_3$, для которого $n(\mathfrak{A}) \geq 14$, а потом в пункте (iv) покажем, что $14 = n^5_3$.

Пусть $X = \{x_j \mid j = \overline{0,4}\}$; $Y = \{y_j \mid j = \overline{0,2}\}$; $Z = \{z_j \mid j = \overline{0,24}\}$, где $x_j = y_{i_1} y_{i_2} \dots y_{i_5} \Leftrightarrow j = \sum_{s=1}^{5} i_s 3^{5-s}$; $\Delta(z_i, x_j) = z_l$; $\lambda(z_i, x_j) = pr_{j+1} z_l$.

Отсюда следует (2) и

$$\forall x\ \forall x'\ \forall y\ \forall y'\ \exists k\ [\lambda(k,x) = y \wedge \lambda(k,x') = y'] . \qquad (5)$$

Ради удобства далее слово v будем называть кодом буквы x (y) длины l^*, если $\varphi(x) = v$ ($\xi(v) = y$) и $l(v) = l^*$.

(i) Пусть $\exists x\ \exists x'\ [\varphi(x) = a_1 a_2 \wedge \varphi(x') = a_3 \bar{a}_3]$, где a_1, a_2 – буквы.

Имея ввиду (2) и то, что $pr_1 \varphi(x) = pr_1 \varphi(x')$, получаем: должны существовать коды v_0, v_1, v_2 длины 2 соответственно для букв y_0, y_1, y_2; притом $pr_2 v_0 = pr_2 v_1 = pr_2 v_2$. Таким образом, ситуация (i) невозможна, ведь эти коды должны быть различными.

Заключаем: существуют не более двух букв алфавита X коды которых длины 2.

(ii) Пусть $\exists x\ \exists x'\ [\varphi(x) = a_1 a_2 \wedge \varphi(x') = a_1 a_2 a_3]$, где a_1, a_2, a_3 – буквы.

Понятно, что (5) имеет место и для x, x'. Отсюда: имеются минимум два кода буквы y_0, один v^1_0 длины 2, другой w^1_0 длины 3. Но так как $\overline{pr}_2 \varphi(x) = \overline{pr}_2 \varphi(x')$, то $v^1_0 = \overline{pr}_2 w^1_0$.

Условие (5) имеет место как для пары (y_0, y_0), так и для пар (y_0, y_1) и (y_0, y_2). Пользуясь тем, что слов длины 3 с началом v^1_0 равно 2, получаем $\exists v^2_0 \neq v^1_0$ $(l(v^2_0) = 2 \wedge \xi(v^2_0) = y_0)$.

Тоже имеет место для пар (y_1, y_0), (y_1, y_1), (y_1, y_2). Следовательно, не найдется такого слова v_2 длины 2, чтобы $\xi(v_2) = y_2$; ведь слов длины 2 имеются ровно 4.

Итак, ситуация (ii) тоже невозможна.

(iii) Пусть $\exists x^0\ \exists x^1\ l\varphi(x^0) = l\varphi(x^1) = 2 \wedge$

$\exists x^2\ \exists x^3\ \exists x^4\quad l\varphi(x^2)=l\varphi(x^3)=l\varphi(x^4)=3.$

Пользуясь (i), заключаем, что $pr_1\varphi(x^0)\neq pr_1\varphi(x^1)$. Отсюда:

$\exists x\ \exists x'\ \exists x''\ [pr_1\varphi(x)=pr_1\varphi(x')=$
$=pr_1\varphi(x'')\wedge l\varphi(x)=2\wedge l\varphi(x')=l\varphi(x'')=3].$

Далее, из (ii) получаем, что $\varphi(x)=a_1a_2\Longrightarrow$
$[\varphi(x')=a_1\bar{a}_2a_3\wedge\varphi(x'')=a_1\bar{a}_2\bar{a}_3].$

Теперь, согласно конструкции автомата $\mathfrak{A}$:

$$\forall w\in Y^*\ \exists z\ [l(w)=3\Rightarrow(\lambda(z,x)=pr_1w\wedge\ \lambda(z,x')=pr_2w\wedge\lambda(z,x'')=pr_3w)]; \qquad (6)$$

потому для каждой тройки y,y',y'' найдутся такие коды v, v', v'', что

$$pr_1v=pr_1v'=pr_1v'', \qquad (7)$$
$$pr_2v'=pr_2v'', \qquad (8)$$
$$pr_3v'=pr_3v''. \qquad (9)$$

Имея ввиду, что y_0,y_1,y_2 различные буквы, заключаем: для этих букв нужны, как минимум, 3 различных кода длины 2. Значит, двое из этих кодов различается уже первыми буквами; ведь слов длины 2 только 4.

Понятно, что слова $y_0y_0y_1$, $y_1y_0y_1$, $y_2y_0y_1$ тоже удовлетворяют условию (6). Отсюда: для каждой буквы y_0,y_1 нужны как минимум два кода длины 3; иметь ввиду (7).

Далее, $y_0y_0y_2$, $y_1y_0y_2$, $y_2y_0y_2$ также удовлетворяют условию (6). Оказывается, что предыдущие коды буквы y_0 длины 3 здесь не годятся. Действительно, по (7)-(9) заключаем, что коды для y_1 и y_2 совпадают - противоречие.

Следовательно, для этих трех слов надо использовать ещё как минимум 4 новые коды длины 3.

Остается рассмотреть слово $y_0y_1y_2$. Аналогично предыдущему заключаем, что для буквы y_1 нужен новый код длины 3. Но ведь использовано не менее 8 различных слов длины 3, чтобы построить новые коды. Итак, для буквы y_1 уже не имеется возможности построить новый код. Следовательно, ситуация (iii) тоже невозможна, если все x^i равные.

Теперь приступим непосредственно к оценке числа $n(\mathfrak{A})$. Имеем $n(\mathfrak{A})=n_1+n_2+n_3+n_4+n_5$, где все $n_i\geqslant 2$, потому $n(\mathfrak{A})\geqslant 10$.

Но из (i) получается, что не более двух $n_i = 2$; значит $n(\mathfrak{A}) \geqslant 13$.

В свою очередь из (iii) получается, что остальные трое n_i не могут быть равны 3. Остались следующие возможности: 2+2+3+3+4=14, или 2+3+3+3+3=14, т.е., $n(\mathfrak{A}) \geqslant 14$.

(iv) Далее предположим, что $X = \{x_j \mid j = \overline{0,4}\}$; $Y = \{y_j \mid j = \overline{0,2}\}$; $Z = \{z_j \mid j = \overline{0,m}\}$; $(\Delta, \lambda): Z \times X \to (Z, Y)$ произвольные фиксированные отображения; $A = B = \overline{0,1}$.

Мы ограничимся конструкцией отображений φ, ψ и определением подходящего автомата $\mathcal{A}$, чтобы $n(\mathfrak{A}) = 14$.

Нижеследующее построение аналогично доказательству предложения 1.

$\varphi(x_0) \leftrightarrows 000$, $\varphi(x_1) \leftrightarrows 001$, $\varphi(x_2) \leftrightarrows 010$,
$\varphi(x_3) \leftrightarrows 11$, $\varphi(x_4) \leftrightarrows 100$.

$c_1 : \{y_1, y_2\} \to B^*$, $c_2 : \{y_0, y_2\} \to B^*$,
$c_3 : Y \to B^*$, $c_4 : Y \to B^*$, где
$c_1(y_1) \leftrightarrows 000$, $c_1(y_2) \leftrightarrows 001$;
$c_2(y_0) \leftrightarrows 010$, $c_2(y_2) \leftrightarrows 011$;
$c_3(y_0) \leftrightarrows 100$, $c_3(y_1) \leftrightarrows 101$, $c_3(y_2) \leftrightarrows 100$;
$c_4(y_0) \leftrightarrows 00$, $c_4(y_1) \leftrightarrows 01$, $c_4(y_2) \leftrightarrows 10$.

$$\psi(v) \leftrightarrows \begin{cases} c_i^{-1}(v), & \text{если } v \in \wp_{c_i}; \\ y_0 & \text{в остальных случаях.} \end{cases}$$

$Q \leftrightarrows \{q^i_{s_j} \mid i = \overline{0,m} \wedge s = \overline{0,2} \wedge j = \overline{0,s}\}$.
$\Gamma(q^i_{s_j}, t) \leftrightarrows q^i_{s,j+t}$, $\tau = \overline{0,1}$;
$\Gamma(q^i_{11}, 0) \leftrightarrows q^i_{22}$;
$\Gamma(q^i_{20}, t) \leftrightarrows q^\sigma_{00} \Leftrightarrow \Delta(z_i, x_t) = z_\sigma$;
$\Gamma(q^i_{2\tau}, t) \leftrightarrows q^\sigma_{00} \Leftrightarrow \Delta(z_i, x_{2\tau}) = z_\sigma$, $\tau = \overline{1,2}$;
$\Gamma(q^i_{11}, 1) \leftrightarrows q^\sigma_{00} \Leftrightarrow \Delta(z_i, x_3) = z_\sigma$.

$$\gamma(q^i_{00}, 0) \leftrightarrows \begin{cases} 0, & \text{если } \Lambda^i \neq Y_{01}; \\ 1, & \text{если } \Lambda^i = Y_{01}, \end{cases}$$

где $\Lambda^i \leftrightarrows \{\lambda(z_i, x_0), \lambda(z_i, x_1)\}$ и $Y_{\sigma\varepsilon} \leftrightarrows \{y_\sigma, y_\varepsilon\}$.

$$\gamma(q^i_{00}, 1) \leftrightarrows \begin{cases} 0, & \text{если } \lambda(z_i, x_3) \neq y_2; \\ 1, & \text{если } \lambda(z_i, x_3) = y_2. \end{cases}$$

Уже теперь можно заметить два основных принципа:

(а) если $\Lambda^i \neq Y_{01}$, то $\forall j = \overline{0,2}$ $\gamma(q^i_{00}, \varphi(x_j))$ определяется исходя из c_1 и c_2, в противном случае – из c_3;

(б) если $\lambda(x_i,x_3)\neq y_2$, то $\lambda(q^i_{00},\varphi(x_4))$ определяется исходя из c_1 и c_2, в противном случае – из c_3.

$$\lambda(q^i_{10},0)\rightleftharpoons\begin{cases}0, & \text{если } \Lambda^i\subseteq Y_{12}\vee\Lambda^i=Y_{01};\\ 1 & \text{в остальных случаях.}\end{cases}$$

$$\lambda(q^i_{10},1)\rightleftharpoons\begin{cases}pr_2c_1\lambda(x_i,x_2), & \text{если } \lambda(x_i,x_2)\in Y_{12}\wedge\lambda(q^i_{00},0)=0;\\ pr_2c_2\lambda(x_i,x_2), & \text{если } \lambda(x_i,x_2)=y_0\wedge\lambda(q^i_{00},0)=0;\\ pr_2c_3\lambda(x_i,x_2), & \text{если } \lambda(q^i_{10},0)=1.\end{cases}$$

$$\lambda(q^i_{11},0)\rightleftharpoons\begin{cases}0, & \text{если } \lambda(x_i,x_4)=y_1;\\ 1, & \text{если } \lambda(x_i,x_4)\neq y_1\wedge\lambda(x_i,x_3)\neq y_2;\\ pr_2c_3\lambda(x_i,x_4), & \text{если } \lambda(x_i,x_3)=y_2.\end{cases}$$

$$\lambda(q^i_{11},1)\rightleftharpoons pr_2c_4\lambda(x_i,x_3).$$

$$\lambda(q^i_{20},t)\rightleftharpoons\begin{cases}pr_3c_1\lambda(x_i,x_t), & \text{если } \Lambda^i\subseteq Y_{12};\\ pr_3c_2\lambda(x_i,x_t), & \text{если } \Lambda^i\subseteq Y_{02};\\ pr_3c_3\lambda(x_i,x_t), & \text{если } \Lambda^i=Y_{01}.\end{cases}$$

$$\lambda(q^i_{21},t)\rightleftharpoons\begin{cases}pr_3c_1\lambda(x_i,x_4), & \text{если } \lambda(x_i,x_2)\in Y_{12}\wedge\lambda(q^i_{00},0)=0;\\ pr_3c_2\lambda(x_i,x_2), & \text{если } \lambda(x_i,x_2)=y_0\wedge\lambda(q^i_{00},0)=0;\\ pr_3c_3\lambda(x_i,x_2), & \text{если } \lambda(q^i_{00},0)=1.\end{cases}$$

$$\lambda(q^i_{22},t)\rightleftharpoons\begin{cases}pr_3c_1\lambda(x_i,x_4), & \text{если } \lambda(x_i,x_4)=y_1\wedge\lambda(q^i_{00},1)=0;\\ pr_3c_2\lambda(x_i,x_4), & \text{если } \lambda(x_i,x_4)\in Y_{02}\wedge\lambda(q^i_{00},1)=0;\\ pr_3c_3\lambda(x_i,x_4), & \text{если } \lambda(q^i_{00},1)=1.\end{cases}$$ ■

Предложение 4. $\forall\xi>1\quad n^2_\xi=2\lceil\log_2\xi\rceil$.

□ Пусть $X=\{x_0,x_1\}$, $Y=\{y_1,\dots y_\xi\}$, $Z=\{z_0,\dots,z_m\}$, $A=B=\overline{0,1}$ и $n=\lceil\log_2\xi\rceil$. Пусть $\varphi(x_0)=0^n$, $\varphi(x_1)=10^{n-1}$.

Имея ввиду, что в алфавите $\overline{0,1}$ разных слов длины n равно $2^n\geq\xi$, получаем, что для каждого $y_i\in Y$ можно построить код $\psi(y_i)=\tilde{c}^i_1\dots\tilde{c}^i_n$ в алфавите $\overline{0,1}$, т.е., ψ – инъекция.

$$Q=\{q^i_t\mid i=\overline{0,m}\}\cup\{q^i_{jk}\mid i=\overline{0,m}\wedge j=\overline{1,n-1}\wedge k=\overline{0,1}\}.$$

$$\Gamma(q^i_t,t)=q^i_{1t};$$

$$\Gamma(q^i_{jk},t)=\begin{cases}q^i_{j+1,k}, & \text{если } j=\overline{1,n-2};\\ q^{i'}_t, & \text{если } j=n-1\wedge\Delta(z_i,x_k)=z_t.\end{cases}$$

$$\lambda(q^i_t,t)=pr_1\psi\lambda(z_i,x_t);$$

$$\lambda(q^i_{jk},t)=pr_{j+1}\psi\lambda(z_i,x_k).$$

Отсюда видно, что $\mathcal{A}\in\mathfrak{A}(\varphi,\psi,\chi)$, где $\forall i=\overline{0,m}\quad\chi(z_i)=q^i_t$. ■

П р е д л о ж е н и е 5. $n_5^3 = 10$.

□ Пользуясь (4), получаем, что $\forall x \in X \; l\varphi(x) \geq 3$, где $\mathcal{A} \in \mathfrak{K}_{2_1}^2$, $\mathcal{B} \in \mathfrak{K}_5^3$ и $\mathcal{A} \geq \mathcal{B}(\varphi, \xi, x)$. В свою очередь из предложения 2 следует, что $\exists x \; l\varphi(x) > 3$. Это означает, что $n_5^3 \geq 10$.

Следующая конструкция, схожая, по-существу, с доказательством предложения 1 показывает, что $n_5^3 = 10$.

Пусть $X = \{x_0, x_1, x_2\}$, $Y = \{y_0, \ldots, y_4\}$, $Z = \{z_0, \ldots, z_m\}$; $A = B = \overline{0,1}$; $\varphi(x_0) = 000$, $\varphi(x_1) = 100$; $\varphi(x_2) = 0100$;
$(c_1, c_2, c_3) : Y \to (B^*, B^*, B^*)$, где
$c_1(y_i) = u_0 u_1 u_2 \Leftrightarrow i = d(u_0 u_1 u_2)$, $u_j \in \overline{0,1}$;
$c_2(y_i) = 0c_1(y_i)$; $c_3(y_i) = 1c_1(y_i)$;

$$\xi(v) = \begin{cases} c_i^{-1}(v), & \text{если } v \in \varphi_{c_i}; \\ y_0 & \text{в остальных случаях.} \end{cases}$$

$Q^i = \{q_*^i\} \cup \{q_0^i, q_1^i\} \cup \{q_{00}^i, q_{11}^i, q_{12}^i, q_{22}^i\}$,
$Q = \bigcup_{i=0}^{m} Q^i$. $\Gamma(q_*^i, t) = q_{i_t}^i$,

$\Gamma(q_0^i, 0) = q_{00}^i$, $\Gamma(q_0^i, 1) = q_{12}^i$,
$\Gamma(q_1^i, t) = q_{11}^i$, $\Gamma(q_{12}^i, t) = q_{22}^i$,
$\Gamma(q_{jj}^i, t) = \Delta(z_i, x_j) = z_3$.
$\vartheta(q_*^i, t) = pz_1 c_1 \lambda(z_i, x_t)$,
$\vartheta(q_0^i, 0) = pz_2 c_1 \lambda(z_i, x_0)$,
$\vartheta(q_{00}^i, t) = pz_3 c_1 \lambda(z_i, x_0)$;
$\vartheta(q_0^i, 1) = pz_2 c_j \lambda(z_i, x_2) \Leftrightarrow \vartheta(q_*^i, 0) = j\text{-}2$;
$\vartheta(q_{\tau 2}^i, t) = pz_{\tau+2} c_j \lambda(z_i, x_2) \Leftrightarrow$
$\vartheta(q_*^i, 0) = j\text{-}2$, $\tau = \overline{1,2}$;
$\vartheta(q_{i_1}^i, t) = pz_2 c_1 \lambda(z_i, x_1)$;
$\vartheta(q_{i_2}^i, t) = pz_3 c_1 \lambda(z_i, x_1)$.

Положив $\forall i = \overline{0,m} \;\; x(z_i) = q_*^i$, можно убедится, что $\mathcal{A} \geq \mathcal{B}(\varphi, \xi, x)$. ■

Моделирование с задержкой

О п р е д е л е н и е. Будем говорить, что $\mathcal{A}$ моделирует $\mathcal{B}$ с задержкой посредством (φ, ξ, x) - символически: $\mathcal{A} >> \mathcal{B}(\varphi, \xi, x)$, если:

(i) $(\psi, \xi, \varkappa): (X^*, B^*, Z) \to (A^*, Y, Q)$;
(ii) $\varphi(ux) = \varphi(u)\varphi(x)$;
(iii) $\exists l^* \; \forall x \in X \quad l\varphi(x) = l^*$,
(iv) $(l(\alpha) = l\varphi(x)) \Rightarrow$
$\varkappa(\Delta(z,u),x) = \xi\partial(\Gamma(\varkappa(z),\varphi(ux)),\alpha)$,
для всех $\alpha \in A^*$, $u \in X^*$, $x \in X$, $z \in Z$.

Отметим, что тут также как и в ситуации моделирования отображения ξ, φ достаточно определять только на конечных множествах.

Введенное понятие согласовано со схемой: кодирующее устройство - моделирующий автомат - декодирующее устройство (рис. 2).

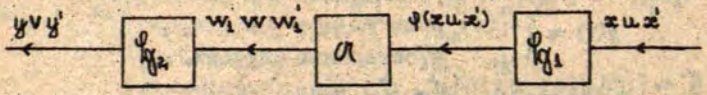


Рис. 2. $\mathfrak{M}_{\gamma_1}$ - моделирующее устройство; $\mathcal{A}$ - моделирующий автомат; $\mathfrak{M}_{\gamma_2}$ - декодирующее устройство.

Принцип работы схемы аналогичен описанной для рис. 1, только в данном случае не нужна верхняя связь, ведь $\forall x \in X \; l\varphi(x) = l^*$. Кроме того, после подачи слова xu ещё надо подать одну букву x'. Так, если автомат $\mathfrak{A}$ слово xu перерабатывает в слово vy', то моделирующее устройство $\mathfrak{M}_{\gamma_1} - \mathcal{A} - \mathfrak{M}_{\gamma_2}$ слово xux' перерабатывает в слово yvy'. Следовательно, если отбросить y, то моделирующее устройство выдает то же самое, что $\mathfrak{A}$.

Подобно случаю моделирования мы будем употреблять запись $\mathcal{A} \overset{l^*}{>>} \mathfrak{A}$.

Предложение 6.
$\forall \mathfrak{A} \in \mathfrak{K}_\xi^\zeta \; \exists \mathcal{A} \in \mathfrak{K}_\theta^\eta \quad \mathcal{A} \overset{l^*}{>>} \mathfrak{A}$, где

$l^* = \max(\lceil \log_\eta \zeta \rceil, \lceil \log_\theta \xi \rceil)$.

□ Пусть $X = \{x_i \mid i = \overline{0,\zeta-1}\}$, $Y = \{y_i \mid i = \overline{0,\xi-1}\}$, $A = \{a_i \mid i = \overline{0,\eta-1}\}$, $B = \{b_i \mid i = \overline{0,\theta-1}\}$ упорядоченные множества, т.е. для каждого

элемента зафиксирован индекс i. Пусть
$\forall a_i \in A\ \hat{a}_i \leftrightarrows i$, $\forall b_i \in B\ \hat{b}_i \leftrightarrows i$; $N_\eta(a_k \dots a_1 a_0) \leftrightarrows \hat{a}_k \eta^k + \dots + \hat{a}_1 \eta + \hat{a}_0$,
$a_j \in A$; $N_\theta(\beta_k \dots \beta_1 \beta_0) \leftrightarrows \hat{\beta}_k \theta^k + \dots + \hat{\beta}_1 \theta + \hat{\beta}_0$, $\beta_j \in B$.

(i) О п р е д е л е н и е отображений $\varphi, \mathfrak{f}$.

$$\varphi(x_i) \leftrightarrows a_1 \dots a_{\ell^*} \iff N_\eta(a_1 \dots a_{\ell^*}) = i.$$

$$\mathfrak{f}(\beta_1 \dots \beta_n) \leftrightarrows \begin{cases} y_i, & \text{если } n = \ell^* \wedge i < \xi \wedge N_\theta(\beta_1 \dots \beta_{\ell^*}) = i; \\ y_0 & \text{в остальных случаях.} \end{cases}$$

(ii) О п р е д е л е н и е множества Q.

$Q \leftrightarrows Z \times Y (Z \times Y)'$, где

$(Z \times Y)' \leftrightarrows \{(x,y)^k_j \mid (x,y) \in Z \times Y \wedge k = \overline{1,\ell^*} \wedge j = \overline{0,\eta^k - 1}\}$.

Кроме того, $(x,y)^{\ell^*}_j \leftrightarrows \begin{cases} (\Delta(x,x_j), \lambda(x,x_j)), & \text{если } j < S; \\ (x,y), & \text{если } j \geq S. \end{cases}$

(iii) О п р е д е л е н и е отображения Γ.

$$\Gamma((x,y)^1_j, a) \leftrightarrows \begin{cases} (x,y)^1_{N_\eta \overline{pr_1} \varphi(x_S)}, & \text{если } \exists s\ a = pr_1 \varphi(x_S), \\ (x,y) & \text{в противном случае.} \end{cases}$$

$$\Gamma((x,y)^{k+1}_j, a) \leftrightarrows \begin{cases} (x,y)^k_{N_\eta \overline{pr_k} \varphi(x_S)}, & \text{если } \exists s\ (a = pr_k \varphi(x_S) \wedge j = N_\eta \overline{pr_{k-1}} \varphi(x_S)); \\ (x,y) & \text{в противном случае;} \end{cases}$$

для всех $k = \overline{2,\ell^*}$.

(iv) О п р е д е л е н и е отображения ϑ.

Пусть $\psi : Y \to B^*$, где $\psi(y_S) \leftrightarrows \beta_1 \dots \beta_{\ell^*} \iff N_\theta(\beta_1 \dots \beta_{\ell^*}) = s$.
Тогда $\vartheta((x,y_S), a) \leftrightarrows pr_1 \psi(y_S)$; $\vartheta((x,y_S)^{k-1}_j, a) \leftrightarrows pr_k \psi(y_S)$, $k = \overline{2,\ell^*}$.

Тем самым автомат $\mathcal{A}$ определен полностью. Прямая проверка дает: $\mathcal{A} >> \mathfrak{B}(\varphi, \mathfrak{f}, \chi)$, где $\chi(x) \leftrightarrows (x, y_0)$. □

ЛИТЕРАТУРА

I. Будс Я.А. Некоторые понятия моделирования конечных детерминированных автоматов // Алгебра и дискретная математика: Теоретические основы математического обеспечения ЭВМ. - Рига: ЛГУ им. П.Стучки, 1986 - С.23-33.

О ПРОБЛЕМЕ ЭКВИВАЛЕНТНОСТИ МНОГОЛЕНТОЧНЫХ АВТОМАТОВ, ОДИН ИЗ КОТОРЫХ ПРОИЗВОЛЬНЫЙ

А.А.Калис

ВЦ ЛГУ им. П.Стучки

Как известно, одной из нерешенных проблем в теории автоматов является проблема эквивалентности многоленточных детерминированных конечных автоматов [1]. Разрешимость проблемы эквивалентности для автомата с двумя лентами была доказана в 1973 году Бердом [2]. Позже была доказана разрешимость проблемы включения, и следовательно, эквивалентности для автоматов, каждое состояние которых содержится не более чем в одном цикле [3]. Близкие к этому результаты были получены в [4]. Для класса автоматов, совершающих ограниченное число переходов на все ленты, кроме двух, была доказана [5] разрешимость проблемы эквивалентности.

В данной работе доказана алгоритмическая разрешимость проблемы эквивалентности для многоленточных автоматов с произвольным числом лент при условии, что один из автоматов совершает переходы на пары лент ограниченное число раз (другой автомат может быть произвольным). Число переходов на пары лент ограничено заранее известной константой. Видно, что данный класс автоматов значительно перекрывает второй из вышеупомянутых классов. Известно (и легко доказуемо), что проблема включения для многоленточных автоматов в общем случае неразрешима. Проблема включения неразрешима также для класса автоматов, рассмотренного в данной работе, так как он содержит в себе класс двухленточных детерминированных автоматов. Тем самым рассмотренный нами класс является самым широким классом из введенных в настоящее время с разрешимой проблемой эквивалентности и неразрешимой проблемой включения.

Приведем основные определения. *n*-**ленточным односторонним автоматом** над конечным алфавитом Σ называется система

$$\mathcal{N} = (Q, M, q_0, F, C_1, C_2, \dots, C_n),$$

где а) (Q, M, q_0, F) - обычный автомат, т.е. Q - конечное множество состояний, q_0 - начальное состояние, M - функция переходов, определенная на $Q \times (\Sigma \cup \{\varepsilon\})$ (ε - специальный символ, обозначающий конец слова на ленте), F - множество благоприятных состояний, б) $Q = \bigcup_{i=1}^{n} C_i$, $i \neq j \Rightarrow C_i \cap C_j = \phi$ (то есть , если $q \in C_i$,то в состоянии q двигается головка на i -той ленте).

Пусть Σ^* - множество всех слов в алфавите Σ ,включая пустое слово Λ . Будем говорить, что n -ка $x = (x_1, x_2, \dots, x_n) \in (\Sigma^*)^n$ **допускается** автоматом $\mathcal{N}$, если $\mathcal{N}$, прочитав все слова $x_1\varepsilon$, $x_2\varepsilon$, $\dots$, $x_n\varepsilon$, приходит в некоторое состояние $q \in F$. Слово $x_i\varepsilon \in \Sigma^*\varepsilon$ $(1 \le i \le n)$ будем называть **содержимым i-й ленты** автомата $\mathcal{N}$, или просто **i-й лентой**. Множество n-ок, допускаемых автоматом $\mathcal{N}$, будем называть **языком автомата** и обозначать через $L(\mathcal{N})$. Через Ω_n , $n \in \{1, 2, \dots\}$,обозначим класс всех n-ленточных автоматов. В дальнейшем нас будут интересовать автоматы из Ω_n , у которых число переходов на пары лент ограничено некоторой константой c . Множество таких автоматов обозначим через Ω_n^c .Определим класс Ω_n^c более точно. Пусть $\mathcal{N} \in \Omega_n$, $x = (x_1, x_2, \dots, x_n) \in (\Sigma^*\varepsilon)^n$ и на этой n-ке автомат проходит через состояния $q_1, q_2, \dots, q_m$. Пусть $1 = i_1 < i_2 < \dots < i_k = m$ такие, что

$$Q_1 = \{q_1, q_2, \dots, q_{i_2}\} \subseteq C_{m_1} \cup C_{t_1},\ Q_1 \cap C_{m_1} \neq \phi,\ Q_1 \cap C_{t_1} \neq \phi;$$

$$Q_2 = \{q_{i_2+1}, \dots, q_{i_3}\} \subseteq C_{m_2} \cup C_{t_2},\ q_{i_2+1} \notin C_{m_1} \cup C_{t_1},$$

$$Q_2 \cap C_{m_2} \neq \phi,\ Q_2 \cap C_{t_2} \neq \phi;$$

$$\vdots \qquad (I)$$

$Q_{i-1}\bar{t}\{q_{i_1},\dots,q_{i_k}\} \subseteq C_{m_{i-1}} \cup C_{e_{i-1}}$, $q_{i_2} \notin C_{m_{i-2}} \cup C_{e_{i-2}}$,

$Q_{i-1} \cap C_{m_{i-2}} \neq \emptyset$, $Q_{i-1} \cap C_{e_{i-2}} \neq \emptyset$;

Если для любых n-ок $x \in (\Sigma^* \varepsilon)^n$ число k ограничено сверху константой c, то будем говорить, что число переходов на пары лент ограничено константой c.

Основным результатом данной работы является

Теорема 1. Для любой пары автоматов $\mathcal{A} \in \Omega_n$ и $\mathcal{B} \in \Omega_n^c$ проблема эквивалентности алгоритмически разрешима.

Сначала докажем более простой факт:

Теорема 2. Проблема эквивалентности для автоматов из класса Ω_n^c алгоритмически разрешима.

Идея доказательства состоит в сведении проблемы эквивалентности к проблеме достижения состояния в некоторой машине M. Пусть $\mathcal{A}, \mathcal{B} \in \Omega_n^c$. Будем моделировать работу автоматов $\mathcal{A}$ и $\mathcal{B}$ на одних и тех же входных лентах. При этом, вообще говоря, автоматы с лент читают неравномерно, т.е. на входных лентах будут участки, прочитанные лишь одним автоматом:

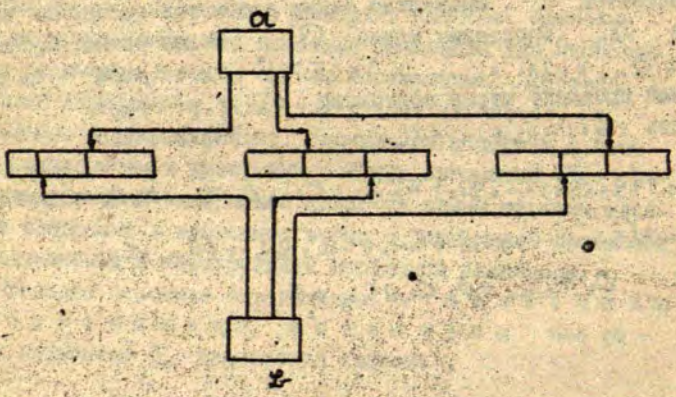


рис.I.

Иногда во время моделирования мы будем производить замену автоматов. В связи с этим будут появляться также двухэтажные участки входных лент:

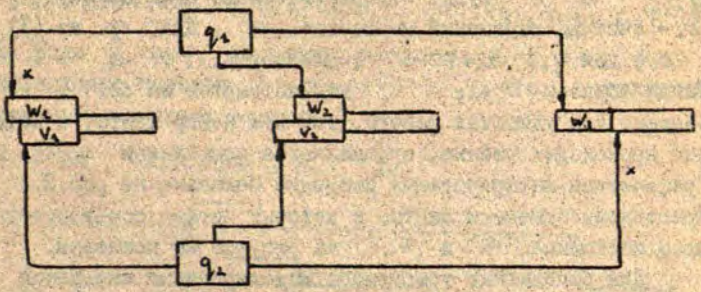


рис.2.

Верхний автомат на двухэтажном участке читает с верхнего участка ленты, а нижний автомат - с нижнего. Когда двухэтажный участок кончается, автоматы начинают читать продолжение ленты.

Ситуацией называется кортеж

$$s = (s^1, s^2) = (\langle q_1, a_1, l_1, t_1, w_1, w_2, \ldots, w_n \rangle, \langle q_2, a_2, l_2, t_2, v_1, v_2, \ldots, v_n \rangle),$$

где $q_1, q_2 \in Q_{a_1} \cup Q_{a_2}$

$a_i = \{r_i^1, r_i^2 \mid 1 \le r_i^1, r_i^2 \le n\}$ - пара номеров лент,

l_i - число переходов на пары лент,

t_i - номер ленты, с которой читает состояние q_i,

$w_i, v_j \in \Sigma^* \varepsilon$, $1 \le i \le 2$, $1 \le k, j \le n$.

Поясним введенные понятия и их содержательный смысл. Каждое состояние автоматов может читать лишь с одной ленты. a_1 (соответственно a_2) указывает, с какой парой лент работает автомат. Например, для q_{i_2} из (I) $a = \{m_1, l_2\}$. Иногда a_i будет содержать лишь один

элемент, именно, когда автомат сходит с одной пары лент на другую пару (в этом случае известна лишь одна лента, с которой автомат будет работать). Таким образом, для $q_{i,\dots}$ a будет содержать только один элемент. $i_{\cdot}$ - счетчик изменений множества $a_{\cdot}$. Для $q_{\cdot}$ из (I) $i_{\cdot}=1$ для $q_{\cdot,\dots}$, для $q_{\cdot,\dots}$ $i_{\cdot}=2$. Таким образом, $i_{\cdot}\le 2$ для автоматов из Ω_{Σ}^{c} . $t_{\cdot}$ введено для удобства работы, так как номер ленты, с которой происходит чтение, определяется состоянием $q_{\cdot}$. Графическая интерпретация ситуации показана на рис.2. Крестиками отмечены ленты, с которых читает соответствующее состояние. $a_{\cdot}$ и $i_{\cdot}$ на рисунке не показаны.

Для выполнения требуемого моделирования автоматов α и β введем понятие n - ленточной моделирующей машины $M=M(\alpha,\beta)$ для автоматов α, β . Под машиной $M(\alpha,\beta)$ будем понимать систему $M=\langle S,\Delta, s_0, A, T\rangle$, где S - множество ситуаций, Δ - функция переходов, A - множество акцептирующих ситуаций, T - внешняя память в виде стэка, в которой будут накапливаться ситуации, $s_0=(\langle q_0^{a}, \{i\},0,i,\Lambda\rangle, \langle q_0^{\beta}, \{j\},0,j,\Lambda\rangle)$ - начальная ситуация.

M начинает работать с начальной ситуации и при каждом шаге переходит в новую ситуацию , при этом, возможно, считывая с одной из входных лент. Функция переходов и множество акцептирующих ситуаций будут описаны ниже. Моделирование, т.е., работа машины M будет происходить поэтапно. Первый этап начинается в начальный момент . Если при переходе от ситуации s на следующую за ней ситуацию s' меняется хотя бы одно из множеств a_1, a_2 , то кончается один этап и начинается другой. В начале каждого этапа стэк T очищается.

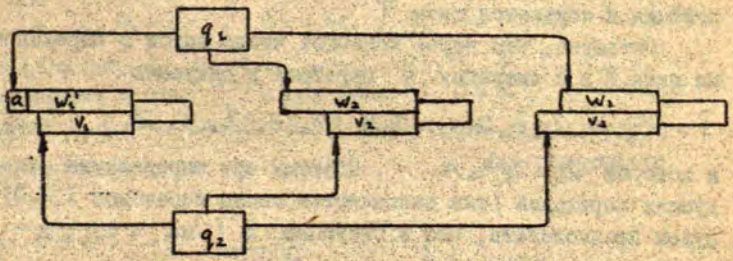
Опишем переходы машины M . Под $s \xrightarrow{a} s'$ будем понимать тот факт, что в M происходит переход от ситуации s к ситуации s' при чтении с соответствующей ленты символ $a\in\Sigma$. $s \xrightarrow{\lambda} s'$ - пустой переход, когда M не читает ни с одной ленты. В автоматах α и β переход от состояния q к состоянию q' обозначим через $q \xrightarrow{a} q'$.

Опишем теперь возможные переходы. Если возможно выполнение нескольких переходов, то выполняется переход с меньшим номером. Пусть M находится в ситуации

$$s=(s^1,s^2)=(\langle q_1,a_1,k_1,t_1,w_1,w_2,\dots,w_n\rangle,\langle q_2,a_2,k_2,t_2,v_1,v_2,\dots,v_n\rangle).$$

I. Если q_1 и q_2 из одного автомата, и $k_1<k_2$, то происходит переход $s\xrightarrow{\lambda}s'=(s^2,s^1)$, т.е., моделируемые автоматы меняются местами.

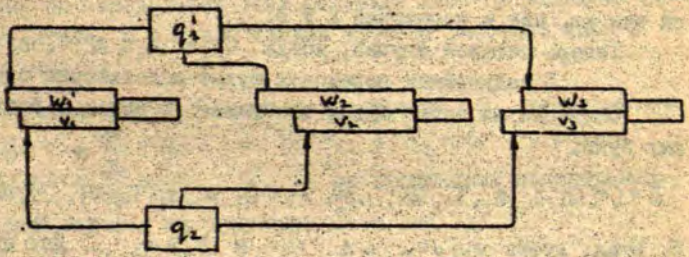
2. Если $w_{t_1}=a\,w'_{t_1}$, $q_1\xrightarrow{a}q'_1$:



то происходит переход

$$s\xrightarrow{\lambda}(\langle q'_1,a'_1,k'_1,t'_1,w_1,\dots,w'_{t_1},\dots,w_n\rangle,$$
$$\langle q_2,a_2,k_2,t_2,v_1,v_2,\dots,v_n\rangle)$$ или

графически



Если $t'_1\in a_1$, то $a'_1=a_1$, $k'_1=k_1$. Если $t'_1\notin a_1$ и $|a_1|=2$, то $a'_1=\{t'_1\}$, $k'_1=k_1+1$. Если $t'_1\notin a_1$ и $|a_1|=1$, то $a_1=a_1\cup\{t'_1\}$, $k'_1=k_1+1$. В последних двух случаях кончается очередной этап моде -

дирования и очищается стек T.

3. Если $v_{t_2} = b v_{c_2}$ и $q_2 \xrightarrow{b} q_2'$, то аналогично предыдущему случаю

$$s \xrightarrow{\lambda} (\langle q_1, a_1, b_1, t_1, w_1, \dots, w_n\rangle, \langle q_1', a_1', b_1', t_1', v_1, \dots, v_{c_2}', \dots, v_n\rangle).$$

Если $t_2' \in a_2$, то $a_2' = a_2$, $b_2' = b_2$. Если $t_2' \notin a_2$ и $|a_2| = 2$, то $a_2' = \{t_2'\}$, $b_2' = b_2 + 1$. Если $t_2' \notin a_2$ и $|a_2| = 1$, то $a_2' = a_2 \cup \{t_2'\}$, $b_2' = b_2 + 1$.

В последних двух случаях кончается очередной этап моделирования и очищается стек T.

Очевидно, что через конечное число шагов с переходами типа 2 и 3 ситуация s перейдет в ситуацию

$$\bar{s} = (\langle \bar{q}_1, \bar{a}_1, \bar{b}_1, \bar{t}_1, \bar{w}_1, \dots, \bar{w}_n\rangle, \langle \bar{q}_2, \bar{a}_2, \bar{b}_2, \bar{t}_2, \bar{v}_1, \dots, \bar{v}_n\rangle),$$

в которой $\bar{w}_{t_1} = \bar{v}_{t_2} = \Lambda$. Поэтому при определении следующих переходов (они выполняются после переходов I,2,3) будем предполагать, что в ситуации s $w_{t_1} = v_{t_2} = \Lambda$.

4. Если $t_1 = t_2$ (и $w_{t_1} = v_{t_2} = \Lambda$), то для каждого $a \in \Sigma$

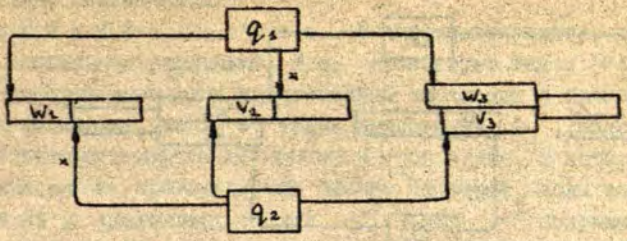
$$s \xrightarrow{a} (\langle q_1', a_1', b_1', t_1', w_1, \dots, w_n\rangle, \langle q_2', a_2', b_2', t_2', v_1, \dots, v_n\rangle),$$

т.е., оба автомата читают один и тот же символ и совершают переходы на новые состояния. a_i' и b_i' определяются так же, как в переходах 2 и 3.

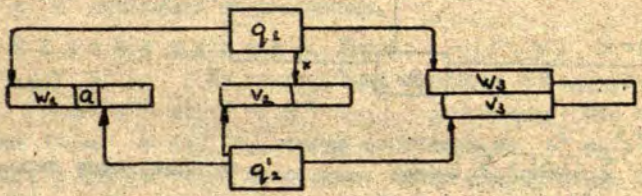
Теперь остался случай, когда $w_{t_1} = v_{t_2} = \Lambda$, а $t_1 \neq t_2$. В дальнейших переходах будет использован стек T. Ситуации на стека будем изображать с помощью больших букв:

$$S = (\langle Q_1, A_1, B_1, T_1, W_1, \dots, W_n\rangle, \langle Q_2, A_2, B_2, T_2, V_1, \dots, V_n\rangle).$$

5. Итак, пусть $w_{t_1} = v_{t_2} = \Lambda$ и $t_1 \neq t_2$. Для простоты будем считать, что $t_2 = 1$, $t_1 = 2$, т.е. $w_2 = v_1 = \Lambda$, и $a_2 = \{1,2\}$:



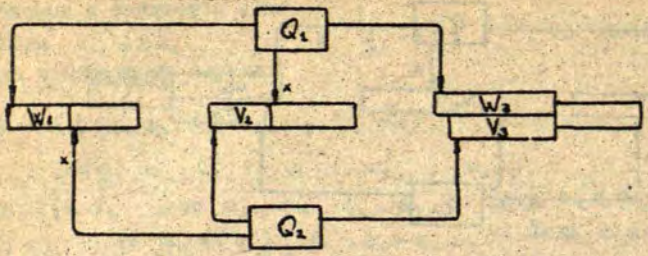
5.1. Если в стэке T нет ситуации S такой, что $Q_1=q_1, V_1=V_1$ $Q_2=q_2$, $V_2=V_2$ (и $W_3=W_3$, $V_3=V_3$, ... , $A_1=a_1$, $A_2=a_2$, $B_1=b_1$, $B_2=b_2$, $T_1=T_1$, $T_2=T_2$), то $s=(\langle q_1,a_1,b_1,t_1,w_1,\dots,w_n\rangle,\langle q_2,a_2,b_2,t_2,v_1,\dots,v_n\rangle)$ добавляется в стэк T и для каждого $a\in\Sigma$ происходит переход $s \xrightarrow{a} (\langle q_1,a_1,b_1,t_1,w_1,\dots,w_1,a,\dots,w_n\rangle,$ $\langle q_2',a_2',b_2',t_2',v_1,\dots,v_n\rangle)$ или графически



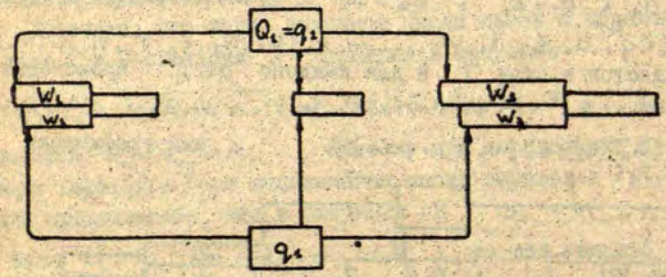
Условия в скобках $w_3=W_3,\dots$ можно опустить, так как по определению стэк содержит лишь ситуации из текущего этапа, а в этапе каждый автомат работает лишь с двумя лентами.

5.2. Пусть в стэке T имеется ситуация S такая, что $Q_1=q_1$, $Q_2=q_2$, $V_1=v_1$, $V_2=v_2$. Если $\sum_{i=1}^{n}|w_i| < \sum_{i=1}^{n}|v_i|$, то происходит недетерминированный переход на ситуацию S и s' :

$$s \xrightarrow{\lambda} S=(\langle Q_1,A_1,B_1,T_1,W_1,\dots,W_n\rangle,$$
$$\langle Q_2,A_2,B_2,T_2,V_1,\dots,V_n\rangle)$$



и $s \xrightarrow{\lambda} s' = (\langle q_1, a_1, b_1, t_1, w_1, \dots, w_n \rangle, \langle Q_1, A_1, B_1, T_1, W_1, \dots, W_n \rangle)$



Если $\sum_{i=1}^{n} |w_i| \geq \sum_{i=1}^{n} |w_i|$, то происходит переход как в 5.I без записи s в стек.
Тем самым переходы машины M описаны.

Будем говорить, что **состояния** $q_1, q_2 \in Q_\alpha \cup Q_\beta$ **различимы в ситуации** s, если существует $\alpha = (\alpha_1, \alpha_2, \dots \alpha_n) \in (\Sigma^*\varepsilon)^n$ такое, что q_1 на $(w_1\alpha_1, w_2\alpha_2, \dots, w_n\alpha_n)$ переходит в q_1', q_2 на $(v_1\alpha_1, v_2\alpha_2, \dots, v_n\alpha_n)$ переходит в q_2', причем один из q_1', q_2' принадлежит множеству благоприятных состояний, а другой не принадлежит.
Будем говорить, что при переходе $s \xrightarrow{a} s'$ $(a \in \Sigma \cup \{\lambda\})$ **сохраняется различимость состояний**, если состояния в s различимы тогда и только тогда, когда состояния в s' различимы.

Лемма I. В переходах I,2,3,4 и 5.I различимость со-

стояний сохраняется.

Доказательство. В действительности, если в s' состояния различимы, т.е. существует такая $\alpha \in (\Sigma^* \varepsilon)^n$, на котором лишь одно из состояний переходит в благоприятное состояние, то в s также существует α', полученная из α при добавлении символа a к той ленте, с которой в ситуации s производилось чтение. Наоборот, если состояния из S различимы, то для S' n-ку α' получим, удаляя в n-ке ситуации S символ a в ленте, из которой произошло чтение.

Теперь рассмотрим переход 5.2.

Лемма 2. Пусть в ситуации s имеются недетерминированные переходы $s \xrightarrow{\lambda} S$, $s \xrightarrow{\lambda} s'$.

1. Если в ситуации s состояния различимы, то состояния различимы или в ситуации S, или в ситуации s'.
2. Если автоматы $\mathcal{A}$ и $\mathcal{B}$ эквивалентны, то в ситуациях S и s' состояния не различимы.

Доказательство. Пусть $s = (s^1, s^2)$, $S = (S^1, S^2)$, $s' = (s^{1'}, s^{2'})$. Из конструкции перехода 5.2 следует, что $s^2 = S^2$, $s^{1'} = s^1$, $s^{2'} = S^1$. Очевидно, что при любой n-ке $\alpha \in (\Sigma^* \varepsilon)^n$ тогда состояния из s^2 и S^2 перейдут в одно и то же состояние. Также в одинаковые состояния перейдут состояния из $s^{1'}, s^1$ и $s^{2'}, S^1$.

Докажем утверждение 1. Пусть состояния из s различимы на α. Для определенности предположим, что s^1 переходит в благоприятное состояние, а s^2 – в неблагоприятное. Тогда $S^2 = s^2$ переходит в неблагоприятное состояние, а $s^{1'} = s^1$ – в благоприятное состояние. Так как $S^1 = s^{2'}$, то в S или в s' состояния будут различимы, а именно, если S^1 и $s^{2'}$ переходят в благоприятное состояние, то различимы будут состояния из S, так как $S^2 = s^{2'}$ при α переходят в неблагоприятное состояние. Если S^1 и $s^{2'}$ переходят в неблагоприятное состояние, то будут различимы состояния из s'.

Докажем утверждение 2. Если $\mathcal{A}$ и $\mathcal{B}$ эквивалентны, то в начальной ситуации s_0 состояния не различимы. Из

леммы I следует, что во всех ситуациях до первого перехода типа 5.2 состояния не различимы. Рассмотрим очередной переход типа 5.2. По предположению в предыдущих ситуациях состояния не различимы. Заметим, что ситуация S в моделировании была когда-то перед s, иначе она не могла попасть в стэк. Поэтому состояния в S не различимы. Пусть для определенности состояния из s при α переходят в благоприятные состояния. Так как $s^2 = S^2$, то при том же α состояние из S^1 переходит в благоприятное состояние. Из того, что $s^{1'} = s^1$ и $S^1 = s^{2'}$ следует, что состояния из $s^{1'}$ и $s^{2'}$ переходят в благоприятные состояния. Таким образом, состояния из S и s' при α переходят в благоприятные состояния, и значит, состояния из S и s' не различимы. Лемма доказана.

Л е м м а 3. Число различных ситуаций, достижимых в машине $M(\alpha, \beta)$ при вариации входных лент, ограничено некоторой константой $C_{\alpha,\beta}$, причем $C_{\alpha,\beta}$ можно эффективно найти по α и β.

Д о к а з а т е л ь с т в о. Рассмотрим следующие два утверждения:

1) число этапов моделирования эффективно ограничено некоторой константой,

2) в каждом этапе моделирования сумма длин слов $\sum_{i=1}^{n} (|w_i| + |v_i|)$ увеличивается ограниченное число раз.

Из этих утверждений следует, что $\sum_{i=1}^{n} (|w_i| + |v_i|)$ для каждой достижимой ситуации ограничено, и поэтому число достижимых ситуаций также ограничено. Таким образом для доказательства леммы достаточно доказать утверждения I и 2. ∎

I. Пусть M моделирует автоматы $\alpha, \beta \in \Omega_n^k$ на некоторой n-ке $x \in (\Sigma^* \varepsilon)^n$, проходя ситуации $s_0, s_1, \ldots, s_t, \ldots$. Новый этап моделирования начинается, если при переходе на новую ситуацию меняется хотя бы один из a_1, a_2. При этом меняется и b_1, b_2. Сопоставим с каждым шагом моделирования t число $l(t) = b_1 + b_2$.

Рассмотрим произвольный переход $s_t \xrightarrow{a} s_{t+1}$. Если новый этап не начинается, то $l(t+1) = l(t)$. Пусть при переходе $s_t \xrightarrow{a} s_{t+1}$ начинается новый этап. Если это переход 2,3,4 или 5.I, то $l(t+1) = l(t)+1$ или $l(t+1) = l(t)+2$. Если это переход типа 5.2 $s \xrightarrow{\lambda} s'$, и состояния в s_t из разных автоматов, то может случиться, что $l(t+1) \le l(t)$, но тогда и в последующих ситуациях состояния будут принадлежать одному автомату. Если в переходе типа 5.2 состояния из одного автомата, то $l(t+1) > l(t)$ из-за применения перехода I. Так как $\alpha, \beta \in \Omega_c^c$, то как было замечено при определении ситуации, $l_1, l_2 \le 2c$. Исходя из свойств $l(t)$ заметим, что $\max_t l(t) \le 3c$. Таким образом число этапов не превосходит $3c+1$.

2. Пусть $s_i, s_{i+1}, \dots, s_t, \dots s_k$ – последовательность ситуаций в некотором этапе. Определяем $l(t) = \sum_{i=1}^{t} |w_i(t)|$, $r(t) = \sum_{i=1}^{t} |v_i(t)|$.

Очевидно, что при переходах I,2,4,3 максимальное значение $l(t)$ и $r(t)$ не меняется или уменьшается. В одном этапе число ситуаций с разными q_1, q_2, v_1, v_2 будут не более чем $|Q| \cdot |Q| \cdot r(i)$. Пусть переход $s_t \to s_{t+1}$ типа 5.I. Тогда $l(t+1) = l(t)+1$, $r(t+1) = r(t)$. Если $s_t \to s_{t+1}$ типа 5.2 и является первым из обеих недетерминированных переходов (т.е. $s \xrightarrow{\lambda} S$), то $l(t+1) < l(t)$, $r(t+1) = r(t)$, если вторым (т.е. $s \xrightarrow{\lambda} s'$), то $l(t+1) = l(t)$, $r(t+1) = r(t)+1$. Таким образом,

$$l(k) \le |Q|^2 \cdot r(i) + l(i),$$
$$r(k) \le \max\{r(i), |Q|^2 \cdot r(i) + l(i)\}$$

Пусть $H_i = \max\{r(i), l(i)\}$. Тогда $l(k) \le H_i \cdot (|Q|^2+1)+1$, $r(k) \le H_i \cdot (|Q|^2+1)+1$. Число переходов ограничено $3c+1$. Поэтому в любой последовательности ситуаций $s_0, s_1, \dots, s_m$, полученной при моделировании автоматов α и β

$$l(m) \le \sum_{i=0}^{3c+1} (|Q|^2+1)^i = \frac{(|Q|^2+1)^{3c+1}}{|Q|^2}.$$

$$r(m) \le (|Q|^2+1)^{3c+1} / |Q|^2 = H$$

Из этого следует, что число возможных ситуаций ограничено константой H. Ясно, что и в стеке T число ситуаций не может превосходить число H. Лемма доказана.

В машине $M(\mathcal{A}, \mathcal{B})$ надо определить еще акцептуальные ситуации. Такими будут все ситуации $s = (\langle q_1, a_1, b_1, t_1, t' \rangle, \langle q_2, a_2, b_2, t_2, t'' \rangle)$ такие, что один из q_1, q_2 принадлежит множеству благоприятных состояний автоматов $\mathcal{A}$ и $\mathcal{B}$, а другой не принадлежит.

Под **языком** $L(M)$ **машины** M будем понимать множество n-ок из $(\Sigma^* c)^n$, на которых M переходит в акцептирующее состояние. Из вышеизложенного следует

Лемма 4. Автоматы $\mathcal{A}, \mathcal{B} \in \Omega_n^c$ эквивалентны тогда и только тогда, когда язык машины $L(M)$ пустой.

Теперь заметим, что M реализуема в недетерминированном n-ленточном автомате. В качестве состояний такого автомата берем пары (s, T), где s - произвольная ситуация, у которой $\sum_{i=1}^{n}(|w_i| + \sum_{i=1}^{n}|u_i|) \in H$, T - стек, т.е. кортеж из всевозможных ситуаций с ограниченной длиной. Переходы автомата будут такими, как в машине M, благоприятными состояниями будут состояния, в которых есть акцептирующие ситуации. Для конечного недетерминированного автомата проблема пустоты языка разрешима. Тем самым теорема 2 доказана.

Рассмотрим теперь доказательство теоремы I. Оно почти полностью совпадает с доказательством теоремы 2, изменяется лишь немного работа машины M. Пусть $\mathcal{A} \in \Omega_n^c$, $\mathcal{B} \in \Omega_n$. В качестве начальной ситуации берем ситуацию

$$s_0 = (\langle q_0^{\mathcal{A}}, \{t_1\}, 0, t_1, \Lambda \rangle, \langle q_0^{\mathcal{B}}, \phi, 0, t_2, \Lambda \rangle).$$

На первом этапе работы машины M в каждой ситуации s определяем $a_2 = \phi$, $b_2 = 0$. Кроме того, на первом этапе в переходах 5.I и 5.2, M не будет требовать выполнения равенств $w_3 = W_3, w_4 = W_4, \ldots$. Этап кончится, когда будет выполнен переход 5.2 $s \Rightarrow s'$, т.е. переход на ситуацию с состояниями из автомата $\mathcal{A} \in \Omega_n^c$. Последующие этапы работы машины M выполняются как

в доказательстве теоремы 2. Заметим, что леммы I, 2, 4 имеют место и в случае теоремы I. Остается показать, что имеет место также и лемма 3. Ясно, что число этапов моделирования и в данном случае ограничено константой. На самом деле, число этапов, следующих за первым этапом, ограничено константой $|Q|+1$, так как в них состояния из $Q \in \Omega^c$. Заметим, что на первом этапе $v_1 = v_2 = \ldots = v_n = \Lambda$ и что в стэке могут находиться не более $|Q| \cdot |Q|$ различных состояний. Так как сумма длин слов в ситуации $\sum_{i=1}^{n} (|w_i| + |v_i|)$ увеличивается лишь тогда, когда ситуация заносится в стэк, то $\sum_{i=1}^{n} (|w_i| + |v_i|) \leq |Q|^2$. Очевидно, что и в последующих этапах $\sum_{i=1}^{n} (|w_i| + |v_i|)$ увеличивается ограниченное число раз. Из этого следует, что лемма 3 истинна и в случае теоремы I. Также как в теореме 2 машина M реализуема в недетерминированном автомате. Тем самым теорема I доказана.

ЛИТЕРАТУРА

1. Rabin M.O., Scott D. Finite automata and their decision problems // IBM J.Res.Dev. - 1959 -Vol.3, N 2 - P.114-125.
2. Bird M. The equivalence problem for deterministic two-tape automata // J.Comp.Syst. Sci. - 1973 - Vol.7, N 3.- P.218-236.
3. Kinber E.B. The inclusion problem for some classes of deterministic multitape automata // Theoretical Computer Science - 1983. - N 26 - P.1-24.
4. Lewis H.R. A new decidable problem, with applications // Proc. 18. Ann.Symp. on Foundations of Comp.Sci. - 1979 - P.62-73.
5. Кинбер Е.Б. Об одном классе многоленточных автоматов с разрешимой проблемой эквивалентности // Программирование. - 1983. - № 3 - С.16-24.

ЛИНЕЙНОСТЬ ОЦЕНКИ ЧИСЛА ПОВТОРЕНИЙ И СИНТЕЗ ПРОСТЫХ РЕГУЛЯРНЫХ ЯЗЫКОВ

К.Х.Черанс
ВЦ ЛГУ им. П.Стучки

I. Введение

Известен ряд работ, в которых изучаются последовательности, не содержащие периодических подпоследовательностей (т.н. неповторяющейся последовательности). А.Туэ (A.Thue) /I/ в 1906 г. показал существование бесконечной неповторяющейся последовательности над 3-символьном алфавитом. Далее это направление развил Я.Бринкхюс (J.Brinkhuis) /2/, оценив число неповторяющихся троичных последовательностей. Ф.М.Декинг (F.M.Dekking) /3/ исследовал повторение и перекрывание блоков в двоичных последовательностях, а также бесконечные двоичные последовательности, имеющие периодические подпоследовательности со сколь угодно длинными периодами. Имеются также исследования по "строго неповторяющимся" последовательностями (см. /4/).

Однако естественно ставить вопрос и о последовательностях (словах), содержащих периодические подпоследовательности (подслова), оценить "меру повторимости" блоков в слове длины N над некоторым алфавитом. Переходим к точным определениям.

2. Теоремы линейности

Пусть S - некоторый алфавит (содержательно $|S| \geq 2$) и $A = x_1 x_2 .. x_N$ - слово над S ($x_i \in S$). Подслово $\Pi = x_p .. x_r$ слова A назовем периодическим с периодом $q \leq r-p$, если $x_i = x_{i+q}$ для $i = p, .., r-q$. Π назовем строго периодическим в A с периодом q , если

1^0 Π периодично с периодом q ;

2^0 Π непериодично с периодом q' , где q' - любой собственный делитель q ;

3^0 $x_{p-1} \neq x_{p-1+q}$ и $x_{r+1} = x_{r+1-q}$;

4^0 <u>коэффициент повторения</u> $k(\Pi) = \left[\frac{\ell(\Pi)}{q}\right] \geq 2$,

где $\ell(\Pi)$ - длина Π .

ТЕОРЕМА I. В слове длины n число строго периодических подслов не превышает $5.25n$.

Легко убедиться, что эта оценка по порядку не может быть улучшена: на словах вида $(I0I00I0I)^*$ число строго периодических подслов не менее $0.7n$ (для достаточно больших n).

Для каждого строго периодического подслова Π в A определим <u>реальный коэффициент повторения</u> $k'(\Pi) = k(\Pi) - 1$

<u>Числом повторений</u> в слове A назовем сумму $\sum_{\Pi} k'(\Pi)$, где Π пробегает множество всех строго периодических подслов слова A.

ТЕОРЕМА 2. В слове длины n число повторений не превышает $6n$.

Для слов длины n имеем нижнюю оценку числа повторений, равную $n-I$, достигаемую на слове $\underbrace{II\ldots I}_{n}$

Имеется гипотеза, что в слове длины n число повторений не превышает $n-I$ (она не доказана).

3. Синтез простых регулярных языков

В данном параграфе рассмотрим применение полученных линейных оценок при изучении индуктивного синтеза языков.

<u>Простым регулярным выражением</u> (над S) назовем регулярное выражение, включающее конкатенации и не более одной итерации Клини, т.е. выражение вида AB^*C, где $A, B, C \in S$

Простым регулярным языком назовем язык, описываемый простым регулярным выражением, например,

$(11)^*0$; $(1)^*$; $01(1)^*0$; $0(1010)^*1$; 110;

Конструкции, соответствующие простым регулярным языкам, существуют и в реальных языках программирования, например, в ПЛ/I.

⟨программа⟩ ::= (⟨оператор⟩)*

⟨идентификатор⟩ ::= ⟨буква⟩ (⟨символ⟩)*

⟨оператор-OPEN⟩ ::= OPEN FILE ⟨имя-файла⟩ (, FILE ⟨имя-файла⟩)* ;

Д.Англуин (D.Angluin) /5/ исследовала на синтезируемость в пределе по положительным данным некоторые подобные, более общие подклассы множества регулярных языков.

Пусть $\mathcal{V}$ - класс всех простых регулярных языков. Если $L \in \mathcal{V}$ описываем выражением AB^*C, будем иметь $L \rightleftharpoons AB^*C$. Язык $L \in \mathcal{V}$ будем считать заданным, если задано одно его описывающее выражение. Пусть $V_i(L) \stackrel{df}{=} AB^iC$, где $L \rightleftharpoons AB^*C$, $i=0,1,2,\ldots$ ..При $i \geq 2$ слово $V_i(L)$ назовем характерным для L.

Два простые регулярные выражения назовем эквивалентными, если они описывают один и тот же $L \in \mathcal{V}$. Например, $1(01)^*$ и $(10)^*1$ эквивалентны, но не одинаковы.

Рассмотрим эффективную процедуру M, которая работает следующим образом: в начале работы M получает входной параметр - слово над S. Затем она последовательно вырабатывает вопросы - слова над S, которые задает "оракулу". Оракул на каждый вопрос возвращает ответ "да" или "нет". В зависимости от ответа оракула M формирует следующий вопрос. В конечном итоге процедура M выдает некоторое простое регулярное выражение и останавливается.

Под оракулом языка L будем понимать "оракул", который на ему сообщенный вопрос B отвечает "да", если $B \in L$ и "нет" в остальных случаях.

Будем говорить, что M правильно синтезирует язык $L \in \mathcal{V}$, если при подаче M в качестве входного параметра характерного слова для L и при оракуле языка L, M в ре-

зультате работы выдает выражение, описывающее L.

Назовем M <u>алгоритмом правильного синтеза</u> (АПС) простых резулярных языков, если M правильно синтезирует каждый $L \in \mathcal{V}$.

ТЕОРЕМА 3. Существует алгоритм правильного синтеза $\mathcal{A}$ простых регулярных языков, который для каждого $L \in \mathcal{V}$ и каждого допустимого входного параметра $H \in L$ задает не более $6n$ вопросов (где $n = \ell(H)$).

Положим $\tilde{\ell}(L) = \ell(v_1(L))$ для $L \in \mathcal{V}$, и пусть $\mathcal{V}_n = \{L \in \mathcal{V} : \tilde{\ell}(L) = n\}$, $n = 1, 2, \ldots$. Будем говорить, что некоторым свойством E обладают <u>почти все</u> простые регулярные языки, если $\lim\limits_{n \to \infty} \frac{P_n^E}{P_n} = 1$, где $P_n = |\mathcal{V}_n|$, $P_n^E = |\{L \in \mathcal{V}_n : E(L)\}|$.

Подробное обсуждение этого понятия см. в /6/.

ТЕОРЕМА 4. Существует АПС $\mathcal{A}$, удовлетворяющий условию теоремы 3 и задающий ровно 2 вопроса для почти всех простых регулярных языков при любом допустимом значении входного параметра.

Пример такого алгоритма дан в §5.

§4. Схема доказательства теорем 1 и 2

Пусть $n_q(\alpha)$ - число строго периодических подслов в A, имеющих период q и коэффициент повторения α. Задача сводится к получению оценок

$$\sum_{\alpha=2}^{n} \sum_{q=1}^{\tilde{n}} n_q(\alpha) \le 5.25n \,, \quad \sum_{\alpha=2}^{n} \sum_{q=1}^{\tilde{n}} n_q(\alpha)(\alpha-1) \le 6n \,.$$

Требуемая оценка получается сложением линейных оценок для подмножеств множества строго периодических подслов слова A. Доказано:

$$\sum_{\alpha=3}^{n}\sum_{q=4}^{n}(\alpha-2)\,n_q(\alpha)\le\frac{n}{2} \qquad (I)$$

$$\sum_{\alpha=3}^{n}\sum_{q=4}^{n}n_q(\alpha)\le\frac{n}{2} \qquad (I.1)$$

$$\sum_{\alpha=3}^{n}\sum_{q=4}^{n}n_q(\alpha)(\alpha-1)\le n \qquad (I.2)$$

$$\sum_{\alpha=2}^{n}\sum_{q=1}^{3}n_q(\alpha)\le\frac{3n}{4} \qquad (2)$$

$$\sum_{\alpha=2}^{n}\sum_{q=1}^{3}n_q(\alpha)(\alpha-1)\le n-1 \qquad (3)$$

$$(1.2)\&(3)\Rightarrow\sum_{q=1}^{n}\sum_{\alpha=3}^{n}n_q(\alpha)(\alpha-1)\le 2n \qquad (4.I)$$

$$\sum_{q=1}^{n}\sum_{\alpha=3}^{n}n_q(\alpha)\le n \qquad (4.2)$$

$$\sum_{q=4}^{n}n_q(2)\le 4n \qquad (5)$$

При доказательстве большинства оценок неоднократно применяется

ЛЕММА I. Пусть $A=x_1..x_\kappa$, $\kappa\ge a+b-d(a,b)$; $a,b\in N$ Тогда при $x_i=x_{i+a}$ для $i=1,..,\kappa-a$ и $x_j=x_{j+b}$ для $j=1,..,\kappa-b$, имеем $x_s=x_{s+d}$ для $s=1,..,\kappa-d$, где $d=d(a,b)$ - наибольший общий делитель чисел a и b.

Для получения (I), (2) и (3) у каждого строго периодического подслова Π выделяем "активную часть" $\mathcal{Z}(\Pi)$ и показываем, что $\mathcal{Z}(\Pi_1)\cap\mathcal{Z}(\Pi_2)\ne e\Rightarrow\Pi_1=\Pi_2$

(где ε - пустое слово). При доказательстве (I) сначала разбиваем все строго периодические подслова на

$$P_j = \{\Pi : 2^j \le q(\Pi) \le 2^{j+1} - 1\}, \; j \in N, \; j \ge 2.$$

От непересечении активных частей получаем указанные оценки.

При оценке (5) существенным оказывается понятие структуры повторений. Поясним это понятие. При определении строго периодического подслова опускаем требование 2^0, получаем определение [строго] псевдопериодического подслова Π в слове A. При $k(\Pi) = 2$ определим представительную часть Π, как любое подслово $\bar{\Pi}$ слова Π, имеющее длину $\ell(\bar{\Pi}) = k(\Pi) \cdot q = 2q$.

Структурой повторений $\mathcal{R}$ назовем множество псевдопериодических подслов, если

1^0 $|\mathcal{R}| \ge 3$ и $\forall \Pi \in \mathcal{R} \;\; k(\Pi) = 2$;

2^0 у всех $\Pi \in \mathcal{R}$ можно выбрать представительные части так, чтобы их центры совпадали;

3^0 представительная часть длиннейшего $\Pi \in \mathcal{R}$ выбрана в крайне левом из возможных положений;

4^0 периоды для $\Pi \in \mathcal{R}$ образуют арифметическую прогрессию с разностью $d \in N$ и первым членом между d и $2d$;

5^0 псевдопериодические подслова в $\mathcal{R}$, имеющие $q > 2d$ - строго периодические подслова слова A.

Структуру повторений $\mathcal{R}$ будем называть полной, если не существует структуры $\mathcal{R}_1 \supsetneq \mathcal{R}$.

Суммируемой частью полной структуры $\mathcal{R}$ назовем $sum(\mathcal{R}) = \mathcal{R} \setminus \{\Pi_0\}$, где Π_0 - строго периодическое подслово с наибольшим периодом в $\mathcal{R}$.

По определению строго периодического подслова можно получить, что если $q_1 < q_0 < q_2 < 2q_1$, $R_1 < R_0 \le R_2$ (или $R_1 > R_0 \ge R_2$), $|R_0 - R_1| \ge d(q_2 - q_0, q_0 - q_1)$, то $|R_2 - R_1| > q_1$, где у строго периодических подслов Π_1, Π_0 и Π_2 ($k(\Pi_1) = k(\Pi_0) = k(\Pi_2) = 2$) периоды q_1, q_0 и q_2 и центры представительных частей R_1, R_0 и R_2 соответственно.

Отсюда, разбив строго периодические подслова по P_j, получим оценку $\leq 2n$ для числа строго периодических подслов, не входящих в суммируемые части полных структур. Число входящих же удается оценить по (4.I).

5. Алгоритм синтеза

Сначала введем ряд вспомогательных понятий. Отбросив в определении строго периодического подслова требование 4^0, получим определение [строго] полупериодического подслова Π в слове A.

Для $i \in N_+$ и простого регулярного выражения AB^*C ($\ell(B) > 0$) определим $\varphi(AB^*C, i) = \Pi$, где Π полупериодическое подслово слова AB^iC, содержащее B^i, и при этом $\ell(B) = t \cdot q(\Pi)$, $t \in N_+$. Заметим, что φ - всюду определено, однозначно, сюръективно, а также эффективно находимо по 2 словам из семейства AB^*C, т.е. языка $L \rightleftharpoons AB^*C$.

Пусть $\dot{I}$ - класс всех простых регулярных выражений над S, $\equiv$ - отношение эквивалентности выражений в $\dot{I}$. Ввиду сказанного для $A_1B_1^*C_1 \equiv A_2B_2^*C_2$ $\varphi(A_1B_1^*C_1, i) = \varphi(A_2B_2^*C_2, i)$ при $i = 1,2,\ldots$ тем самым φ может быть определено на $\mathcal{O} = \dot{I}/\equiv$ (исключив случай $\ell(B) = 0$).

Для каждого полупериодического подслова Π (в любом слове над S) имеем конечность и эффективную находимость $\varphi^{-1}(\Pi)$, а также имеем возможность эффективно найти $L \in \mathcal{O}$ по $V_0(L)$ и $V_1(L)$. Ввиду характерности входного слова $H \in L$ имеем для полупериодического подслова $\Pi = \varphi(L, i)$ слова H $\kappa(\Pi) \geq i \geq 2$.

Опишем алгоритм $\mathcal{A}$, удовлетворяющий теореме 4. $\mathcal{A}$ будет выбирать подряд все строго периодические подслова Π_j, поданного характерного слова $H \in L$, выясняя для каждого из них путем вопросов, существует ли такое $m \in N_+$, $m \geq 2$ $\varphi(L, m) = \Pi_j$, до получения положительного ответа.

Положим G_{ij} - слово, полученное от H вычеркиванием у него $(\kappa(\Pi_j) - i) \cdot q(\Pi_j)$ последовательных символов

из Π_j . Алгоритм $\mathcal{A}$ работает следующим образом:

I. (Организация выбора). Найти из оставшихся строго периодических подслов то, которое имеет наибольший период. Перейти к 2. Если все строго периодические подслова в H исчерпаны, перейти к 3 со значением (H,H).

2. (Проверка). Для $i = 0,1,..\kappa(n)-2$ до появления второго положительного ответа

I) построить слово G_{ij} ,

2) задать вопрос о $G_{ij} \in L$.

- Если нет положительных ответов, отметить Π_j как рассмотренное, перейти к I.

- Если получено два положительных ответа при $i=k$ и $i=\ell$ $(k<\ell)$, то перейти к 3 со значением $(G_{kj}, G_{k\ell})$.

- Если получен один положительный ответ, перейти к 3 со значением $(G_{\kappa(n)-2,j}, G_{\kappa(n)-1,j})$

3. (Синтез). Полученной паре слов $(V_0(L), V_1(L))$ построить и выдать L . Кончить работу.

Не вдаваясь в подробности , отметим, что оценку числа вопросов в общем случае получаем по теореме 2. Ввиду свойств отображения φ имеем по крайней мере линейную верхнюю оценку числа вопросов для любого АПС.

Имея описанную организацию выбора, можно показать, что $\mathcal{A}$ почти во всех случаях в первых двух вопросах задаст соответственно $V_0(L)$ и $V_1(L)$ (при любом характерном слове), где L - синтезируемый язык. Идея доказательства состоит в сопоставлении одного единственного описывающего регулярного выражения каждому простому регулярному языку.

При оценке числа простых регулярных выражений, сопоставленных с языком сперва зафиксируем n , а также способ выборки итерации Клини, затем при оценке числа выражений не синтезируемых за 2 вопроса сведем случай существования "негодного" характерного слова к негодности слова $V_{n+1}(L)$.

Оценим сверху вероятность того, что в слове $V_{n+1}(L)$ существует строго периодическое подслово с периодом большим чем $\ell(b)$, откуда, освободив положение итерации Клини, получим оценку отношения .

Заметим, что при оценке синтеза требование характерности H для L необходимо: если рассмотреть процедуры, синтезирующие также по $V_1(L)$, то ввиду квадратичности числа полупериодических подслов и свойств имеем по крайней мере квадратичную верхнюю оценку числа задаваемых вопросов для любого такого алгоритма.

ЛИТЕРАТУРА

1. Thue A. Über unendliche Zeichenreichen// Norske Vid. Selsk. Skr. I Mat.-Nat.Kl.(Kristiania)-7-1906. - С. 1-22.
2. Brinkhuis J. Non-Repetitive Sequences on Three Symbols// Quart. J.Math. Oxford(2) -34-1983. - С.145-149.
3. Dekking F.M. On repetitions of blocks in binary sequences// J.Combin.Theory(A) -20-1976. - С.292-299.
4. Dekking F.M. Strongly Non-Repetitive Sequences and Progression-Free Sets// J.Combin.Theory(A) - 27 - 1979. - С.181-185.
5. Angluin D. Finding Patterns Common to a Set of Strings // Proc. 11th Annual Symposium on Theory of Computing - 1979. - ACM.
6. Трахтенброт Б.А., Барздинь Я.М. Конечные автоматы: Поведение и синтез. - М.: Наука, 1970. - 400 с.

МОДЕЛЬ БАЗЫ ДАННЫХ С НЕДЕТЕРМИНИРОВАННЫМИ ПЕРЕХОДАМИ

Я.П. Цирулис
ЛГУ им. П.Стучки

I. В в е д е н и е. В [2] была предложена модель реляционной базы данных, представляющая собой автомат вида (S,Q,A) с заданной операцией $*: S*Q \to B$, где S - множество состояний базы, Q - алгебра запросов, A - подходящая алгебра отношений, а $s*q$ понимается как отношение, соответствующее в состоянии s запросу q. Там же коротко обсуждалась модель базы данных с изменяемыми состояниями - автомат (S,Σ,Q,A), где Σ - система команд управления и задана дополнительная операция $\circ: S*\Sigma \to S$; кроме того, Σ еще должно определенным образом действовать на Q. Это модель с детерминированными переходами. В [3], [4] была описана подобная модель с недетерминированными переходами, когда Σ действует на Q неоднозначно.

Здесь мы более подробно чем в [3], [4] обсудим предложенную там модель, точнее, некоторый более общий ее вариант. Под базой данных будем понимать автомат вида (S,Σ,Q,A), где S, Σ, Q, A имеют прежний смысл и заданы операции

$$\circ: S*\Sigma \to \mathcal{P}(S), \quad *: S*Q \to A, \quad \diamond: \Sigma*Q \to Q,$$

подчиняющиеся, разумеется, определенным условиям. Все необходимые уточнения приводятся ниже. Для простоты мы ограничиваемся случаем, когда рассматриваются не алгебры, а лишь множества данных, причем все эти множества односортные. Работа не содержит существенных доказательств; основным ее результатом следует считать предложенное точное определение базы.

2. А л г е б р ы о т н о ш е н и й. Пусть в дальнейшем V - фиксированное бесконечное множество, а V^* - множество всех конечных подмножеств V, включая пустое. Элементы V будем называть атрибутами, а множества из V^* - родами.

В литературе можно найти много различных вариантов по-

нятия алгебры отношений, и в принципе не важно, какое из них использовать в определении базы данных. Те варианты, где набор аргументов отношения не предполагается упорядоченным, можно сгруппировать в два класса следующим образом. Можно называть отношением рода X над множеством данных D любое подмножество множества D^X и в основе алгебры отношений положить семейство $(Rel_X : X \in V^*)$ множеств всех отношений любых родов. Тогда получается многосортная алгебра. Но часто удобно отождествлять отношения, различающиеся лишь т.наз. фиктивными (несущественными) аргументами и, в частности, рассматривать каждое отношение и как отношение любого большего рода. Тогда отношениями называют подмножества множества D^V, "зависящие" лишь от конечного числа атрибутов. Точнее, в этом случае отношением рода X над D называется любое такое множество $R \subset D^V$, что для любых двух $\varphi, \psi \in D^V$

$$\varphi \in R, \quad \psi | X = \varphi | X \rightarrow \psi \in R$$

или, что то же самое,

$$\psi | X = \varphi | X \rightarrow (\psi \in R \leftrightarrow \varphi \in R).$$

Пусть Rel_X по-прежнему означает множество всех отношений рода X, и пусть $Rel := U(Rel_X : X \in V^*)$ - множество всех отношений над D любых родов; это множество потом тем или иным путем превращается в алгебру - на этот раз односортную. Понятно, что действительно $Rel_X \subset Rel_Y$ при $X \subset Y$. Оба подхода тесно связаны между собой (см., например, [5]) и по-существу даже равнозначны. В [3], [4] мы придерживались первой точки зрения как несколько более естественной, но здесь выберем вторую: она более удобна технически.

Множество Rel (в случае необходимости будем писать $Rel(D)$), также как и каждое из множеств Rel_X, замкнуто относительно теорико-множественных операций объединения, пересечения и дополнения и является поэтому булевой алгеброй. Опять-таки, для наших целей не важно, какие другие, специфические для отношений операции на Rel будут выбраны. Для определенности выберем на Rel структуру, уже известную в алгебраической логике. Пусть $R \in Rel$ и $x, y \in V$; тогда положим:

$\exists_x R := \{\tau \in D^V$: для некоторого $\tau' \in R$ и всех $z \neq x$ $\tau z = \tau' z\}$,
$d_{xy} := \{\tau \in D^V : \tau x = \tau y\}$.

Тогда множество Rel, рассматриваемое как алгебра сигнатуры $\Lambda := \{\vee, \wedge, -, \exists_x, d_{xy}\}_{x,y \in V}$ становится цилиндрической алгеброй множеств [5]. Понятно, что $d_{xy} \in Rel_{\{x,y\}}$ и что если $R \in Rel_X$, то $\exists_x R \in Rel_{X-\{x\}}$. Всякое подмножество Rel, замкнутое относительно операций из Λ и содержащее все диагонали d_{xy}, будем называть алгеброй отношений над D. Если A такая алгебра, через A_X обозначим пересечение $A \cap Rel_X$.

3. Языки запросов и команд. Зафиксируем теперь некоторое множество $\underline{R}$, каждому элементу r которого приписан некоторый род $\rho(r) \subseteq V$. Элементы $\underline{R}$ будем называть символами отношений. Определяемый этим множеством язык реляционных выражений – алгебра сигнатуры Λ, свободно порождаемая множеством $\underline{R}$ – мог бы служить языком запросов для базы данных (любопытно, что он с точностью до технических деталей совпадает с языком логической системы $\mathcal{L}$ из [6]). Однако мы хотим учитывать в запросах и команды управления (см. определение операции $\circ$ в разделе I) и поэтому должны расширить этот язык, допуская на нем действие команд управления. В результате получится некоторая разновидность языка динамической логики.

Пусть зафиксировано еще одно множество – множество $\underline{C}$ простых команд управления. Их возможная структура и смысл нас сейчас не будет интересовать (но см. [4]). Будем считать что разрешается из простых команд с помощью подходящих операций образовать и составные. Для определенности допустим, что имеются двуместные операции $\cdot$ и $|$, одноместная операция * и нульместная операция ε. Тогда язык команд управления $L_{\underline{C}}$ представляет собой алгебру сигнатуры $\theta := \{\cdot, |, ^*, \varepsilon\}$, свободно порождаемую множеством $\underline{C}$. Возвращаясь к языку реляционных выражений, будем теперь считать, что им является алгебра $L_{\underline{R}}$ сигнатуры $\Lambda \cup L_{\underline{C}}$, свободно порождаемая множеством $\underline{R}$. Целесообразно стать на более абстрактную точку зрения и соединить

оба языка в одну двусортную алгебру $L:=(L_{\underline{C}}, L_{\underline{R}})$ сигнатуры $\Xi:=$ $=\Theta\cup\Lambda\cup\{\circ\}$, где $\circ$ - операция смешанного типа $L_{\underline{C}}\times L_{\underline{R}}\to L_{\underline{R}}$, определяемая условием $m\circ f=mf$; здесь справа - выражение языка $L_{\underline{R}}$, в котором m понимается уже как операция на $L_{\underline{R}}$. Алгебра L свободно порождается "двусортным" множеством $(\underline{C},\underline{R})$, т.е является, по терминологии [7], алгеброй Ξ-термов над множеством "переменных" $\underline{C}\cup\underline{R}$.

Мы будем иметь дело и с произвольными алгебрами указанных сигнатур. Если, например, (Σ,Q) - алгебра сигнатуры Ξ, будем называть операцию $\circ$ действием алгебры Σ на Q и писать σq вместо $\sigma\circ q$, как бы превращая этим Q в $(\Lambda\cup L_{\underline{R}})$-алгебру наподобие $L_{\underline{R}}$. Нам будет удобно рассматривать также расширенные сигнатуры $\Theta^*:=\Theta\cup\underline{C}$, $\Lambda^*:=\Lambda\cup\underline{R}$ и $\Xi^*:=\Theta^*\cup\Lambda^*\cup\{\circ\}$, где элементы $\underline{C}$ и $\underline{R}$ понимаются как дополнительные константы. Тогда каждая из алгебр $L_{\underline{C}}$, $L_{\underline{R}}$ и L оказывается (свободной) алгеброй с пустым множеством порождающих, или т.наз. инициальной алгеброй [7] в классе всех алгебр соответствующей сигнатуры. Это означает, что для каждой Θ^*-алгебры Σ имеется единственный гомоморфизм $\chi_{\underline{C}}: L_{\underline{C}}\to\Sigma$, а для каждой $(\Lambda^*\cup L_{\underline{C}})$-алгебры Q - единственный гомоморфизм $\chi_{\underline{R}}: L_{\underline{R}}\to Q$, и что пара $\chi:=(\chi_{\underline{C}},\chi_{\underline{R}})$ является единственным "двусортным" гомоморфизмом Ξ^*-алгебры $(L_{\underline{C}},L_{\underline{R}})$ в (Σ,Q). Эти гомоморфизмы будем называть каноническими.

Произвольную алгебру сигнатуры Ξ^* будем называть минимальной, если она не имеет собственных подалгебр. Очевидно, что алгебра минимальна т.т.т., когда она является гомоморфным образом L, и что гомоморфный образ любой минимальной алгебры сам минимален.

4. О п р е д е л е н и е б а з ы д а н н ы х. Если задана область данных D, то интерпретацией языка $L_{\underline{R}}$ в D будем называть любое отображение $i:\underline{R}\to Rel(D)$, сохраняющее роды (т.е. такое, что $i(r)\in Rel_{ar(r)}(D)$). Пусть Int - множество всех интерпретаций. Можно было бы уже четверку $(Int, L_{\underline{C}}, L_{\underline{R}}, Rel)$ превратить в автомат интересующего нас вида. Мы все же будем допускать более общие конструкции. Прежде всего, часто не все

интерпретации обязательно реализуются в базе как возможные ее состояния, а кроме того, мы не хотим исключать ситуации, когда разные состояния реализуют одну и ту же интерпретацию. Далее, желательно учитывать ту или иную "степень эквивалентности" выражений языков $L_{\underline{C}}$ и $L_{\underline{R}}$ и поэтому вместо самих языков использовать, скажем, какие-либо их фактор-алгебры. Наконец, обычно не все отношения из Rel доступны в базе, тем или иным образом "встроены"в нее. Поэтому разумно в качестве четвертой компоненты автомата допускать произвольные алгебры отношений над D. Учитывая все сказанное, а также некоторые другие соображения, приходим к следующим образом уточненному определению базы данных.

ОПРЕДЕЛЕНИЕ. Базой данных над множеством D называется всякая система (S,Σ,Q,A) с операциями $\circ: S\times\Sigma\to\mathcal{P}(S)$, $*: S\times Q\to A$, $\circ: \Sigma\times Q\to Q$, где S - произвольное множество, A - какая-либо алгебра данных над D, Σ - алгебра сигнатуры Θ^*, Q - алгебра сигнатуры Λ^*, и выполняются следующие условия (для $S'\subset S$ мы пишем $S'\circ\sigma$ вместо $\{s\circ\sigma : s\in S'\}$ и $S'*q$ вместо $\{s*q : s\in S'\}$):

(i) $s\circ(\sigma_1\cdot\sigma_2)=(s\circ\sigma_1)\circ\sigma_2$, $s\circ(\sigma_1|\sigma_2)=s\circ\sigma_1\cup s\circ\sigma_2$, $s\circ\varepsilon=\{s\}$, $s\circ\sigma^*=\bigcup_{n\geq 0}(s\circ\sigma^n)$, где $\sigma^0=\varepsilon$ и $\sigma^{n+1}=\sigma\sigma^n$;

(ii) $s*(q_1\vee q_2)=s*q_1\cup s*q_2$, $s*(q_1\wedge q_2)=s*q_1\cap s*q_2$, $s*\neg q=-(s*q)$, $s*\exists x q=\exists x(s*q)$, $s*d_{xy}=d_{xy}$;

(iii) $s*\tau\in A_{alg(\tau)}$;

(iv) $s*(\sigma\circ q)=\cup((s\circ\sigma)*q)$;

(v) Ω^+-алгебра (Σ,Q) минимальна.

Алгебра (Σ,Q) называется входной алгеброй базы.

Мы не включили в схему базы какие-либо ограничения на состояния (например, условия целостности), как это иногда делают. В нашем случае такие ограничения - аксиомы базы - имеют вид тождеств в алгебре L. Рассмотрим этот вопрос подробнее.

Пусть $\mathcal{B}:=(S,\Sigma,Q,A)$ - какая-либо база данных. Два элемента σ_1,σ_2 из Σ или q_1,q_2 из Q будем называть неразличимыми в $\mathcal{B}$ (обозначения: $\sigma_1\overset{\mathcal{B}}{\equiv}\sigma_2$, $q_1\overset{\mathcal{B}}{\equiv}q_2$), если для любого состояния s из S $s\circ\sigma_1=s\circ\sigma_2$ и, соответственно, $s*q_1=s*q_2$. Мы можем говорить и

о двусортном отношении неразличимости $\overset{B}{\sim} = (\overset{B}{\sim}_\Sigma, \overset{B}{\sim}_Q)$ на алгебре (Σ, Q). Теперь, следуя примеру общей алгебры, назовем системой тождеств в L любое двусортное бинарное отношение $E = (E_\underline{C}, E_\underline{R})$ на L: $E_\underline{C} \subset L^2_\underline{C}$, $E_\underline{R} \subset L^2_\underline{R}$. Будем далее говорить, что база B удовлетворяет системе тождеств E, если для любого $\underline{C}$-тождества $(m_1, m_2) \in E_\underline{C}$ и любого $\underline{R}$-тождества $(f_1, f_2) \in E_\underline{R}$ элементы $\lambda_\underline{C} m_1$ и $\lambda_\underline{C} m_2$, а также $\lambda_\underline{R} f_1$ и $\lambda_\underline{R} f_2$ неразличимы в B. Две базы можно считать эквивалентными, если они удовлетворяют одним и тем же тождествам.

5. С ж а т и е а л г е б р ы L. Входную алгебру базы $\mathcal{B} = (S, \Sigma, Q, A)$ можно считать в некотором смысле избыточной, если в Σ или в Q имеются неравные неразличимые элементы. Если такой ситуации нет, т.е. если отношение неразличимости $\overset{B}{\sim}$ оказывается двусортным отношением равенства на (Σ, Q), будем называть базу редуцированной. Мы увидим ниже, что, не теряя общности, можно ограничиться рассмотрением лишь редуцированных баз.

Пусть теперь $\mathcal{B}$ - какой-либо класс баз данных в схеме $(\underline{C}, \underline{R})$. Алгебру L можно использовать как язык для описания свойств баз из этого класса. Однако нетрудно догадаться, что профакторизовав ее по системе $E(\mathcal{B})$ всех тождеств, которым удовлетворяют члены $\mathcal{B}$, мы должны получить менее избыточный язык, пригодный для тех же целей. Покажем, в каком смысле это действительно так.

Сперва построим одно важное семейство Σ-алгебр. Пусть заданы множество S и область данных D. Множество $\mathcal{P}(S^2)$ всех бинарных отношений на S можно рассматривать как θ-алгебру относительно операций композиции, объединения и взятия рефлексивного транзитивного замыкания, а также отношения тождества на S. В то же время на множестве $Rel(D)^S$ всех функций типа $S \to Rel(D)$ стандартным образом (поточечно) индуцируется структура алгебры, однотипной с $Rel(D)$. Следующим образом определим действие $\bullet$ алгебры $\mathcal{P}(S^2)$ на $Rel(D)^S$:

$$(\varphi\bullet\psi)(s) := \bigcup (\psi(s') : s' \in \varphi(s)),$$

где $\varphi(s) := \{s' \in S : s \varphi s'\}$. Полученную в результате Σ-алгебру

$(\mathcal{P}(S^2), Rel(D)^S)$ обозначим через $[S,D]$. Нам понадобится

ЛЕММА. Пусть заданы Ξ-алгебра (Σ,Q), а также операции $\circ: S\times\Sigma\to\mathcal{P}(S)$, $*: S\times Q\to Rel(D)$ и отображения $\beta_\Sigma: \Sigma\to\mathcal{P}(S^2)$, $\beta_Q: Q\to Rel(D)^S$ такие, что

$$s\circ\sigma = \rho_\sigma(s), \qquad s * q = \xi_q(s),$$

где $\rho_\sigma=\beta_\Sigma(\sigma)$, $\xi_q=\beta_Q(q)$. Тогда операции $\circ$ и $*$ в системе $(S,\Sigma,Q,Rel(D))$ подчиняются условиям (i), (ii), (iv) из определения базы данных в том и только в том случае, когда пара $\beta=(\beta_\Sigma,\beta_Q)$ является гомоморфизмом алгебры (Σ,Q) в $[S,D]$, т.е. когда выполнены соотношения

(i') $\rho_{\sigma_1\sigma_2}=\rho_{\sigma_1}\rho_{\sigma_2}$, $\rho_{\sigma_1\cup\sigma_2}=\rho_{\sigma_1}\cup\rho_{\sigma_2}$, $\rho_{\sigma^*}=tr(\rho_\sigma)$, $\rho_\varepsilon=\Delta$;

(ii') $\xi_{q_1\vee q_2}=\xi_{q_1}\vee\xi_{q_2}$, $\xi_{q_1\wedge q_2}=\xi_{q_1}\wedge\xi_{q_2}$, $\xi_{\neg q}=\neg\xi_q$, $\xi_{\exists x\xi}=\exists_x\xi$, $\xi_{d_{xy}}=d_{xy}$;

(iv') $\xi_{\sigma q}=(\rho_\sigma\xi_q)$,

где Δ – отношение тождества, а tr – операция транзитивного замыкания.

Мы опустим доказательство, заключающееся в проверке большого числа простых соотношений. В следующей теореме $E(B)$ означает множество всех тождеств, которым удовлетворяет база B.

ТЕОРЕМА. Для каждой базы $B=(S,\Sigma,Q,A)$ отношения $\overset{B}{\sim}$ и $E(B)$ являются конгруэнциями на алгебрах L и (Σ,Q), и соответствующие им фактор-алгебры L/B и $(\Sigma,Q)/B$ минимальны и изоморфны между собой.

ДОКАЗАТЕЛЬСТВО. Отношение $\overset{B}{\sim}$ является, очевидно, ядерной эквиваленцией гомоморфизма β из леммы, поэтому оно является Ξ-конгруэнцией, а значит и Ξ^*-конгруэнцией на (Σ,Q). В свою очередь $E(B)$ является ядерной эквиваленцией Ξ-гомоморфизма $\beta\lambda: L\to[S,D]$, где λ – канонический гомоморфизм L в (Σ,Q), поэтому и $E(B)$ является конгруэнцией алгебры L. А т.к. $(\beta\lambda)(L)=\lambda(\Sigma,Q)$, обе фактор-алгебры изоморфны одной и той же подалгебре алгебры $[S,D]$. Их минимальность вытекает непосредственно из определений.

СЛЕДСТВИЕ.1. Каждая база эквивалентна редуцированной базе, имеющей то же множество состояний и ту же алгебру отношений.

СЛЕДСТВИЕ 2. Пусть $\mathcal{B}$ - произвольный класс баз. Отношение $E(\mathcal{B})$ является конгруэнцией на L, и для любой базы $B = (S,\Sigma,Q,A)$ из $\mathcal{B}$ существует единственный Σ^*-гомоморфизм $L/\mathcal{B} \to (\Sigma,Q)/B$.

ДОКАЗАТЕЛЬСТВО. Понятно, что $E(\mathcal{B})$ является пересечением всех конгруэнций $E(B)$, где $B \in \mathcal{B}$, и поэтому и само является конгруэнцией. А т.к. $E(\mathcal{B}) \subset E(B)$, алгебра L/B, а значит и $(\Sigma,Q)/B$ изоморфна фактор-алгебре алгебры $L/\mathcal{B}$. Ввиду минимальности этих алгебр натуральный гомоморфизм $L/\mathcal{B} \to (\Sigma,Q)/B$ будет единственным.

Добавим еще, что в силу двусортного аналога известной теоремы общей алгебры (см., например, [1], предложение 7.II главы 2) $L/\mathcal{B}$ является подпрямым произведением всех L/B (или $(\Sigma,Q)/B$).

Итак, алгебра $L/\mathcal{B}$ обладает некоторым свойством свободы относительно баз данных из $\mathcal{B}$, и ее можно рассматривать как некоторый язык для описания свойств этих баз. Таким образом, тождества для класса $\mathcal{B}$ действительно можно писать в языке $L/\mathcal{B}$; то, в каком случае база из $\mathcal{B}$ удовлетворяет некоторой системе таких тождеств, определяется в точности так же, как выше.

6. З а к л ю ч и т е л ь н ы е з а м е ч а н и я. Для баз данных имеется естественное понятие гомоморфизма, с помощью которого класс всех баз в фиксированной схеме превращается в категорию. Имеет несомненный практический интерес и сравнение баз, определенных в разных схемах. Отметим еще одну чисто алгебраическую задачу: описать класс всевозможных алгебр вида $L/\mathcal{B}$ (рассматриваемых с точностью до изоморфизма) аксиоматически. Нетрудно понять, что если $(\Sigma,Q)^*$ - такая алгебра, то Q должна быть (локально конечной) цилиндрической алгеброй, Σ - регулярной алгеброй Клини и и что элементы Σ, так же как в случае динамической алгебры, действуют на Q как аддитивные, сохраняющие булев ноль операторы; они, кро-

ме того, перестановочны со всеми кванторами и сохраняют все диагонали. Но полное решение указанного вопроса трудно; оно включает в себе, как частные случаи, теоремы представления и для цилиндрических, и для динамических алгебр.

Идеи привлечения методов динамической логики в теорию баз данных принадлежит Б.И.Плоткину. Им же была поставлена задача определения понятия "динамической" базы данных.

ЛИТЕРАТУРА

I. Кон П. Универсальная алгебра.-М.: Мир, 1968.-351 с.

2. Плоткин Б.И. Алгебраическая модель базы данных-автомата// Латв. мат. ежегодник.-Рига: Зинатне, 1983.-Вып. 27.-С. 216-232.

3. Цирулис Я.П. Разновидность динамической логики в теории реляционных баз данных//7-я Всес. конф. "Проблемы теор. киберн.". Тезисы докл. Ч. I.-Иркутск, 1985.-С.197-198.

4. Цирулис Я.П. К логической теории реляционных баз данных// Всес. конф. по прикл. лог. Тезисы докл.-Новосибирск, 1985.-С.225-227.

5. Imielinski T., Lipski W. The relational model of data and cylindric algebras//J. Comp. Syst. Sci.-1984.-Vol. 28.-P.80-102.

6. Monk D. Substitutionless predicate logic with identity// Aerch math. Log. Grundl.-1965.-Vol.7,N 3-4.-P.102-121-

7. Reichel H. Structural induction on partial algebras// Berlin: Academie-Verlag, 1984.-205 p.

РАСПОЗНАВАНИЕ ЯЗЫКОВ В ОДНОБУКВЕННОМ АЛФАВИТЕ ВЕРОЯТНОСТНЫМИ АВТОМАТАМИ С МАГАЗИННОЙ ПАМЯТЬЮ

Я.Я.Канепс
ВЦ ЛГУ им. П. Стучки

Детерминированный односторонний автомат с магазинной памятью имеет входную ленту, на которой головка может стоять на месте или двигаться слева направо (но не наоборот), и один магазин (рабочую ленту, на которой головка может производить записи только в крайней правой ячейке; для чтения записей, расположенных левее, головка должна предварительно уничтожить записи, которые расположены правее). Работа автомата задается программой, состоящей из команд. Команды по внутреннему состоянию в очередной момент и по буквам, обозреваемым головками на входной ленте и на магазине (эту тройку будем называть левой частью команды), определяет следующее внутреннее состояние, движения головок и, возможно, букву, записываемую в магазин (эту четверку будем называть правой частью команды). Во множестве внутренних состояний выделены начальное состояние и два заключительных состояния (принимающее и отвергающее). Входное слово принимается (отвергается), если в результате его обработки автомат приходит в принимающее (отвергающее) заключительное состояние.

Недетерминированный односторонний автомат с магазинной памятью отличается от детерминированного тем, что программа может содержать команды с одинаковыми левыми, но с различными правыми частями. Говорят, что недетерминированный автомат принимает данное входное слово, если существует такая последовательность выполняемых команд, при которой автомат, работая на данном входном слове, приходит к принимающему заключительному состоянию. Подробное определение детерминированных и недетерминированных односторонних автоматов с магазинной памятью приведено в / I /.

Указанные выше типы автоматов значительно сильнее конечных автоматов. Например, язык принимается недетерминированным односторонним автоматом с магазинной памятью тогда и только тогда, когда этот язык является бесконечным.

Известно, что класс бесконечных языков существенно шире класса языков, распознаваемых конечными автоматами / I /.

С другой стороны, из теоремы Парика / I / вытекает, что если язык в однобуквенном алфавите принимается недетерминированным (или, тем более, детерминированным) односторонним автоматом с магазинной памятью, то этот язык распознается и конечным детерминированным автоматом.

Вероятностный односторонний автомат с магазинной памятью отличается от детерминированного тем, что левая часть всех команд зависит еще от выходного значения простейшего бернуллиевского датчика случайных чисел, осуществляющего равновероятный выбор символа из конечного алфавита. Говорят, что вероятностный автомат распознает язык L с вероятностью p ($p > \frac{1}{2}$), если для любого входного слова x с вероятностью, не меньшей чем p, автомат принимает x, если $x \in L$, и отвергает x, если $x \notin L$. Говорят, что вероятностный автомат распознает язык L с изолированной точкой сечения, если существует такое $p > \frac{1}{2}$, что автомат распознает L с вероятностью p.

ТЕОРЕМА. Если некоторый язык в однобуквенном алфавите распознается вероятностным односторонним автоматом с магазинной памятью с изолированной точкой сечения, то этот язык распознается и конечным детерминированным автоматом.

Доказательство этой теоремы технически сложно. Из-за недостатка места оно будет опубликовано в другом издании.

Для сравнения отметим, что аналог нашей теоремы для языков в более богатых алфавитах не имеет места. В /2/ доказано, вероятностные односторонние автоматы с магазинной памятью могут распознавать языки, не являющиеся бесконтекстными.

ЛИТЕРАТУРА

I. Гладкий А.В. Формальные грамматики и языки. - М.: Наука, 1973. - 368 с.

2. Фрейвалд Р.В. Распознавание языков с высокой вероятностью на различных классах автоматов // Доклады АН СССР. - 1978. - Т.239, № I. - С. 60-62.

О ВЫЧИСЛИТЕЛЬНОЙ СИЛЕ ДВУХЛЕНТОЧНЫХ КОНЕЧНЫХ АЛЬТЕРНИРУЮЩИХ АВТОМАТОВ

М.Я.Альбертс
ВЦ ЛГУ им.П.Стучки

В [I] было введено мощное обобщение понятия недетерминированных машин - понятие "альтернирующая машина". Альтернирующие конечные автоматы распознают только регулярные языки, и альтернирующие машины Тьюринга - только рекурсивно перечислимые множества [I]. В [I] введена естественная иерархия альтернирующих машин. В настоящей работе рассматривается та же иерархия многоленточных конечных автоматов и доказывается, что возможности различных типов автоматов этой иерархии различны на нижних этажах. n-ленточный конечный альтернирующий автомат есть обычный n-ленточный конечный недетерминированный автомат, множество состояний которого разделено на множество экзистенциальных /$\exists$/ и множество универсальных /$\forall$/ состояний.

Автомат имеет управляющий элемент с конечным числом состояний и обрабатывает n-ки слов. На каждой ленте написано слово в конечном алфавите Σ, которое заканчивается на специальном символе $\#$. Каждая лента имеет одну только читающую головку, которая в каждый момент может либо стоять на месте, либо двигаться вправо. Один шаг автомата состоит в том, что он переходит в новое состояние и меняет положение головок на лентах согласно функции переходов или переходит в новую конфигурацию. Переход из некоторой конфигурации α в другую конфигурацию β обозначим через $\alpha \vdash \beta$. Если какая-то головка достигает символа $\#$, то она далее не двигается. Обработка n-ки слов заканчивается, когда все головки обозревают $\#$.

Процесс вычисления автомата M на x можно отобразить в виде конечного дерева вычислений, корень которого начальная конфигурация M на x $G_M(x)$, а одна ветвь - это вычисление /последовательность конфигураций/, которое заканчивается на принимающей или отвергающей конфигурации.

Через "$\vee$" обозначим операцию логического сложения, а через "$\wedge$" - операцию логического умножения. Определим рекурсивную процедуру разметки вершин /конфигураций/ дерева вычислений автомата M на x, используя функцию, определенную ниже f.

$$f(\alpha) = \begin{cases} \bigvee\limits_{\alpha \to \beta} f(\beta), & \text{если } \alpha \text{ } \exists\text{-конфигурация,} \\ \bigwedge\limits_{\alpha \to \beta} f(\beta), & \text{если } \alpha \text{ } \forall\text{-конфигурация,} \\ 1, & \text{если } \alpha \text{ принимающая конфигурация,} \\ 0, & \text{если } \alpha \text{ отвергающая конфигурация.} \end{cases}$$

Автомат M принимает /отвергает/ слово x, если $f(G_M(x)) = 1$ / $f(G_M(x)) = 0$/.

Будем говорить, что автомат M распознает язык L, если

1/ M принимает каждое $x \in L$,

2/ M отвергает каждое $x \in \overline{L}$.

Альтернирующий автомат M будем называть $\exists_k$-альтернирующим / $\forall_k$-альтернирующим/ автоматом, если начальное состояние является $\exists$-состоянием / $\forall$-состоянием/ и на любом вычислении кванторы $\exists$ и $\forall$, приписанные у конфигураций, меняются не более чем $k-1$ раз. Условимся, что $\exists_0$-/ $\forall_0$-/автоматы есть обычные детерминированные автоматы. Класс языков, который распознается двухленточными конечными $\exists_k$ / $\forall_k$ /-альтернирующими автоматами, обозначим через Σ_k / Π_k /.

Определим языки

$B_1 = \{ (x, yx) \mid x, y \in \{0,1\}^* \}$

$A_1 = \{ (0^{x_1} 0^{x_2} 1 0^{x_3} 1 \ldots 1 0^{x_n}) \mid \forall i \; x_i = x, \; n \in N \}$

$E_1 = \{ (0^{x_1} 0^{x_2} 1 0^{x_3} 1 \ldots 1 0^{x_n}) \mid \exists i \; x_i \geq x, \; n \in N \}$

$A_2 = \{ (x, u2v) \mid (x,u) \in A_1 \text{ и } (x,v) \in E_1 \}$

$E_2 = \{ (x, u2v) \mid (x,u) \in A_1 \text{ или } (x,v) \in E_1 \}$

$B_2 = \{ (x2y, u2v) \mid (x,u) \in A_1 \text{ или } (y,v) \in E_1 \}$

$F_2 = \{ (x2y, u2v) \mid (x,u) \in E_1 \text{ и } (y,v) \in A_1 \}$

В [2] доказаны следующие соотношения между классами Σ_1, Π_1, Π_0 /теоремы I-3/.

ТЕОРЕМА I. $B_1 \in (\Sigma_1 \cap \Pi_1) \setminus \Pi_0$.

ТЕОРЕМА 2. $E_1 \in \Sigma_1 \setminus \Pi_1$

ТЕОРЕМА 3. $A_1 \in \Pi_1 \setminus \Sigma_1$

Ниже доказываются аналогичные результаты для классов Σ_2, Π_2, Σ_1 и Π_1.

ТЕОРЕМА 4. I/ $B_2 \in (\Sigma_2 \cap \Pi_2) \setminus (\Pi_1 \cup \Sigma_1)$

2/ $F_2 \in (\Sigma_2 \cap \Pi_2) \setminus (\Pi_1 \cup \Sigma_1)$

ДОКАЗАТЕЛЬСТВО. I/ Опишем работу двухленточного конечного $\exists_2$-альтернирующего автомата M_1, распознающего язык B_2, на паре слов $(x2y, u2v)$. Автомат M_1 экзистенциально выбирает и проверяет следующие условия:

/4.I/ $(x,u) \in A_1$

/4.2/ $(y,v) \in E_1$

Поскольку $A_1 \in \Pi_1$ и $E_1 \in \Sigma_1$, то $B_2 \in \Sigma_2$.

Далее опишем работу двухленточного конечного $\forall_2$-альтернирующего автомата M_2, распознающего язык B_2, на паре слов $(x2y, u2v)$. В начале работает $\forall_1$-автомат, который производит проверку условия /4.I/. Если он попал в принимающее состояние, то заканчивает работу. Если автомат попал в отвергающее состояние, то начинает работать $\exists_1$-альтернирующий автомат, который проверяет условие /4.2/. Следовательно, $B_2 \in \Pi_2$.

Допустим от противного, что существует двухленточный конечный $\exists_1$-альтернирующий автомат M_3, распознающий язык B_2, и число его состояний равно s. Рассмотрим пару слов вида $(0^l 20, (0^l 1)^l 0^l 200)$ $(l > s+2)$. принадлежащую языку B_2. Легко видеть, что M_3 принимает и некоторую пару слов вида $(0^l 20, 0^l 1 \ldots 10^l 1\, 0^{l-k} 1\, 0^l \ldots 10^l 200)$ $(k > 1)$ не принадлежащую языку B_2. Противоречие.

Допустим от противного, что существует двухленточный конечный $\forall_1$-альтернирующий автомат M_4, распознающий язык B_2, и число его состояний равно s. Рассмотрим пару слов вида $(020^l, 02(0^{s+1}1)^l 0^{s+1})$ $(l > s+2)$ не принадлежащую

языку B_2. Легко видеть, что M_4 не принимает и некоторую пару слов вида $(02^{\ell}, 02(0^{s+1}1)^{*}0^{\ell+n}(10^{s+1})^{*})$ $(n \geq 1)$ из языка B_2. Противоречие.

2/ доказывается по схеме I/.

ТЕОРЕМА 5. $E_2 \in \Sigma_2 \setminus \Pi_2$

ДОКАЗАТЕЛЬСТВО. I/ Опишем работу требуемого двухленточного конечного $\exists_2$-альтернирующего автомата M_1, распознающего язык E_2, на паре слов $(x, u2v)$. Автомат M_1 экзистенциальным образом выбирает и проверяет следующие условия:

/5.I/ $(x,u) \in E_1$

/5.2/ $(x,v) \in A_1$

Поскольку $A_1 \in \Pi_1$ и $E_1 \in \Sigma_1$, то $E_2 \in \Sigma_2$.

2/ Допустим от противного, что существует двухленточный конечный $\forall_2$-альтернирующий автомат M_2, распознающий язык E_2, и число его состояний равно s. Рассмотрим пару слов вида $(x_0, y_0) = (0^{\ell}, (0^{s+1}1)^{\ell}0^{s+1}2(0^{\ell}1)^{(\ell+1)^2(s+2)^2}0^{\ell})$ $(\ell = s!)$ принадлежащую языку E_2. На любом вычислении M_2 на (x_0, y_0) назовем критической первую $\exists$-конфигурацию. Далее произведем раскраску дерева вычислений $\Delta_{M_2}(x_0, y_0)$ автомата M_2 на (x_0, y_0) следующим образом. Если соответствующая конфигурация такая, что головка на второй ленте находится левее символа "2", то покрасим вершину в красный цвет, в противном случае в синий цвет. Поскольку $(x_0, y_0) \in E_2$, то все критические конфигурации $\Delta_{M_2}(x_0, y_0)$ являются принимающими. Для этого достаточно существования вычисления из каждой критической конфигурации в некоторую принимающую конечную конфигурацию. Для каждой критической вершины зафиксируем одно такое вычисление таким образом, чтобы при одинаковых критических конфигурациях они совпали бы. Число различных критических конфигураций не больше чем $(\ell+1)^2 \cdot (s+2)^2$. Легко видеть, что на второй ленте найдется такой сегмент 0^{ℓ}, при чтении которого на всех зафиксированных вычислениях из критических конфигураций головка на первой ленте стоит на месте. Заменяем его на $0^{2\ell}$. Рассмотрим отвергающее дерево вычислений M_2 на паре слов вида $(x_1, y_1) = (0^{\ell}, (0^{s+1}1)^{\ell}0^{s+1}2(0^{\ell}1)^{*}0^{2\ell}(10^{\ell})^{*})$ не из языка E_2. Легко видеть, что все красные критические конфигурации дере-

ва $\Delta_{M_2}(x_0, y_0)$ являются принимающими. Следовательно, должна существовать отвергающая синяя критическая конфигурация. Зафиксируем вычисление из начальной конфигурации в эту синюю критическую конфигурацию. Очевидно, на второй ленте найдется сегмент $0^{3^{t+1}}$, при чтении которого на зафиксированном вычислении головка на первой ленте стоит на месте. Используя повторение состояний, этот сегмент удлиняем. Таким образом, мы построили пару слов вида $(0^{\ell}, (0^{3^{t+1}}1)^{*}\, 0^{\ell+\sim}(10^{3^{t+1}})^{*}\, 2\,(0^{t}1)^{*}\, 0^{2\ell}(10^{t})^{*})$ $(\sim \geqslant 0)$ из языка E_2, дерево вычислений автомата M_2 на котором содержит отвергающую критическую конфигурацию. Противоречие.

ТЕОРЕМА 6. $A_2 \in \Pi_2 \setminus \Sigma_2$

ДОКАЗАТЕЛЬСТВО. Утверждение $A_2 \in \Pi_2$ доказывается по схеме I/ теоремы 5.

Утверждение $A_2 \notin \Sigma_2$ доказывается по схеме 2/ теоремы 5 с той разницей, что в начале рассматривается пара слов вида $(0^{\ell}, (0^{t}1)^{\ell}\, 0^{t}\, 2\, (0^{3^{t+1}}1)^{3\cdot(\ell+2)^{3}}\, 0^{3^{t+1}})$ $(\ell \geqslant 1+2)$ не принадлежащая языку A_2, который по предположению распознается некоторым двухленточным конечным $\exists_2$-альтернирующим автоматом.

Теоремы I-6 дают соотношение между классами Π_0, Π_1, Σ_1, Π_2, Σ_2 в виде следующей картины.

ЛИТЕРАТУРА

I. Мучник А.А. Добавление переводчика к статьям "Об альтернировании, I, 2"// Кибернетический сборник.- М.: Мир, I983. - Вып.2. - С.I4I-I58.

2. Албертс М.Я. О возможностях различных типов альтернирующих автоматов// Кибернетика - в печати.

РАЗРЕШИМОСТЬ ПРОБЛЕМЫ ЭКВИВАЛЕНТНОСТИ ДЛЯ ГРАФИЧЕСКИХ ВЫРАЖЕНИЙ

А.Н.Бразма
ВЦ ЛГУ им.П.Стучки

Введение

В работе [I] введен формальный язык, удобный для описания общих закономерностей встречающийся при индуктивном синтезе программ. В данной работе введено некоторое обобщение этого языка (т.н. язык графических выражений) и исследована разрешимость проблемы эквивалентности.

Прежде чем дать точное определение языка графических выражений поясним его на содержательном примере. Рассмотрим алгоритм "пузырьковой" сортировки. Одним из наиболее простых методов изложения данного алгоритма является описание его работы на примере массива длины 4:

ВХОД: A(1:4)
ОПИСАНИЕ:

```
IF A(1) > A(2) THEN A(1) ↔ A(2);
IF A(2) > A(3) THEN A(2) ↔ A(3);
IF A(3) > A(4) THEN A(3) ↔ A(4);          (0.I)
IF A(1) > A(2) THEN A(1) ↔ A(2);
IF A(2) > A(3) THEN A(2) ↔ A(3);
IF A(1) > A(2) THEN A(1) ↔ A(2);
```

Используя язык графических выражений такой пример можно записать следующим образом:

ВХОД: A(1:4)
ОПИСАНИЕ: (0.2)

$$[[\text{IF } A(I) > A(I+1) \text{ THEN } A(I) \leftrightarrow A(I+1);]_{I=1,2}]_{2=3,1}$$

Обобщая данную запись для массива произвольной длины, получаем

ВХОД: $A(1:N)$

ОПИСАНИЕ: (0.3)

$$[[\text{IF } A(\underline{I}) > A(\underline{I+1}) \text{ THEN } A(\underline{I}) \leftrightarrow A(\underline{I+1});]_{I=1,2}\ ]_{\underline{J=N-1,1}} .$$

где N – произвольное натуральное число.
Под индуктивным синтезом здесь понимается восстановление выражения вида (0.3) по примеру вида (0.I).

В данной работе будет определен язык графических выражений, сформулировано понятие эквивалентности графических выражений и доказана алгоритмическая разрешимость проблемы эквивалентности.

I. Основные понятия

Пусть $\Sigma = \Sigma' \cup \mathbb{Z} \cup W \cup \mathbb{I}$ – некоторый алфавит, где Σ' – конечное множество произвольных символов, $\mathbb{Z}$ – множество целых чисел, $W = \{[,],=,\cdot,+\}$ и $\mathbb{I} = \{I, J, \ldots\}$ – т.н. множество параметров. Множества Σ', $\mathbb{Z}$, W и $\mathbb{I}$ попарно не пересекаются. Целые числа будем представлять в десятичной форме, при этом числа, состоящие из более чем одной цифры будем подчеркивать и принимать за один символ.

Пусть S – слово в алфавите Σ, в котором некоторые подслова вида I или $I+c$ ($I \in \mathbb{I}$, $c \in \mathbb{Z}$) могут быть подчеркнуты. Термом $\bar{S}$ назовём слово вида

$$[S]_{K=\alpha,\beta} , \qquad (I.I)$$

где $K \in \mathbb{I}$ и α, β – либо константы, либо слова вида $\underline{I}$ или $\underline{I+c}$ ($I \in \mathbb{I}$, $c \in \mathbb{Z}$). S назовём телом терма $\bar{S}$, α, β – его границами, а K – его собственным параметром. Терм $\bar{S}$ назовём корректным, если

его тело S по крайней мере в одном месте содержит подчеркнутое подслово $\underline{K}$ или $\underline{K+c}$ ($c \in \mathbb{Z}$). В дальнейшем, если не оговорено противное, под **термами** будем понимать именно корректные термы. Через $S(I_1|a_1, \dots, I_n|a_n)$, где $a_i \in \mathbb{Z}$, $I_i \in \mathbb{I}$ ($i \in \{1, \dots, n\}$), обозначим слово, которое получается из S , если во всех вхождениях выражений вида $\underline{I_i}$ или $\underline{I_i+c}$ вместо I_i подставить a_i и вычислить значения подчеркнутых выражений. Например, если $S = A\underline{I}B(\underline{I+3})$, то $S(I|3) = A3B(6)$.

Будем говорить, что границы α, β терма **фиксированы**, если $\alpha, \beta \in \mathbb{Z}$. Определим **значение** $val(\bar{S})$ для термов

$$\bar{S} = [S]_{\underline{K}=c_1, c_2} \qquad (I.2)$$

с фиксированными границами:

$$val(\bar{S}) = S(K|c_1)S(K|c_1+\delta)S(K|c_1+2\delta) \dots S(K|c_2), \qquad (I.3)$$

где $\delta = 1$, если $c_1 \le c_2$, и $\delta = -1$, если $c_1 > c_2$. δ называется **направлением** терма $\bar{S}$. Например, если $\bar{S} = [A\underline{I}B(\underline{I+3})]_{\underline{I}=4,6}$, то

$$val(\bar{S}) = A1B(4)A2B(5)A3B(6)A4B(7)A5B(8) .$$

Определим теперь **графическое выражение**. Пусть X - некоторое слово в алфавите $\Sigma \cup \mathbb{Z} \cup \mathbb{I}$ и $I_1, \dots, I_n \in \mathbb{I}$ - параметры встречающиеся в X . Пусть все I_i ($i \in \{1, \dots, n\}$) входят в X в виде $\underline{I_i}$ или $\underline{I_i+c}$ ($c \in \mathbb{Z}$). Тогда такое слово X назовём **простым графическим выражением**.

Определим **графическое выражение со свободными параметрами** индуктивно.

1. Простое графическое выражение является графическим выражением; все параметры, входящие в это выражение, являются свободными.
2. Если X и Y - графические выражения с множеством свободных параметров $\mathbb{I}_X$ и $\mathbb{I}_Y$, то слово XY является графическим выражением (X и Y будем называть подвыражениями выражения XY) с множеством свободных параметров $\mathbb{I}_X \cup \mathbb{I}_Y$.
3. Пусть X - графическое выражение с множеством свободных параметров $\mathbb{I}_X$. Тогда терм

$$[X]_{I=\alpha,\beta}$$

является графическим выражением с множеством параметров

$$(\mathbb{I}_X \setminus \{I\}) \cup \mathbb{I}_\alpha \cup \mathbb{I}_\beta ,$$

где $\begin{cases} \{I_1\} & \text{, если } \alpha = \underline{I_1} \text{ или } \alpha = \underline{I_1 + c_1} \\ \varnothing & \text{, если } \alpha = c_1 \end{cases}$

и $\begin{cases} \{I_2\} & \text{, если } \beta = \underline{I_2} \text{ или } \beta = \underline{I_2 + c_2} \\ \varnothing & \text{, если } \beta = c_2 \end{cases}$,

если $I \neq I_1$ и $I \neq I_2$.

4. Других графических выражений нет.

Выражение будем считать **корректным**, если все входящие в него термы являются корректными и имеют различные собственные параметры. В дальнейшем, если не оговорено противное, под выражениями будем понимать именно корректные выражения. Если множество свободных параметров графического выражения состоит из одного элемента (пусть это будет N), то такое выражение будем называть унарным.

Свободные параметры слова X назовём **внешними** для терма $[X]_{I=\alpha,\beta}$.

Определим **глубину** $depth(E)$ **выражения** E индуктивно:

1. Если E – простое выражение, то $depth(E)=0$.

2. Если $E = E_1 E_2$, то $depth(E) = \max(depth(E_1), depth(E_2))$.

3. Если $E = [S]_{I=\alpha,\beta}$, то $depth(E) = depth(S) + 1$.

Под **глубиной терма** $\bar{S}$ понимается глубина выражения $E = \bar{S}$. Термы глубины 1 мы будем называть **простыми термами**, а термы большей глубины – **сложными термами**. Терм $\bar{S}$ назовём **внешним** в выражении E, если он не содержится в другом терме выражения E.

Если $I_1, \dots, I_n$ – некоторые свободные параметры выражения E, то часто будем писать $E(I_1, \dots, I_n)$. В таком случае вместо $E(I_1|a_1, \dots, I_n|a_n)$, где $a_1, \dots, a_n \in \mathbb{Z}$ будем писать просто $E(a_1, \dots, a_n)$. Так, например, значение терма $[S(I)]_{I=c_1,c_2}$ можно записать

$$val([S(I)]_{I=c_1,c_2}) = S(c_1)\,S(c_1+\delta)\,\dots\,S(c_2),$$

где δ – направление терма $[S(I)]_{I=c_1,c_2}$.

Пусть $W(I_1, \dots, I_n)$ – графическое выражение, где $I_1, \dots, I_n$ – свободные параметры данного выражения. **Конкретизацией** выражения $W(I_1, \dots, I_n)$ назовём графическое выражение $W(c_1, \dots, c_n)$, где $c_i \in \mathbb{Z}$ $(i \in \{1, \dots, n\})$. Если $I_1, \dots, I_n$ – все свободные параметры выражения W, то полученное выражение $W(c_1, \dots, c_n)$ назовём **конкретным**. Очевидно, в конкретном выражении границы всех внешних термов фиксированы.

Первую развёртку $VAL^1(W(c_1, \dots, c_n))$ конкретного выражения мы получаем заменяя в $W(c_1, \dots, c_n)$ все внешние термы их значениями. Например,

$$VAL^1([[\underline{I}]_{\underline{I}=1,2}]_{2=3,1}) = [\underline{I}]_{\underline{I}=1,3}\,[\underline{I}]_{\underline{I}=1,2}\,[\underline{I}]_{\underline{I}=1,1}.$$

З н а ч е н и е $VAL(W)$ конкретного выражения W определяется индуктивно:

1. Если $depth(W) = 0$, то $VAL(W) = W$
2. Если $depth(W) > 0$, то

$$VAL(W) = VAL(VAL^1(W)) .$$

Например, $VAL([[\underline{I}]_{I=1,2}]_{2=3,4}) = 123121$.
Часто вместо $VAL(W(c_1,\dots,c_n))$ будем писать $VAL(W; c_1,\dots,c_n)$.

Будем говорить, что два графические выражения $E_1(N_1,\dots,N_n)$ и $E_2(M_1,\dots,M_n)$ (где $N_1,\dots,N_n$ и $M_1,\dots,M_\ell$ – все свободные параметры E_1 и E_2) э к в и в а л е н т н ы ($E_1 \cong E_2$), если для каждого набора натуральных чисел $a_1,\dots,a_n$ существуют такие $b_1,\dots,b_\ell$, и для каждого набора натуральных чисел $b_1,\dots,b_\ell$ существуют такие $a_1,\dots,a_n$, что

$$VAL(E_1; a_1,\dots,a_n) = VAL(E_2; b_1,\dots,b_\ell) .$$

Будем говорить, что E_1 и E_2 а с и м п т о т и ч е с к и э к в и в а л е н т н ы, если существует такое $c \in \mathbb{N}$, что для каждого набора $a_1,\dots,a_n$ такого, что $\forall i : a_i > c$ и $\forall i,j \ |a_i - a_j| > c$, существуют такие натуральные $b_1,\dots,b_\ell$ и для каждого набора $b_1,\dots,b_\ell$ такого, что $\forall i : b_i > c$ и $\forall i,j : |b_i - b_j| > c$ существуют такие натуральные $a_1,\dots,a_n$, что

$$VAL(E_1; a_1,\dots,a_n) = VAL(E_2; b_1,\dots,b_\ell) .$$

В данной работе доказаны следующие теоремы.

ТЕОРЕМА I. Проблема асимптотической эквивалентности для графических выражений алгоритмически разрешима.

Из доказательства теоремы I следует также

ТЕОРЕМА 2. Проблема эквивалентности для унарных графических выражений алгоритмически разрешима.

2. Идея доказательства

Доказательство разрешимости проблемы асимптотической эквивалентности основано на следующем факте: два выражения асимптотически эквивалентны тогда и только тогда, если некоторое достаточно большое значение одного выражения совпадает с некоторым значением второго выражения. Поясним это более подробно для унарных выражений. Пусть даны два унарных выражения E и F, и пусть X равно $VAL(E,a)$ для некоторого достаточно большого $a \in \mathbb{N}$. Если $E \sim F$, то существует такое $b \in \mathbb{N}$, что $VAL(F,b) = VAL(E,a) = X$. Легко видеть, что b можно эффективно найти. Доказательство разрешимости асимптотической эквивалентности основано на факте, что существование такого $b \in \mathbb{N}$ является и достаточным условием того, что $E \sim F$.

Для того, чтобы описать идею доказательства данного утверждения введем несколько понятий.

В дальнейшем нам будет удобно графические выражения представлять в несколько другой форме. Простые термы (т.е. термы глубины один) будем записывать с помощью т.н. многоточечных термов. Так вместо $[T(I)]_{I=\alpha_1,\alpha_2}$ будем писать $\langle T(\alpha_1) \dots T(\alpha_2) \rangle$. Например, терм $[A\underline{I}B(\underline{I+2})]_{I=1,N}$ записывается как $\langle A1\,B(3) \dots AN\,B(N+2) \rangle$, а выражение $[[\underline{I}]_{I=1,2}]_{I=1,1}$ - как $[\langle 1 \dots \underline{1} \rangle]_{I=N,1}$.

Наряду со значением $VAL(E,a)$ выражения E, будем рассматривать и **развёртку выражения до глубины** k — $VAL_k(E,a)$, которая отличается от $VAL(E,a)$ тем, что термы глубины k или меньше не заменяются их значениями. Например, если $E = [A[B\langle 1 \dots \underline{I} \rangle]_{I=1,2}]_{I=1,N}$, то $VAL_1(E,3) = A[B\langle 1 \dots \underline{I} \rangle]_{I=1,1}\ A[B\langle 1 \dots I \rangle]_{I=1,2}\ A[B\langle 1 \dots \underline{I} \rangle]_{I=1,3}$, а $VAL_4(E,3) = AB\langle 1 \dots 1 \rangle AB\langle 1 \dots 1 \rangle B\langle 1 \dots 2 \rangle AB\langle 1 \dots 1 \rangle B\langle 1 \dots 2 \rangle B\langle 1 \dots 3 \rangle$.

Идея доказательства следующая. По последовательности X с помощью некоторого алгоритма синтеза S постепенно будем сторить последовательности $X_1, X_2, \ldots, X_n$. С другой стороны, с помощью некоторого алгоритма преобразования Π_1 будем преобразовывать выражения E и F в $E_1, E_2, \ldots, E_n$ ($E_1 \sim E_2 \sim \ldots \sim E_n$) и $F_1, F_2, \ldots, F_n$ ($F_1 \sim F_2 \sim \ldots \sim F_n$) соответственно так, чтобы для всех $i = 1, \ldots, n-1$ имело место: $VAL_i(E_i, a) = X_i$ и $VAL_i(F_i, b) = X_i$, а в конце $VAL_n(E_n, a) = E_n(a)$ и $VAL_n(F_n, b) = F_n(b)$, откуда $E_n(a) = F_n(b)$. Из этого будет следовать $E \sim F$.

Опише теперь эту идею более подробно.

Рассмотрим подпоследовательность Y последовательности X. Назовём Y (ρ, t) - **регулярной** ($\rho, t \in \mathbb{N}$), если существует такой простой терм $\bar{T} = [T]_{1=1,t}$, с $|T| = \rho$ (где $|T|$ - число символов в последовательности T), что $Y = val(\bar{T})$. ρ будем называть **периодом**, а t - **фактором**. Подпоследовательность $T(n)$, $n \in \{1, \ldots, t\}$ последовательности Y назовём **n-телом** Y. Иногда n-тела регулярных последовательностей будем просто называть **телами**.

Пусть $X = \alpha_1 \ldots \alpha_{k-\rho} \ldots \alpha_k \alpha_{k+1} \ldots \alpha_l \alpha_{l+1} \ldots \alpha_n$ ($\alpha_i \in \Sigma$) и пусть $Y = \alpha_k \ldots \alpha_l$ - (ρ, t)-регулярная подпоследовательность ($\rho, t \in \mathbb{N}$). Будем говорить, что Y можно **сдвинуть вправо**, если $\alpha_{k+1} \ldots \alpha_{l+1}$ также (ρ, t) - регулярная подпоследовательность. Будем говорить, что Y можно **расширить влево**, если $\alpha_{k-\rho} \ldots \alpha_k$ - $(\rho, t+1)$ - регулярная подпоследовательность.

Пусть Y (ρ, t)- регулярная подпоследовательность последовательности X и пусть $m \in \mathbb{N}$. Будем говорить, что подпоследовательность Y, m - **покрывает** подпоследовательность Y', если хотя бы одно из n - тел ($n \in \mathbb{N}$) подпоследовательности Y

содержит m или более тел подпоследовательности Y'.

В доказательстве важное значение будет иметь следующая лемма о регулярных последовательностях.

ЛЕММА I. Пусть подпоследовательность X - (ρ_1, t_1)-регулярное, а подпоследовательность Y - (ρ_2, t_2)-регулярное. Пусть X и Y содержат общую подпоследовательность Z. Тогда, если $|Z| \geq \max(\rho_1, \rho_2) + 4 \cdot \min(\rho_1, \rho_2)$, то $\rho_1 = \rho_2$.

ДОКАЗАТЕЛЬСТВО. Пусть $\rho_1 \leq \rho_2$ и пусть общая подпоследовательность $Z = \alpha_1 \alpha_2 \dots \alpha_n$. Согласно условию леммы $n \geq \rho_2 + 4\rho_1$.

Сначала опишем некоторый алгоритм, который будет работать на слове Z и в итоге выдаёт определённое число S_0. Затем докажем, что, во-первых, если Z регулярно с периодами ρ_1 и ρ_2, то $S_0 = 0$ и, во-вторых, если $S_0 = 0$, то $\rho_1 = \rho_2$. В алгоритме мы будем использовать одну переменную S и будем считать, что в начале она имеет значение 0. Ещё в алгоритме мы будем использовать маркер, который будем передвигать по слову Z.

Так как слово Z регулярное с периодом ρ_1, то в нем существует α_ρ, $1 \leq \rho < \rho_1$, такое, что

$$\alpha_{\rho+\rho_1} = \alpha_\rho + 1 \quad \text{или} \quad \alpha_{\rho+\rho_1} = \alpha_\rho - 1.$$

Установим в начале маркер на α_ρ. Далее алгоритм будет работать следующим образом. Если в какой-то момент маркер стоит на символе α_j и $j \leq \rho_2$, то переместим маркер на $\alpha_{j+\rho_1}$. Если $j > \rho_2$, то переместим маркер на $\alpha_{j-\rho_2}$. При каждом таком передвижении маркера переменной S мы прибавим, соответственно, $\alpha_{j+\rho_1} - \alpha_j$ или $\alpha_{j-\rho_2} - \alpha_j$ (ниже будет доказано, что α_j, $\alpha_{j+\rho_1}$ и $\alpha_{j-\rho_2}$ всегда являются числами и поэтому упомянутые разницы имеют смысл). После $\rho_1 + \rho_2$ таких шагов алгоритм закончит работу. Значение переменной S в этот момент обозначим через S_0.

Легко видеть, что если маркер будет находиться на каком-то символе α_j, то имеет место $\alpha_j = \alpha_p + S$. Также легко доказать, что после $p_1 + p_2$ передвижений маркера (т.е. по окончании работы алгоритма будет сделано p_2 передвижений вправо и p_1 передвижений влево и маркер снова будет на α_p. Поэтому в момент окончания работы $\alpha_p = \alpha_p + S$. Отсюда получаем, что $S_0 = S = 0$.

Теперь докажем, что если $S_0 = 0$, то $p_1 = p_2$. Сначала заметим, что:

$$|\alpha_{j+p_1} - \alpha_j| \leq 1 \qquad (2.1)$$

и

$$|\alpha_{j+p_2} - \alpha_j| \leq 1 \qquad (2.2)$$

Докажем, что

$$\text{из } \alpha_{j+p_2} - \alpha_j = 1 \text{ следует } \alpha_{j+p_1+p_2} - \alpha_{j+p_2} = 1 \qquad (2.3)$$

а из $\alpha_{j+p_1} - \alpha_j = -1$ следует $\alpha_{j+p_1+p_2} - \alpha_{j+p_2} = -1$.

Перед тем как доказывать (2.3) покажем, что из него следует $p_1 = p_2$.

Для этого допустим, что в начале мы выбрали α_p, для которого $\alpha_{p+p_1} - \alpha_p = 1$ (соответственно, $\alpha_{p+p_1} - \alpha_p = -1$). Из (2.3) следует, что для любого α_j, на котором в какой-то момент находится маркер, $\alpha_{j+p_2} - \alpha_j = 1$ (соответственно, $\alpha_{j+p_2} - \alpha_j = -1$). Отсюда в свою очередь следует, что всякий раз, когда мы передвигаем маркер влево, мы к S прибавляем $+1$ (соответственно, -1). Так как согласно вышесказанному, по окончании работы алгоритма маркер вправо будет передвинут всего p_2 раз, то к S из-за правых переходов будет прибавлено p_2

(соответственно, $-\rho_2$). С другой стороны, так как маркер влево будет передвинут ρ_1 раз, то учитывая (2.2), мы получим, что к S из-за левых переходов будет прибавлено число не меньше, чем $-\rho_1$, и не больше, чем ρ_1. Таким образом, $S \geq \rho_2 - \rho_1$ (соответственно, $S \leq -\rho_2 + \rho_1$). Но так как $S_0 = 0$, то отсюда следует, что $\rho_1 \geq \rho_2$. Согласно начальному предположению $\rho_2 \geq \rho_1$. Отсюда получаем, что $\rho_1 = \rho_2$.

Таким образом, для завершения доказательства остаётся доказать утверждения (2.3). Докажем первое из них.

Предположим противное, т.е. что $\alpha_{j+\rho_1} - \alpha_j = 1$, а $\alpha_{j+\rho_2+\rho_1} - \alpha_{j+\rho_2} \neq 1$.

Из (2.1) следует, что возможны три случая:

1. $\alpha_{j+\rho_1+\rho_1} \notin Z$ (т.е. $\alpha_{j+\rho_2+\rho_1} \in \Sigma'$)
2. $\alpha_{j+\rho_2+\rho_1} - \alpha_{j+\rho_2} = 0$
3. $\alpha_{j+\rho_2+\rho_1} - \alpha_{j+\rho_2} = -1$.

Случай I невозможен, так как легко видеть, что в этом случае получаем $\alpha_{j+\rho_1} \in \Sigma'$ и, следовательно, $\alpha_j \in \Sigma'$, но это противоречит $\alpha_{j+\rho_1} - \alpha_j = 1$.

Покажем, что случаи 2 и 3 также невозможны. Для каждого из них возможны три подслучая

а) $\alpha_{j+\rho_1} = \alpha_j + 1$

б) $\alpha_{j+\rho_1} = \alpha_j$

с) $\alpha_{j+\rho_1} = \alpha_j - 1$.

Таким образом, всего мы имеем 6 случаев, которые показаны в следующей таблице:

Таблица I

	α_j	$\alpha_{j+1} = \alpha_{j+\rho_1}$	$\alpha_{j+\rho_1}$	$\alpha_{j+2} = \alpha_{j+\rho_2}$	$\alpha_{j+\rho_2+\rho_1}$	$\alpha_{j+3} = \alpha_{j+\rho_3}$	$\alpha_{j+\rho_3+2\rho_1}$	$\alpha_{j+\rho_3+3\rho_1}$
2a			α_j+1		α_j+1		α_j+1	α_j+1
3a			α_j+1		α_j		α_j-1	α_j-2
2b			α_j		α_j		α_j	α_j
3b			α_j		α_j-1		α_j-2	α_j-3
2c			α_j-1		α_j-1		α_j-1	α_j-1
3c			α_j-1		α_j-2		α_j-3	α_j-4

Рассмотрим разницу $|\alpha_{j+3p_1+p_2} - \alpha_{j+3p_1}|$. Из таблицы видно, что в любом из 6 случаев эта разница строго больше I. Однако это противоречит (2.2). Таким образом, случаи 2 и 3 также невозможны.

Если теперь ещё раз просмотреть приведённые выше рассуждения, то легко убедиться, что для их справедливости достаточно, чтобы $|Z| = p_2 + 4p_1$.

Лемма доказана. Из леммы I легко получаем

СЛЕДСТВИЕ I. Пусть зафиксировано некоторое число $c_0 \in \mathbb{N}$ такое, что $c_0 > 5$. Пусть X (p_1, t_1)-регулярная последовательность с $t_1 \geq c_0$, содержащая (p_2, t_2) - регулярную последовательность Y с $t_2 \geq c_0$ такую, что $p_1 \neq p_2$. Тогда последовательность (c_0-5) покрывает последовательность Y.

Теперь мы можем описать алгоритм синтеза S . Алгоритм зависит от некоторой константы $c_0 \in \mathbb{N}$ и работает следующим образом. Алгоритм берет первую, начиная с левой стороны данной последовательности X , регулярную подпоследовательность Y такую, чтобы

I) фактор t подпоследовательности Y был бы больше или равен c_0 ,

2) не существовало бы другой регулярной подпоследовательности Y' с фактором $t' \geq c_0$ такой, что подпоследовательность Y (c_0-5) покрывает подпоследовательность Y' .

Затем алгоритм максимально расширяет и смещает вправо найденную подпоследовательность Y так, чтобы не нарушилось (2). После этого алгоритм заменяет эту подпоследовательность на соответствующий терм $T = [T_0]_{1 \cdot 4,5}$ (т.е. такой, что $val(T) = Y$), и продолжает обрабатывать таким же образом оставшуюся часть последовательности X за термом T . Результат работы алгоритма синтеза S на последовательность X обозначим через $S(X, c_0)$. Например, если

$X = ABA1A2A3A4A56789$, $(A, B \in \Sigma')$

то

$$S(X,3) = AB\langle A1 \dots A4\rangle A\langle 5 \dots 9\rangle .$$

Пусть теперь $E_0 = E$, $F_0 = F$ и $X_0 = VAL(E_0, a) = VAL(F_0, b)$. Выберем c_0 некоторым образом в зависимости от E_0 и F_0 . Пусть

$$X_1 = S(X_0, c_0) .$$

Преобразуем E_0 в E_1 ($E_1 \sim E_0$) и F_0 в F_1 ($F_1 \sim F_0$) так, чтобы имело место

$$VAL_1(E_1, a) = VAL_1(F_1, b) = X_1 . \qquad (2.4)$$

Алгоритм Π_1 , осуществляющий такое преобразование, является наиболее сложной частью доказательства и ему в основном и посвящены следующие главы. C_0 выбрано так, чтобы такие преобразования были бы возможны.

Таким образом,(2.4) можно переписать следующим образом:

$$\begin{aligned} VAL_1(\Pi_1(E_0), a) &= S(X_0, c_0) \\ VAL_1(\Pi_1(F_0), b) &= S(X_0, c_0) . \end{aligned} \qquad (2.5)$$

Последовательность $X_1 = S(X_0, c_0)$ содержит многоточечные термы глубины один. Будем временно их считать специальными символами и игнорировать их семантику. Тогда к X_1 опять можно применить алгоритм синтеза S. Также поступим и с термами глубины один выражений E_1 и F_1 и применим ещё раз алгоритм Π_1 . Получим

$$\begin{aligned} VAL_2(\Pi_1(E_1), a) &= S(X_1, c_1) \\ VAL_2(\Pi_1(F_1), b) &= S(X_1, c_1) , \end{aligned} \qquad (2.6)$$

где c_i выбрано в зависимости от E_i и F_i.

Продолжая таким образом, получаем индукцией

$$VAL_k(\Pi_1(E_{k-1}), a) = S(X_{k-1}, c_{k-1})$$
$$VAL_k(\Pi_1(F_{k-1}), b) = S(X_{k-1}, c_{k-1}), \qquad (2.7)$$

где $E_i = \Pi_1(E_{i-1})$, $F_i = \Pi_1(F_{i-1})$,

$X_i = S(X_{i-1}, c_{i-1})$ и c_i выбрано в зависимости от E_i и F_i $(i \in \{1, \dots, k-1\})$.

Фактически это означает, что из

$$VAL_k(E_k, c) = VAL_k(F_k, b) \qquad (2.8)$$

следует

$$VAL_{k-1}(\Pi_1(E_k), a) = VAL_{k-1}(\Pi_1(F_k), b). \qquad (2.9)$$

Легко видеть, что если всегда $c_i > 1$, то существует такое n, что имеет место:

$$S(X_n, c_n) = X_{n+1} = X_n,$$

т.е. алгоритм S на X_n уже ничего не просинтезирует. Из определения преобразований выражений будет легко вытекать, что для этого n имеет место

$$VAL_{n+1}(E_{n+1}, a) = E_{n+1}(a) \qquad (2.10)$$

и

$$VAL_{n+1}(F_{n+1}, b) = F_{n+1}(b). \qquad (2.11)$$

Следовательно, так как

$$VAL_{n+1}(E_{n+1}, c) = VAL_{n+1}(F_{n+1}, b), \qquad (2.12)$$

то

$$E_{n+1}(a) = F_{n+1}(b) \qquad (2.13)$$

Имеет место следующая

ЛЕММА 2. Если $E(a) = F(b)$ для достаточно больших $a \in \mathbb{N}$ и $b \in \mathbb{N}$, то $E(N) = F(N+c)$, где $c = b - a$ (лемма может быть обобщена и на случай многих параметров).

Из леммы следует $E_{n+1} \sim F_{n+1}$. Так как по индукции для каждого $k \le n+1$: $E_k \sim E$ и $F_k \sim F$, то $E \sim E_{n+1} \sim F_{n+1} \sim F$.

Этим показано, что если для достаточно больших a и b существует такое X, что $X = VAL(E,a) = VAL(F,b)$ то $E \sim F$.

Таким образом, остаётся определить алгоритм Π_1 и доказать, что имеет место

$$VAL_k(\Pi_1(E_{k-1}), a) = S(VAL(E_{k-1}, a), c_{k-1}) . \qquad (2.14)$$

Так как в E_{k-1} термы глубины $k-1$ считаются символами, то достаточно доказать, что имеет место

$$VAL_1(\Pi_1(E), a) = S(VAL(E,a), c_0) \qquad (2.15)$$

для произвольного E.

В дальнейшем будем считать, что c_0 зафиксировано и будем его опускать.

Ниже мы определим ещё один алгоритм преобразования Π_2 для развёрток глубины один - $VAL_1(E,a)$, и докажем, что

1) $VAL_1(\Pi_1(E), c) = \Pi_2(VAL_1(E,a))$

и

2) $\Pi_2(VAL_1(E,c)) = S(VAL(E,a))$.

Из чего и следует равенство (2.15).

Опишем идею алгоритмов Π_1 и Π_2. Алгоритмы будут основаны на некоторых правилах преобразования. Поясним их на примерах.

PI. Правило сдвига:

$$O\langle A_1 \dots A_{\underline{K}}\rangle A \rightarrow OA\langle 1A \dots {}_{\underline{K}}A\rangle$$

P2. Правило расширения:

$$OA\langle 1A \dots {}_{\underline{K}}A\rangle \rightarrow \langle OA \dots {}_{\underline{K}}A\rangle$$

P3. Правило объединения:

$$\langle A_1 \dots A_{\underline{K-1}}\rangle\langle A_{\underline{K}} \dots A_{\underline{L}}\rangle \rightarrow \langle A_1 \dots A_{\underline{L}}\rangle$$

P4. Правило синтеза

$$A_1A_2A_3A_4A_5A_6A_7 \rightarrow \langle A_1 \dots A_7\rangle$$

P5. Правило развертки коротких термов

$$\langle A_1 \dots A_3\rangle \rightarrow A_1A_2A_3$$

Равенство $VAL_1(\Pi_1(E),a) = \Pi_2(VAL_1(E,a))$ означает, что упомянутые правила PI-P5 работают на $\Pi_1(E)$ и $VAL_1(E,a)$ в некотором смысле одинаково. Например, если

$$E = [\langle 1 \dots \underbrace{I}\rangle \underbrace{I+1}]_{I=5,N}$$

$$VAL_1(E,8) = \langle 1\dots 5\rangle 6\langle 1\dots 6\rangle 7\langle 1\dots 7\rangle 8\langle 1\dots 8\rangle 9 ,$$

то после применения сдвига вправо и расширения влево в E подвыражения $\langle 1\dots \underline{I}\rangle \underline{I+1}$ и в $VAL_1(E,8)$ соответственно $\langle 1\dots 5\rangle 6$, $\langle 1\dots 6\rangle 7$, $\langle 1\dots 7\rangle 8$ и $\langle 1\dots 8\rangle 9$ получаем соответственно,

$$\Pi_1(E) = [\langle 1 \dots \underline{I+1}\rangle]_{I=5,N} \quad \text{и}$$

$$\Pi_2(VAL_1(E,8)) = \langle 1\dots 6\rangle\langle 1\dots 7\rangle\langle 1\dots 8\rangle\langle 1\dots 9\rangle$$

В данном случае

$$VAL_1(\Pi_1(E),8) = \Pi_2(VAL_1(E,8)) .$$

Но, к сожалению, не всегда это верно. Покажем на примерах основные трудности.

Пример I.

Пусть

$$E_1 = [1\langle \underline{I} \cdots \underline{N} \rangle]_{\underline{I}=1,\underline{N}}$$

$$VAL_1(E_1, 6) = 1\langle 1 \cdots 6\rangle\, 1\langle 2 \cdots 6\rangle\, 1\langle 3 \cdots 6\rangle\, 1\langle 4 \cdots 6\rangle\, 1\langle 5 \cdots 6\rangle\, 1\langle 6 \cdots 6\rangle .$$

Пусть $c_0 = 1$. Легко видеть, что применить какое-либо из правил к E невозможно, но к $VAL_1(E,6)$ можно применить следующие правила:

I) к $1\langle 2 \cdots 6\rangle$ - правило расширения,

2) к $\langle 6 \cdots 6\rangle$ - правило развёртки.

Только в подслове $1\langle 3 \cdots 6\rangle\, 1\langle 4 \cdots 6\rangle\, 1\langle 5 \cdots 6\rangle$ правила "работают" также как в E_1 , т.е. правила не применимы. Однако E_1 можно преобразовать в асимптотически эквивалентную E'_1 , такую, чтобы равенство имело место:

$$E'_1 = 1\langle 1 \cdots 6\rangle 1\langle 2 \cdots 6\rangle\, [1\langle \underline{I} \cdots \underline{N}\rangle]_{\underline{I}=3,\underline{N}-1}\, 1\langle 6 \cdots 6\rangle .$$

Легко убедиться, что $E' \sim E$ и что

$$VAL_1(\Pi_1(E'_1), 6) = \Pi_1(VAL_1(E'_1, 6)) .$$

К сожалению, раскрытие не всегда "спасает". Рассмотрим второй пример.

Пример 2.

$$E_2 = [[\underline{I}\langle \underline{2} \cdots \underline{N+3}\rangle]_{\underline{2}=0,\underline{N-1}}]_{\underline{I}=1,\underline{N-1}}$$

$$VAL_1(E_2, 4) = 1\langle 1 \cdots 8\rangle 1\langle 2 \cdots 8\rangle 1\langle 3 \cdots 8\rangle 1\langle 4 \cdots 8\rangle$$
$$2\langle 1 \cdots 8\rangle 2\langle 2 \cdots 8\rangle 2\langle 3 \cdots 8\rangle 2\langle 4 \cdots 8\rangle$$
$$3\langle 1 \cdots 8\rangle 3\langle 2 \cdots 8\rangle 3\langle 3 \cdots 8\rangle 3\langle 4 \cdots 8\rangle$$
$$4\langle 1 \cdots 8\rangle 4\langle 2 \cdots 8\rangle 4\langle 3 \cdots 8\rangle 4\langle 4 \cdots 8\rangle$$

В $VAL_4(E_2,4)$ к $1\langle 2\cdots 8\rangle$, $2\langle 3\cdots 8\rangle$ и $3\langle 4\cdots 8\rangle$ можно применить правила расширения. Легко видеть, что раскрытие в этом примере не поможет. Чтобы можно было "спасти ситуацию" в этом случае сначала разобьем внешний терм E_2 на два терма. Получим

$$E_2' = [[I\langle J_1 \cdots \underline{N+3}\rangle]_{J_1=I+1,\underline{I}}\,[I\langle J_1 \cdots \underline{N+3}\rangle]_{J_1=\underline{I+1},\underline{N}}]_{\underline{I}=1,\underline{N}}$$

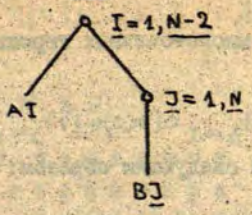
Теперь после раскрытия терма $[I\langle J_1 \cdots \underline{N+3}\rangle]_{J_1=\underline{I+1},\underline{N}}$ к полученному выражению можно применить те же правила, что и к развертке.

Таким образом, перед работой преобразователя Π_4 мы должны предварительно преобразовать выражение E в выражение E' такое, чтобы достаточно близко стоящие разные параметры имели в развертке $VAL_4(E,a)$ достаточно отличающиеся значения. Более точно это будет определено в следующем параграфе, где и будет дан алгоритм такого преобразования.

3. Предварительные преобразования выражения

В данном параграфе мы опишем предварительные преобразования выражения, которые необходимы для того, чтобы правила преобразования, упомянутые в 2, одинаково преобразовали выражение и его развертку до глубины I.

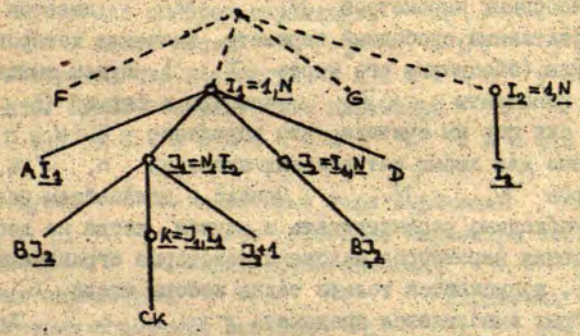
Нам будет удобно представить выражения в виде дерева. Каждому терму T_i будет соответствовать вершина дерева S_i, к которой приписан собственный параметр терма и его границы. Если терм T_j входит в терм T_i, то соответствующая вершина S_j является потолком вершины S_i. Простым выражениям соответствуют листья дерева. Например, выражению $E = [A\underline{I}[B\underline{J}]_{J=I+1,\underline{N}}]_{\underline{I}=1,N-1}$ соответствует следующее дерево:



Если выражение состоит из конкатенации нескольких внешних термов, то его можно представить как упорядоченный лес. Иногда нам будет удобно объединить этот лес тоже в дерево с помощью ребер другого вида. Например, выражение

$$E = [AI_1 [BI_2 [CK]_{K=J_1,I_2} \; J_1+1]_{J_1=N,I_1} [BJ_3]_{J_3=I_1,N} D]_{I_1=1,N} \; G[I_2]_{I_2=1,N}$$

представляется в виде дерева следующим образом

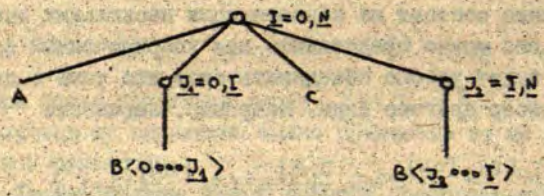


Отметим, что глубина $depth(\bar{T})$ терма $\bar{T}$ тогда равна глубине соответствующего поддерева. Расстояние от вершины s, соответствующей терму T, до корня дерева (соответствующего внешнему терму), назовём **включенностью терма** T и обозначим через $incl(\bar{T})$. Можно также представить в виде дерева выражение до определенной глубины, а как листья дерева оставить остальное выражение. Далее нам будет удобно представлять в виде дерева все составные термы, а простые

термы оставить в виде многоточечных термов. Например, выражение

$$E = [A[B\langle 0 \cdots J_1 \rangle]_{J_1=0,I} C[D\langle J_2 \cdots I \rangle]_{J_2=I,N}]_{I=0,N}$$

в виде дерева выглядит следующим образом:



Такое дерево назовём **деревом выражения.**

В этом параграфе нам также будет удобно считать, что кроме свободных параметров $N_1, \dots, N_k$ имеется ещё один – фиктивный свободный параметр, значение которого равно нулю (обозначим его через σ). Будем считать, что все константы $c \in \mathbb{Z}$ записаны в форме $\sigma + c$.

До сих пор мы считали, что выражение $E(N_1, \dots, N_k)$ определено для любых натуральных значений $a_1, \dots, a_k$ параметров $N_1, \dots, N_k$. Однако в дальнейшем нам будет необходимо рассматривать и случаи, когда на допустимые значения параметров наложены некоторые ограничения, например, допускаются только такие наборы чисел $a_1, \dots, a_k$, для которых выполняется предикат $\pi(a_1, \dots, a_k)$. Любой набор чисел $a_1, \dots, a_k$ удовлетворяющий некоторому предикату, назовем **выборкой, допустимой данным предикатом.**

В дальнейшем важное место будут иметь выборки $a_1, \dots, a_k$, допустимые предикатом

$$a_j - a_i \geq c \quad , \qquad (3.1)$$

где $c \in Z$; $i,j \in \{1,\dots,k\}$. Для обозначения такого условия будем писать

$$N_i \lesssim_c N_j \qquad (3.2)$$

и говорить, что **параметр** N_j данного выражения **выше параметра** N_i **на константу** c . Условие (3.3) фактически означает, что допустимы только такие выборки значений $a_1,\dots,a_k$ параметров $N_1,\dots,N_k$, для которых имеет место (3.I).

Будем говорить, что **параметр** N_j **просто выше параметра** N_i и писать

$$N_i \lesssim N_j ,$$

если имеет место

$$\exists c \in Z : \quad N_i \lesssim_c N_j .$$

Пусть для выборок параметров $N_1,\dots,N_k$ выражения E имеет место $N_1 \lesssim_{c_1} N_2 \lesssim_{c_2} \dots \lesssim_{c_{k-1}} N_k$. Тогда будем говорить, что параметры **линейно упорядочены**. Будем говорить, что параметры $N_1,\dots,N_k$ **линейно упорядочены с запасом** c $(c \in Z)$, если $N_1 \lesssim_c N_2 \lesssim_c \dots \lesssim_c N_k$.

Пусть $\mathcal{T}$ – некоторый терм выражения E , и $I_1,\dots,I_n$ – все свободные параметры терма $\mathcal{T}$ (включая α). Зафиксируем значения $b_1,\dots,b_k$ свободных параметров $N_1,\dots,N_k$ выражения E . Рассмотрим возможные выборки $a_1,\dots,a_n$ значений параметров $I_1,\dots,I_n$ при данных $b_1,\dots,b_k$. Будем говорить, что **терм** $\mathcal{T}$ **– линейный** (**линейный с запасом** c) при данной выборке $b_1,\dots,b_k$, если существуют такие $i_1,\dots,i_n$, что имеет место $I_{i_1} \lesssim I_{i_2} \lesssim \dots \lesssim I_{i_n}$ (соответственно, $I_{i_1} \lesssim_c I_{i_2} \lesssim_c \dots \lesssim_c I_{i_n}$). Будем говорить, что **терм** $\mathcal{T}$ ρ**-отделимый** $(\rho \in N)$, если

он линейный с запасом $P(c^{T}_{max}+1)$, где c^{T}_{max} - максимальное абсолютное значение чисел, встречающихся в теле T терма T. Будем говорить, что **выражение** E **P-отделимое** (для данных $b_1, \dots, b_n$), если все сложные термы E P-отделимые.

Рассмотрим терм $\overline{T} = [\tau(I)]_{I=\alpha,\beta}$. Пусть $\alpha = I_1 + c_1$, $\beta = I_2 + c_2$ $(c_1, c_2 \in \mathbb{Z})$. Тогда будем писать $\alpha \lesseqgtr \beta$, если имеет место $I_1 \underset{c_1-c_2, c_1}{\lesseqgtr} I_2$. Назовём терм $\overline{T}$ **односторонним**, если $\alpha \lesseqgtr \beta$ или $\beta \lesseqgtr \alpha$. Будем говорить, что **направление** δ одностороннего терма $\overline{T}$, равно $+1$, если $\alpha \lesseqgtr \beta$, а -1 если $\beta \lesseqgtr \alpha$.

Пусть дан односторонний терм $T = [\tau(I)]_{I=\alpha,\beta}$. **Частичным раскрытием терма T степени n** назовём выражение

$$T(\alpha)\dots T(\alpha+\delta\cdot(n-1))\,[\tau(I)]_{I=\alpha+\delta_n,\beta-\delta_n}\,T(\beta-\delta\cdot(n-1))\dots T(\beta),$$

где δ - направление терма, $n \geq 1$.

В дальнейшем мы будем использовать преобразования выражений, в которых термы будут заменяться их частичными раскрытиями, а частичные раскрытия иногда будут заменяться обратно термами. Поэтому удобно ввести понятие т.н. аннотированного частичного раскрытия. Для этого будем использовать специальные символы $[$ и $]$. **Аннотированным частичным раскрытием терма** $T = [\tau(I)]_{I=\alpha,\beta}$ назовём выражение:

$$\left[T(\alpha)\dots T(\alpha+\delta\cdot(n-1))\,[\tau(I)]_{I=\alpha+\delta_n,\beta-\delta_n}\overset{\circ}{}\,T(\beta-\delta\cdot(n-1))\dots T(\beta)\right].$$

Теперь можно определить **виртуальное частичное раскрытие терма** $T = [\tau(I)]_{I=\alpha,\beta}$. Если терм T имеет глубину

I, то виртуальным частичным раскрытием терма $\bar{T}$ является просто аннотированное раскрытие терма $\bar{T}$. Пусть определено виртуальное частичное раскрытие для термов имеющих глубину меньше κ (где $\kappa > 1$). Пусть терм $\bar{T} = [\tau(I)]_{I=\alpha,\beta}$ имеет глубину κ. Виртуальным частичным раскрытием терма $\bar{T}$ назовём выражение:

$$\left[T^{(i)}(\alpha) \ldots T^{(n)}(\alpha+\delta\cdot(n-1))\right] [\tau(I)]_{I=\alpha,\beta-\delta_n} \left[T'^{(n)}(\beta-\delta\cdot(n-1)) \ldots T'^{(i)}(\beta)\right],$$

где $T^{(i)}(\alpha+\delta\cdot(i-1))$, $T'^{(i)}(\beta-\delta\cdot(i-1))$ - выражения полученные из $\tau(I)$ заменой термов их произвольными частичными раскрытиями. Подвыражения $T^{(i)}(\alpha) \ldots T^{(n)}(\alpha+\delta\cdot(n-1))$ и $T'^{(i)}(\beta-\delta\cdot(n-1)) \ldots T'^{(i)}(\beta)$ назовём **виртуальными продолжениями терма $\bar{T}$ длины n.**

Будем говорить, что в выражении E терм $\bar{T} = [\tau(I)]_{I=\alpha,\beta}$ **Q - изолирован**, если он односторонний и входит в подвыражение

$$T^{(i)}(\alpha-\delta Q) \ldots T^{(i)}(\alpha-\delta) [\tau(I)]_{I=\alpha,\beta} T'^{(i)}(\beta+\delta) \ldots T'^{(i)}(\beta+\delta Q),$$

где $T^{(i)}$, $T'^{(i)}$ - виртуальные частичные раскрытия τ. Иными словами, терм $\bar{T}$ Q-изолирован, если с обоих сторон от него стоят его виртуальные продолжения длины Q.

Будем говорить, что выражение E Q-изолировано, если все его термы Q-изолированы.

Оказывается, что P-отделимость и Q-изолированность при достаточно больших P и Q как раз и являются теми условиями, которым должно удовлетворять выражение, чтобы правила преобразования описанные в 2 преобразовали одинаково развёртку до глубины I и выражение. Поэтому важна следующая

ЛЕММА 3. Пусть дано выражение $E(N_1, \ldots, N_k)$. Пусть $a_1, \ldots, a_k$ некоторые достаточно большие и

отличающиеся между собой значениями параметров. Тогда существует алгоритм, который для натуральных P и Q преобразует данное выражение E в ассимптотически эквивалентное выражение E' такое, что E' является P -отделимым и Q -изолированным.

Доказательству этой леммы будет посвящена остальная часть данного параграфа. Из доказательства будет видно, что полученное выражение фактически будет P -отделимым и Q -изолированным не только при данных значениях $a_1, \dots, a_k$, но также при любой выборке значений, удовлетворяющей условию $N_{i_1} \lesssim N_{i_2} \lesssim \dots \lesssim N_{i_k}$ при достаточно большом c, где $i_1, i_2, \dots, i_k$ такие, что $a_{i_1} < a_{i_2} < \dots < a_{i_k}$.

В дальнейшем для простоты будем рассматривать только унарные выражения, для произвольных выражений доказательство аналогичное.

Множество, состоящее из всех свободных параметров терма $\bar{T}$ и его собственного параметра, назовём множеством **внешних** параметров терма $\bar{T}$. Определим **отношение подчиненности параметров** $(\to)$ в выражении.

1. Для свободных параметров выражения N и σ имеет место

$$\sigma \to N$$

2. Пусть для свободных параметров $I_1, \dots, I_n$ терма $\bar{T}$ имеет место отношение подчиненности:

$$I_{i_1} \to I_{i_2} \to \dots \to I_{i_k} \to I_{i_{k+1}} \to \dots \to I_{i_n}$$

и пусть $\bar{T} = [T(I_0)]_{\underline{I_0} = \underline{I_{i_k}} + c_1,\ \underline{I_{i_{k+1}}} + c_2}$

или $\bar{T} = [T(I_0)]_{\underline{I_0} = \underline{I_{i_{k+1}}} + c_1,\ \underline{I_{i_k}} + c_2}$

Тогда имеет место отношение подчиненности:

$$I_{i_1} \to I_{i_2} \to \dots \to I_{i_k} \to I_0 \to I_{i_{k+1}} \to \dots \to I_{i_n}.$$

Термы, у которых для всех свободных параметров $I_1, \dots, I_k$ имеет место

$$I_{i_1} \to I_{i_2} \to \dots \to I_{i_k}$$

назовём **исправимыми**. Термы, у которых для всех внешних параметров $I_0, \dots, I_k$ имеет место

$$I_{i_0} \to I_{i_1} \to I_{i_2} \to \dots \to I_{i_k}$$

назовём **правильными**. Например, в выражении

$$E = [[\underline{I_1}]_{\underline{I_1}=0,\underline{I}}\ [\underline{I_2}]_{\underline{I_2}=\underline{I},N}\ [\underline{I_3}]_{\underline{I_3}=0,\underline{N}}]_{I=0,N}$$

термы $[\underline{I_1}]_{\underline{I_1}=0,\underline{I}}$ $(0 \to I_1 \to I \to N)$ и $[\underline{I_2}]_{\underline{I_2}=\underline{I},N}$ $(0 \to I \to I_2 \to N)$ – правильные, а терм $[\underline{I_3}]_{\underline{I_3}=0,\underline{N}}$ – неправильный.

Заметим, что прямые подтермы правильных термов являются исправимыми.

Очевидно, из $I_1 \to I_2$ следует $I_1 \leqslant I_2$ при любом достаточно большом α. Если кроме $I_1 \to I_2$ имеет место и $I_1 \leqslant I_2$, то будем писать $I_1 \overset{c}{\to} I_2$. Терм, для всех внешних параметров которого имеют место отношения

$$I_{i_0} \overset{c}{\to} I_{i_1} \overset{c}{\to} \dots \overset{c}{\to} I_{i_k}$$

назовём **c-правильным**.

Перейдём к описанию алгоритма преобразований. Он будет основан на трёх нижеописанных процедурах.

Первая процедура – **процедура раскрытия терма $\overline{T}$ в степени c** $(c \in \mathbb{N})$ применима к термам с односторонними границами и заменяет терм

$$\overline{T} = [T(I)]_{I=\alpha,\beta}$$

на аннотированное частичное раскрытие

$$\left[\tau(\alpha)\dots\tau(\alpha-\delta\cdot(c-1))\,[\tau(I)]_{I=\alpha-\delta c,\beta+\delta c}\,\tau(\beta+\delta\cdot(c-1))\dots\tau(\beta)\right].$$

Очевидно, если $\alpha \lesssim \beta$ или $\beta \lesssim \alpha$, то полученное выражение асимптотически эквивалентно начальному. Главное свойство этой процедуры заключается в том, что после его применения к терму $\overline{\tau} = [\tau]_{I=\alpha,\beta}$ имеет место $\alpha \lesssim_{c'} I \lesssim_{c'} \beta$ (или $\beta \lesssim_{c'} I \lesssim_{c'} \alpha$), где c' зависит только от c (если $\alpha = \underline{I}_1 + c_1$ и $\beta = \underline{I}_2 + c_2$, то $c' = \min\{c - |c_1|, c - |c_2|\}$), но не от α.

Вторая процедура - **процедура разбиения терма** применима только к исправимым термам и она заменяет исправимый терм $\overline{\tau}$ на конкатенацию правильных термов $\overline{\tau}_1, \dots, \overline{\tau}_n$ той же глубины как $\overline{\tau}$. Точнее, если для свободных параметров терма

$\overline{\tau} = [\tau(I)]_{I=\underline{I}_{i_k}+c_k,\ \underline{I}_{i_\ell}+c_\ell}$ имеет место:

$$I_{i_1} \to \dots \to I_{i_k} \to \dots \to I_{i_\ell} \to \dots \to I_{i_m}$$

или $I_{i_1} \to \dots \to I_{i_\ell} \to \dots \to I_{i_k} \to \dots \to I_{i_m}$,

то $\overline{\tau}$ заменяется на конкатенацию:

$$[\tau(I^0)]_{\underline{I}^0=\underline{I}_{i_k}+c_k,\ \underline{I}_{i_{k+1}}}\ [\tau(I^1)]_{\underline{I}^1=\underline{I}_{i_{k+1}}+\delta,\ \underline{I}_{i_{k+2}}}$$

$$[\tau(I^1)]_{\underline{I}^1=\underline{I}_{i_{k+2}}+\delta,\ \underline{I}_{i_{k+3}}}\ \dots\ [\tau(I^{\delta(\ell-k)})]_{\underline{I}^{\delta(\ell-k)}=\underline{I}_{i_{\ell-1}},\ \underline{I}_{i_\ell}+c_\ell}$$

Очевидно, если для терма $\overline{\tau}$ имеет место

$$I_{i_k} \lesssim_{c_k} I_{i_{k+1}} \lesssim_1 I_{i_{k+2}} \lesssim_1 I_{i_{k+3}} \lesssim_1 \dots \lesssim_1 I_{i_{\ell-1}} \lesssim_{c_\ell+1} I_{i_\ell}$$

или

$$I_{i_\ell} \lesssim_{c_\ell+1} I_{i_{k+1}} \lesssim_1 I_{i_{k+2}} \lesssim_1 I_{i_{k+3}} \lesssim_1 \dots \lesssim_1 I_{i_{k-1}} \lesssim_{c_k} I_{i_k},$$

то полученное выражение асимптотически эквивалентно первоначальному. Назовём такие термы **разбивными**. Очевидно, все термы $\overline{T}_j = [T(I^j)]_{\underline{I}^j = I_{i_{j-1}+1}, I_{i_{j-1}+2}}$ правильные и имеют односторонние границы.

Третья процедура - **процедура отдаления параметров терма $\overline{T}$ на расстояние d** $(d \in \mathbb{N})$ применима к правильным термам с односторонними границами, все предки которых также имеют односторонние границы. Определим процедуру рекурсивно.

Пусть $\overline{T} = [T]_{\underline{I}=\underline{I}_{i_k}+c_k,\ \overline{I}_{i_k}+c_k}$ и $c = max(-c_1, c_2, 0) + d$

Если $incl(\overline{T}) \geq 1$, то

- применить процедуру отдаления параметров к первому предку (т.е. к прямому предку $\overline{T}$) на расстоянии $2c$;
- применить процедуру раскрытия терма $\overline{T}$ в степени c.

Если $incl(\overline{T}) = 0$, то - применить процедуру раскрытия терма $\overline{T}$ в степени c.

Заметим, что работа процедуры фактически заключается в последовательном применении процедур раскрытия ко всем предкам терма $\overline{T}$. В полученном выражении можно естественным образом определить терм соответствующий терму $\overline{T}$. Очевидно, имеют место следующие свойства.

1. Полученное выражение асимптотически эквивалентно исходному выражению.
2. Для терма полученного выражения соответствующему терму $\overline{T}$ имеет место:

$$I_{i_0} \xrightarrow{d} I_{i_1} \xrightarrow{d} \cdots \xrightarrow{d} I_{i_k}$$

(процедура "отдаляет" параметры на расстояние d).

Процедуру отдаления параметров можно использовать для преобразования исправимых термов в разбивные. На самом деле, пусть терм $\overline{T} = [T]_{\underline{I}=\underline{I}_{i_k}+c_k,\ \overline{I}_{i_k}+c_k}$ - исправимый терм. Тогда его предки являются правильными

термами. Применим к прямому предку терма $\bar{T}$ процедуру отделения параметров на расстояние $d = \max(|c_1|, |c_2|, 1)$, 1) . Тогда после преобразования терм $\bar{T}'$, соответствующий терму $\bar{T}$, получится разбивным. К терму $\bar{T}'$ теперь можно применить процедуру разбиения, в результате чего получим конкатенацию правильных термов $\bar{T}_1, \ldots, \bar{T}_n$ с односторонними границами. Теперь их потомки являются исправимыми, с помощью отделения параметров их можно преобразовать в разбивные, а дальше, с помощью процедуры разбиения - в правильные с односторонними границами. Таким образом, спускаясь вниз по дереву выражения, мы постепенно можем преобразовать все термы в правильные с односторонними границами.

Опишем более подробно алгоритм преобразования выражения в асимптотически эквивалентное, все термы которого правильные с односторонними границами. Мы будем обходить дерево выражения сверху вниз [2] и применять к вершинам (т.е. к термам) попеременно процедуры отделения параметров и разбиения. По определению, самый внешний терм является правильным.

В отличие от простого обхода дерева сверху вниз, при применении процедур отделения параметров и разбиения образуются новые ветви (новые подтермы) в дереве. Поэтому необходимо доказать, что обход когда-нибудь закончится. Для этого заметим, что все новые неправильные подтермы, которые образуются, имеют глубину на единицу меньше глубины рассматриваемого терма.

Таким образом, мы получим выражение, все термы которого правильные и имеют односторонние границы. В результате теперь можно применить процедуру отделения параметров к любому подтерму этого выражения. Заметим, что после применения процедуры отделения в полученном выражении все термы также будут правильными с односторонними границами. Покажем теперь, как такое выражение преобразовать в P-отделимое. Из конструкции будет видно, что алгоритм можно легко изменить так, чтобы получить и Q-изолированность.

Сначала заметим, что если абсолютное значение максимальной константы, встречающейся в теле T терма $\bar{T}$ равно $c_{l_{max}}$, то применяя к терму $\bar{T}$ процедуру отделения параметров на расстояние $d = (c_{l_{max}}+1)\cdot P$ терм $\bar{T}$ преобразуется в P-отделимый. Чтобы преобразовать выражение E в P-отделимое, необходимо применить такую процедуру ко всем термам. Однако здесь появляется трудность, так как применяя процедуру к терму $\bar{T}$, у других термов константы могут увеличиться. Поэтому обход дерева надо построить так, чтобы преобразованные термы уже не портились. Это возможно, так как процедура отделения параметров обладает следующими свойствами:

Свойство 1. Процедура, применённая к терму $\bar{T}$, меняет только предков терма $\bar{T}$ и его границы, а тело $\bar{T}$ терма $\bar{T}$ и все его потомки и братья не меняются.

Свойство 2. Если для какого-нибудь потомка $\bar{T}'$ терма $\bar{T}$ имеет место $I_{i_0} \supseteq \dots \supseteq I_{i_m}$, то после применения процедуры к $\bar{T}$ для $\bar{T}'$ это свойство остаётся в силе.

Будем теперь обходить дерево выражения, применяя процедуру отделения параметров на определенное расстояние снизу вверх [2], т.е. будем применять эту процедуру к очередному терму и отмечать его как обработанный только после того, как все его потомки (подтермы) обработаны. Тогда из свойств 1 и 2 следует, что ни один из обработанных термов не будет позже "портиться". Но так как при применении процедуры отделения параметров число подтермов выражения увеличивается, мы должны доказать, что процесс обхода остановится. Для этого отметим следующее свойство процедуры отделения параметров.

Свойство 3. Пусть к какому нибудь потомку $\bar{T}'$ глубины d_0 терма $\bar{T}$ применена процедура отделения параметров. Тогда новые подтермы, которые могут образоваться у терма $\bar{T}$, будут иметь глубину не больше $d_0 - 1$.

Справедливость этого свойства непосредственно следует из конструкции процедуры отделения параметров.

Докажем теперь индукцией, что процесс обхода закончится для любого поддерева глубины d_0. Для этого предложим по индукции, что процесс обхода завершится для любого поддерева глубины $d_0 - 1$. Пусть далее, терм $\bar{T}$ глубины d_0 имеет прямые поддеревья $\bar{T}_1, \ldots, \bar{T}_k$, глубина которых $d_1, \ldots, d_k \leq d_0 - 1$. Начнём обход этих поддеревьев начиная с тех, которые имеют самую большую глубину - $d_0 - 1$. В процессе обхода у дерева $\bar{T}$ образуются новые поддеревья, но согласно свойству 3, они имеют глубину не больше $d_0 - 2$. Затем обходим поддеревья глубины $d_0 - 2$, в результате образуются новые поддеревья глубиной не более $d_0 - 3$. Затем обходим поддеревья глубиной $d_0 - 3$, $d_0 - 4$, и т.д. до 1. Таким образом, процесс обхода завершится. Мы показали, как преобразовать произвольное выражение в P-отделимое.

Заметим теперь, что свойство P-отделимости сохранилось также и в случае, если мы при обходе дерева выражения применили бы процедуру отделения параметров на расстояние большее, чем $d = (c^{T_{d_0}} + 1) \cdot P$. Но так как процедура отделения параметров фактически заключается в применении процедуры раскрытия, то очевидно, если взять достаточно большое d мы можем в результате получить Q-изолированное выражение для любого Q.

Лемма доказана.

4. Основные преобразования выражений

В данном параграфе будет описан некоторый алгоритм Π_1 преобразования выражений, обладающий следующим свойством. Если выражение E P-отделимое и Q-изолированное для достаточно больших P, Q, то для преобразованного выражения $\Pi_1(E)$ имеет место (2.15):

$$VAL_1(\Pi_1(E), a) = S(VAL(E, a)).$$

Основную часть алгоритма преобразования Π_1 составляет т.н. линейный преобразователь Λ, преобразующий линейные выражения (т.е. содержащее только термы глубины один). Λ в свою очередь будет основан на некоторых нижеописанных правилах преобразования линейных выражений. При этом Λ, так же как и S, будет зависеть от некоторого параметра Пусть этот параметр зафиксирован.

Опишем упомянутые выше правила преобразования.

Пусть

$$L = l_1 t_1 l_2 t_2 \dots l_i t_i l_{i+1} \dots l_n t_n l_{n+1}$$

– линейное выражение, где $l_i = \gamma_1^i \gamma_2^i \dots \gamma_{m_i}^i$ – простые слова, а $t_i = \langle \alpha_1^i \dots \alpha_{n_i}^i \bullet\bullet\bullet \beta_1^i \dots \beta_{n_i}^i \rangle$ – простые термы.

Здесь для терма $t = \langle \alpha_1 \dots \alpha_n \bullet\bullet\bullet \beta_1 \dots \beta_n \rangle$ иногда нам будет удобно пользоваться записью

$$t = \langle \alpha_1 \dots \alpha_n \circ \alpha_1' \dots \alpha_n' \bullet\bullet\bullet \beta_1' \dots \beta_n' \circ \beta_1 \dots \beta_n \rangle$$

где

$$\alpha_i' = \begin{cases} \alpha_i, & \text{если } \alpha_i = \beta_i, \\ \alpha_i + \delta \ (\delta - \text{направление терма } t), & \text{если } \alpha_i \neq \beta_i. \end{cases}$$

$$\beta_i' = \begin{cases} \beta_i, & \text{если } \alpha_i = \beta_i, \\ \beta_i - \delta, & \text{если } \alpha_i \neq \beta_i \end{cases}$$

Определим операцию $\dot{-}$ следующим образом:

$$a \dot{-} b = \begin{cases} a - b, & \text{если } a, b \text{ имеют вид } c \text{ или } \underline{I+c}, \\ & \text{где } I \in \mathbb{N},\ c \in \mathbb{Z} \\ 0, & \text{если } a = b \\ & \text{не определено в остальных случаях} \end{cases}$$

Под разницей $a - b$ здесь понимается алгебраическая разница, например, $\underline{I+7} - 8 = \underline{I-1}$.

Определим теперь сами правила преобразования. Будем говорить, что терм t_i можно сдвинуть вправо в выражении L, если l_{i+1} непустое, $\gamma_1^{i+1} \dot{-} \beta_1^i$ определено и имеет место

$$\gamma_1^{i+1} \dot{-} \beta_1^i = \beta_1^i \dot{-} \beta_1^{i1}.$$

Если терм t_i можно сдвинуть вправо, то под сдвигом t_i вправо в выражении

$$L = L_1 t_i \gamma_1^{i+1} L_2$$

будем понимать замену выражения L на

$$\hat{L} = L_1 \alpha_1^i \hat{t}_i L_2,$$

где $\hat{t}_i = \langle \alpha_1^i \dots \alpha_{n_i}^i \alpha_1^{i1} \dots \beta_1^i \dots \beta_{n_i}^i \gamma_1^{i+1} \rangle$

Будем говорить, что терм t_i можно расширить влево в выражении L, если l_i – непустое, $\alpha_j^i \dot{-} \gamma_{n_i - n_i + j}^i$ определено для $j \in \{1, 2, n\}$ и

$$\alpha_j^{i1} \dot{-} \alpha_j^i = \alpha_1^i \dot{-} \gamma_{n_i - n_i + j}^i.$$

Если терм t_i можно расширить влево в L, то под расширением терма t_i влево в выра-

жении:

$$L = L_1 \gamma^i_{m_i - n_i + 1} \dots \gamma^i_{m_i} t_i L_2$$

будем понимать замену выражения на

$$\hat{L} = L_1 \hat{t}_i L_2 ,$$

где $\hat{t}_i = \langle \gamma^i_{m_i - n_i + 1} \dots \gamma^i_{m_i} \circ\circ\circ \beta^i_1 \dots \beta^i_{n_i} \rangle$

Будем говорить, что **термы** t_i и t_{i+1} **можно объединить** в выражении L, если l_{i+1} - пустое, периоды термов t_i и t_{i+1} одинаковы и

$$\alpha^{i+1}_j \div \alpha^{i+1}_j = \alpha^{i+1}_j \div \beta^i_j = \beta^i_j \div \beta^i_j$$

для $j \in \{1, \dots, n_i\}$.

Если термы t_i и t_{i+1} можно объединить в L, то под их **объединением** будем понимать замену выражения

$$L = L_1 t_i t_{i+1} L_2$$

на $\hat{L} = L_1 \hat{t}_i L_2$,

где $\hat{t}_i = \langle \alpha^i_1 \dots \alpha^i_{n_i} \circ\circ\circ \beta^{i+1}_1 \dots \beta^{i+1}_{n_{i+1}} \rangle$.

Рассмотрим терм $\bar{T} = [T]_{\Gamma = \alpha, \beta}$ выражения E. **Фактором этого терма** назовем значение $\beta \div \alpha$. Если $\beta \div \alpha = c$, где $c \in \mathbb{Z}$, то будем говорить, что **фактор терма** $\bar{T}$ **постоянный**. В противном случае определим **минимальное значение фактора терма** $\bar{T}$, как минимальное из возможных числовых значений выражения $\beta \div \alpha$, при данных значениях свободных параметров выражения.

Пусть зафиксировано некоторое число $c_0 \in \mathbb{N}$. Тогда будем говорить, что **терм** t_i в выражении L **можно**

заменить его значением при данном факторе c_0, если фактор терма постоянный и меньше c_0.

Если терм t_i в L можно заменить его значением при данном факторе, то под заменой будем понимать замену выражения

$$L = L_1 t_i L_2$$

на $L' = L_1 val(t_i) L_2$.

Остается еще определить правило синтеза при данном факторе c_0, под которым будем понимать один шаг алгоритма синтеза S упомянутый в 2. Правило применяется к простым подсловам рассматриваемого выражения.

Отметим, что все правила преобразования обладают следующим свойством: полученное в результате применения одного из правил выражение асимптотически эквивалентно первоначальному.

Опишем теперь сам алгоритм Λ. Алгоритм Λ, так же как и алгоритм синтеза S, будет зависеть от некоторого параметра $c_0 \in N$. Алгоритм Λ применим только к линейным выражениям с односторонними термами. Фактически мы опишем один шаг алгоритма - последовательное применение правил преобразования в определенном порядке к определенному подвыражению рассматриваемого выражения L. Выполнять шаги алгоритма мы будем слева направо, при этом каждое выполнение шага будет начинаться с того места, где закончилось предыдущее. Правую часть выражения, к которой еще не применялись шаги алгоритма, назовем необработанной частью выражения.

Один шаг алгоритма Λ заключается в следующем.

1. Сначала применяем правило синтеза (с параметром c_0) к простому подвыражению необработанной части выражения, находящемуся левее первого необработанного терма справа. Синтезированный в результате этого терм отмечается как необработанный.

2. Затем к первому необработанному терму слева применяем:

а) правило расширения, пока это возможно;
б) правило объединения, пока это возможно;
в) правило сдвига, один раз, если это возможно.

Пункты а), б) и в) повторяются один за другим в указанном выше порядке, пока хотя бы один из них применим.

3. Далее применяем к первому необработанному терму правило развертки с фактором c_0, если это возможно.

После выполнения пунктов I, 2 и 3 полученный терм (или слово $val(t)$ - если применялось правило развертки) отмечаем как обработанный и применяем опять описанные выше действия I, 2 и 3 к необработанной части.

Легко видеть, что описанный выше алгоритм Λ, работая на произвольном линейном выражении L, работу закончит после выполнения конечного числа шагов, и для полученного выражения $\Lambda(L)$ будет иметь место $\Lambda(L) \sim L$.

Для описания алгоритма преобразования выражений нам понадобится еще **операция перестановки квадратных скобок**. Но сначала отметим некоторые свойства алгоритма Λ.

Рассмотрим линейное выражение L. Пусть $R(L)$ - класс всех линейных выражений, полученных из L, разными частичными раскрытиями термов выражения L. Через d_0^i обозначим максимальную длину тел термов выражения $L_i = \Lambda(L_i)$, где $L_i \in R(L)$. Пользуясь следствием 2.I, можно доказать, что существует $\max_i \{d_0^i\}$ по всем выражениям данного класса $R(L)$, и его можно оценить. Пусть

$$d_0 = \max_i \{d_0^i\}$$

(ниже будет дана эта оценка для более общего случая).

Пусть теперь данное выражение L является Q-изолированным, где

$$Q \geq 4d_0 .$$

Такое выражение назовем развернутым. Пользуясь леммой 2.I, можно доказать следующее свойство развернутых выражений.

Свойство I. Пусть $L = L_1 t_1 \dots t_2 L_2$ - развернутое вы-

ражение, где L_1, L_2 - линейные подвыражения, t_1, t_2 - простые термы, l - простое подвыражение. Тогда существует такое l', что

$$l = l_1 l' l_2 \qquad \text{и}$$

$$\Lambda(L) = \Lambda(L_1 t_1 l_1)\, l' \,\Lambda(l_2 t_2 L_2)$$

(т.е. алгоритм Λ на подвыражениях $L_1 t_1 l$ и $l t_2 L_2$ работает независимо).

Символы подслова l' назовем изолирующими. В общем случае слово l' может быть и пустым, тогда будем говорить, что между подвыражениями $L_1 t_1 l_1$ и $l_2 t_2 L_2$ находится изолирующая точка. В этом случае подвыражения $L_1 t_1 l_1$ и $l_2 t_2 L_2$ будем называть независимыми. Очевидно, следующее свойство независимых подвыражений.

Свойство 2. Пусть $L = L_1 L_2$, где L_1 и L_2 - два независимых подвыражения. Тогда

$$\Lambda(L) = \Lambda(L_1)\,\Lambda(L_2) .$$

Таким образом, развернутые выражения можно разделить в независимые подвыражения, к которым Λ можно применить отдельно. Любые два терма входят в одно и то же независимое подвыражение тогда и только тогда, когда они объединены.

Отметим следствие свойства 2.

СЛЕДСТВИЕ 2. Пусть $L = L_1 t L_2$ - линейное выражение и t - терм, к которому при работе алгоритма Λ не применяется правило развертки. Пусть $L' = L_1 t L_2$, где L' - произвольное линейное выражение. Тогда в подвыражениях выражений L и L' имеются одни и те же изолированные точки.

Пользуясь следствием I, можно доказать следующее важное свойство алгоритма Λ

Свойство 3. Пусть L - конкретное линейное выражение. Тогда, если $c \geq |t|_{max} + 5$, где $|t|_{max}$ - максимальная длина тел термов выражения L, то

$$\Lambda(L) = S(VAL(L))$$

(через $S(A)$ обозначено $S(A,c_0)$ при фиксированном c_0).

Более подробно это свойство доказано в работе /3/ (только в несколько ином контексте).

Рассмотрим теперь выражение

$$F = [T(I)]_{I=\alpha,\beta}\, T(\beta+\delta)\,, \qquad (4.I)$$

где δ – направление терма $\overline{T}$ (по определению терм $\overline{T}$ – односторонний).

Пусть $T(I) = T_1(I)\,T_2(I)$, где $T_1(I)$ и $T_2(I)$ – подвыражения. Тогда легко доказать, что выражение (4.I) эквивалентно следующему выражению:

$$F' = T_1(\alpha)\,[T_2(I)\,T_1(I+\delta)]_{I=\alpha,\beta}\, T_2(\beta+\delta)\,.$$

Такое преобразование F в F' назовем перестановкой квадратных скобок на κ символов, где $\kappa = |T_1(I)|$ – длина подслова $T_1(I)$.

Перестановку квадратных скобок можно определить и для более общего случая. Рассмотрим выражение

$$\overline{F} = [T(I)]_{I=\alpha,\beta}\, T'(\beta+\delta)\,,$$

где $T'(\beta+\delta)$ – виртуальное продолжение терма $[T(I)]_{I=\alpha,\beta}$. Так как по T' можно однозначно восстановить T, то операцию перестановки квадратных скобок очевидным образом можно определить и в этом случае. Если $T(I) = T_1(I)\,T_2(I)$ (где $T_1(I)$ и $T_2(I)$ – подвыражения), то и $T'(I)$ можно представить в форме $T'(I) = T_1'(I)\,T_2'(I)$, где $T_1(I) \sim T_1'(I)$ и $T_2(I) \sim T_2'(I)$. Таким образом, результатом перестановки квадратных скобок в выражении $\overline{F}$ будет выражение

$$\overline{F}' = T_1'(\alpha)\,[T_2(I)\,T_1(I+\delta)]_{I=\alpha,\beta}\, T_2'(\beta+\delta)\,.$$

Теперь опишем алгоритм преобразования выражений Работа алгоритма будет заключаться в циклическом выполнении некоторого шага. Этот шаг будет применяться к подвыражениям, состоящим из внешних сложных термов выражения, и линейному подвыражению, находящемуся левее данного сложного терма $\overline{T}$. Шаг будет заключаться в а) применении линейного преобразователя Λ к некоторому т.н. расширенному линейному фрагменту данного выражения; б) перестановке квадратных скобок рассматриваемого сложного терма $\overline{T}$; в) рекурсивном запуске самого алгоритма Π_1 на выражении T, являющемся телом терма $\overline{T}$ (точнее, не самого терма $\overline{T}$ а терма $\overline{T}'$, полученного после перестановки квадратных скобок). Таким образом, Π_1 должен быть описан для произвольных, т.е. не только для унарных выражений. В случае, если рассматриваемое выражение линейное, то Π_1 совпадает с Λ.

Если выражение E является Q-изолированным для $(Q \geq 1)$, то любому смежному терму $\overline{T}$ всегда предшествует линейное подвыражение L. E можно записать в форме

$$E = L_1\overline{T}_1 L_2\overline{T}_2 \dots L_n\overline{T}_n L_{n+1},$$

где $L_1, \dots, L_{n+1}$ – линейные подвыражения, а $\overline{T}_1, \dots, \overline{T}_n$ – внешние сложные термы.

Рассмотрим некоторое подвыражение $L_i\overline{T}_i$ и определим понятие расширенного линейного фрагмента для $L_i\overline{T}_i$.

Пусть $\overline{T}_i = [T_i(I)]_{I=\alpha,\beta}$. Рассмотрим два случая:

1) $depth(\overline{T}_i) = 2$, т.е. $T_i(I)$ – линейное. Тогда расширенным линейным фрагментом подвыражения назовем линейное выражение

$$L_i^* = L_i T_i(\alpha)$$

2) $depth(\overline{T}_i) > 2$.

Тогда T_i также можно записать в форме

$$T_i(I) = L_i^1(I)\overline{T}_i^1(I)L_i^2(I)\overline{T}_i^2(I)\ldots L_i^{n_i}(I)\overline{T}_i^{n_i}(I)L_i^{n_i+1}(I)$$

Назовем терм $\overline{T}_i^1$ первым внутренним термом терма $\overline{T}_i$. Расширенным линейным фрагментом подвыражения $L_i\overline{T}_i$ в этом случае назовем линейное выражение

$$L_i^* = L_i L_i^1(\alpha) \cdot$$

Очевидно, каждому символу линейного расширенного фрагмента соответствует символ в $L_i\overline{T}_i$ и, следовательно, в E.

Опишем теперь алгоритм Π_1 точнее.

Пусть

$$E = L_1\overline{T}_1 \ldots L_{i-1}\overline{T}_{i-1}L_i\overline{T}_i \ldots L_n\overline{T}_n L_{n+1},$$

где $L_1, \ldots, L_{n+1}$ – линейные подвыражения, $\overline{T}_1, \ldots, \overline{T}_n$ – внешние сложные термы.

Пусть подвыражение $L_1T_1 \ldots L_{i-1}T_{i-1}$ отмечено как обработанная часть выражения. Тогда алгоритм Π_1 действует следующим образом:

1. Запускается алгоритм линейных преобразований Λ, на первый расширенный линейный фрагмент необработанной части выражения, т.е. на $L_iT_i(\alpha)$, если $depth(\overline{T}_i)=2$ и на $L_iL_i^1(\alpha)$, если $depth(\overline{T}_i) > 2$.

2. Переставляются квадратные скобки терма $\overline{T}_i$ до символа E, соответствующего первому изолирующему символу $L_iT_i(\alpha)$ (соответственно, $L_iL_i^1(\alpha)$, начиная с первого символа $T_i(\alpha)$ (соответственно, $L_i(\alpha)$). Полученное выражение обозначим через $L_i\overline{T}_i'$.

3. В случае, если $depth(\overline{T}_i) > 2$ и левая скобка первого внутреннего терма $\overline{T}_i^1$ стоит сразу после левой переставленной скобки терма $\overline{T}_i$, то к $\overline{T}_i'$ применяется алгоритм Π_1. Если $depth(\Pi_1(T_i'))$, то переставляются квадратные скобки терма $\overline{T}_i$ за термом $\overline{T}_i^1$ и заменяется $\overline{T}_i^1$ на $[\Pi_1(T_i')]$. Затем осуществляется

переход к I.

4. Применяется алгоритм Π_1 к телу терма T_i. Далее, если $depth(\Pi_1(T_i)) > 0$, то отмечается $L_i T_i$ как обработанное и осуществляется переход к 2.

Глубину терма $[\Pi_1(T)]$ будем называть **фактической глубиной терма** $\bar{T} = [T]_{i=\alpha,\beta}$ т.е. фактическая глубина терма $\bar{T}$ определяется как $depth(\Pi_1(T)) + 1$.

Для доказательства корректности алгоритма, т.е. для доказательства равенства

$$VAL_1(\Pi_1(E); a_1, \dots, a_n) = S(VAL(E; a_1, \dots, a_n))$$

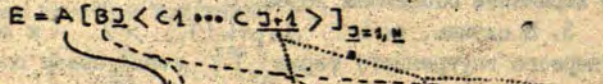
определим еще один алгоритм преобразования Π_2, преобразующий развертку до глубины один (т.е. $VAL_1(E; a_1, \dots, a_n)$), и докажем, что

$$VAL_1(\Pi_1(E); a_1, \dots, a_n) = \Pi_2(VAL_1(E; a_1, \dots, a_n)) \quad (4.2)$$

и

$$\Pi_2(VAL_1(E; a_1, \dots, a_n)) = S(VAL(E; a_1, \dots, a_n)). \quad (4.3)$$

Фактически алгоритм Π_2 будет работать не на развертке $VAL_1(E; a_1, \dots, a_n)$, но на т.н. аннотированной развертке $\overline{VAL}_1(E; a_1, \dots, a_n)$. Для того, чтобы определить аннотированную развертку, необходимо установить связь между символами и подсловами выражения E и символами и подсловами разверток $VAL_k(E; a_1, \dots, a_n)$ этого же выражения. Например,



$$VAL_1(E,3) = A\,B1\langle c1 \dots c1\rangle\,B2\langle c1 \dots c3\rangle\,B3\langle c1 \dots c4\rangle,$$

где стрелками $\longrightarrow$, $\dashrightarrow$ и $\cdots\!\!\rightarrow$ указаны символы развертки $VAL_k(E; a_1, \dots, a_n)$, соответствующие символам выражения E : A, B и $\underline{J+1}$, соответственно. Их

назовем **отображениями** соответствующих **символов** выражения, а соответствующие символы E назовем **прообразами**. Точно это понятие можно определить индуктивно. Таким же образом можно определить и отображения подслов выражения в развертке.

Часто нам будет удобно говорить не об отображениях самих символов, а об отображениях **позиций**, занимаемых символами или находящихся между данными символами данного выражения. Например, в $VAL_1(E,3)$ между третьим и четвертым символами слева (т.е. между "$_1$" и "$<$") находится позиция отображения позиции E, занимаемой квадратной скобкой: "$[$".

Иногда нам понадобится указать в развертке $VAL_k(E;c_1,\dots,c_n)$ границы подслов, являющихся образами тел термов выражения Например, для нашего примера

$$VAL_1(E;3) =$$

$$= A[B1\langle c1 \dots c2\rangle \cdot B2\langle c1 \dots c3\rangle \cdot B3\langle c1 \dots c4\rangle] \cdot$$

Символами "$[$" и "$]$" отмечены границы первого и последнего отображения тела терма, а символом "$\cdot$" – границы между различными отображениями данного тела терма. Если в такой развертке дополнительно еще указаны прообразы всех символов, то развертка называется аннотированной и обозначается через $\overline{VAL}_k(E;c_1,\dots,c_n)$.

Опишем теперь работу алгоритма Π_2 на аннотированной развертке $\overline{VAL}_1(E;c_1,\dots,c_n)$. Для этого, во-первых, мы определим операцию перестановки квадратных скобок для аннотированных разверток. Пусть

$$\overline{VAL}_1([T(I)]_{I=\alpha,\beta}T'^{(\beta+\delta)};\ c_1,\dots,c_n) =$$

$$= [\overline{VAL}_1(T'^{\alpha})\cdot T(\alpha+\delta)\cdot \dots \cdot T(\beta);\ c_1,\dots,c_n)]\,\overline{VAL}_1(T'^{(\beta+1)};c_1,\dots,c_n)$$

– аннотированная развертка выражения $[T(I)]_{I=\alpha,\beta}T'^{(\beta+\delta)}$, где δ – направление терма $T=[T(I)]_{I=\alpha,\beta}$, а $T'^{(\beta+\delta)}$

– первое продолжение терма $\bar{T}$. Пусть $T(I)=T_1(I)T_2(I)$ и $T(\beta+\delta)=T_1(\beta+\delta)T_2(\beta+\delta)$. Тогда перестановку квадратных скобок в аннотированном примере определим как преобразование данного выражения в выражение

$$\overline{VAL}_1(T_1(\alpha);a_1,\dots,a_n)\,[\overline{VAL}_1(T_1(\alpha)T_1(\alpha+\delta)\cdot T_1(\alpha+\delta)T_1(\alpha+2\delta)\cdot\dots$$

$$\dots\cdot T_2(\beta)T_2(\beta+\delta);a_1,\dots,a_n)]\,\overline{VAL}_1(T_2(\beta+\delta);a_1,\dots,a_n)\ .$$

В случае аннотированной развертки также можно определить и понятие расширенного линейного фрагмента. Пусть дана аннотированная развертка

$$\bar{A}=\bar{L}_1[\bar{A}_1^1\cdot\bar{A}_1^2\cdot\dots\cdot\bar{A}_1^{n_1}]\,\bar{L}_2[\bar{A}_2^1\cdot\bar{A}_2^2\cdot\dots\cdot\bar{A}_2^{n_2}]\dots$$

$$\dots\,\bar{L}_i[\bar{A}_i^1\cdot\bar{A}_i^2\cdot\dots\cdot\bar{A}_i^{n_i}]\dots\bar{L}_m[\bar{A}_m^1\cdot\bar{A}_m^2\cdot\dots\cdot\bar{A}_m^{n_m}]\,\bar{L}_{m+1}\ ,$$

где $\bar{L}_1,\dots,\bar{L}_{m+1}$ – линейные подслова, а $[\bar{A}_i^1\cdot\bar{A}_i^2\cdot\dots\cdot\bar{A}_i^{n_i}]$ – аннотированные развертки внешних подтермов. Тогда расширенным линейным фрагментом подслова

$$\bar{L}_i[\bar{A}_i^1\cdot\bar{A}_i^2\cdot\dots\cdot\bar{A}_i^{n_i}]$$

назовем слово

$\bar{L}_i\bar{A}_i^1$, если $\bar{A}_i^1$ не содержит квадратных скобок, и

$\bar{L}_i\bar{L}_i^1$, где $\bar{L}_i^1$ – левая часть $\bar{A}_i^1$ до первой квадратной скобки, если $\bar{A}_i^1$ содержит квадратные скобки.

Алгоритм Π_2 работает таким же образом, как и алгоритм Π_1, если

а) под расширенным линейным фрагментом понимать расширенный линейный фрагмент для аннотированных разверток;

б) под перестановкой скобок понимать перестановку скобок в аннотированной развертке,

в) под рекурсивным запуском алгоритма на тело терма понимать запуск алгоритма на всех отображениях тела данного терма,

г) если при выполнении пунктов 3 или 4 алгоритма глубина какого-то терма оказывается равной единице, то квадратные скобки, соответствующие этому терму, вычеркиваются.

Докажем теперь утверждение (4.2):

$$VAL_1(\Pi_1(E); \vec{\alpha}) = \Pi_2(\overline{VAL}_1(E, \vec{\alpha})),$$

где под $\Pi_2(\overline{VAL}_1(E; \vec{\alpha}))$ понимается линейное выражение, которое получается из $\overline{VAL}_1(E; \vec{\alpha})$ после работы алгоритма Π_2 и вычеркивания всех квадратных скобок. Это равенство должно выполняться, если в $\vec{\alpha} = (\alpha_1, \dots, \alpha_n)$ все значения $\alpha_1, \dots, \alpha_n$ достаточно большие и достаточно отличаются друг от друга.

Рассмотрим сначала линейный случай, когда Π_1 и Π_2 редуцируются к линейному преобразователю Λ

Рассмотрим линейное выражение

$$L(\vec{N}) = L(N_1, \dots, N_n)$$

Пусть $\vec{\alpha} = (\alpha_1, \dots, \alpha_n)$ и $\vec{\beta} = (\beta_1, \dots, \beta_n)$,

где α_i и β_i либо целые числа, либо имеют вид: $\underline{I+c}$, где $I \in \mathbf{I}$, $c \in \mathbb{Z}$. Будем говорить, что л и н е й -

ный преобразователь Λ работает одинаково на выражениях $L_1 = L(\vec{\alpha})$ и $L_2 = L(\vec{\beta})$, если имеют место:

$$\Lambda(L(\vec{\alpha})) = \Lambda(L(\vec{\beta}))(\vec{\beta}|\vec{\alpha}) \qquad (4.4)$$

и $$\Lambda(L(\vec{\beta})) = \Lambda(L(\vec{\alpha}))(\vec{\alpha}|\vec{\beta}) \cdot \qquad (4.5)$$

Фактически достаточно требовать лишь одно из этих равенств, так как из одной следует другая.

Таким образом, мы должны доказать, что если $L(\vec{N})$ - линейное выражение и $\vec{a} = (a_1, \dots, a_k)$ - некоторый набор, где значения $a_1, \dots, a_k$ достаточно большие и достаточно отличаются друг от друга, то на выражениях $L(N)$ и $L(\vec{a})$ алгоритм Λ работает одинаково. Более точно, мы докажем следующую лемму:

ЛЕММА 4. Пусть L - линейное выражение, c_{max} - максимальное абсолютное значение констант выражения L, d_0 - максимальная длина тел термов выражения $\Lambda(L)$ (как мы уже упомянули, его можно оценить) и пусть $N_1, \dots, N_k$ - некоторые параметры выражения L. Пусть $L' = L(N_1|a_1, \dots, N_k|a_k)$, где $a_1, \dots, a_k \in \mathbb{N}$ такие, что

1) $\forall i \in \{1, \dots, k\}: \; a_i - c_{max} > 2(c_{max} + d_0) + 1$,

2) $\forall i, j \in \{1, \dots, k\}: \; |a_i - a_j| > 2(c_{max} + d_0) + 1$. ○

Тогда алгоритм Λ на выражениях L и L' работает одинаково.

ДОКАЗАТЕЛЬСТВО. По определению мы должны доказать:

$$\Lambda(L(N_1, \dots, N_k))(N_1|a_1, \dots, N_k|a_k) = \Lambda(L(a_1, \dots, a_k)) \; .$$

Выберем некоторый параметр выражения: N_i. Пусть максимальное значение аддитивных констант параметра N_i равно c_i, т.е. для любой константы, которая встречается в подвыражении $N_i + c$, имеет место: $|c| \le c_i$. Пусть

$L_1 = L(N_1|c_1)$, где $c_1 > c_{max} + c_1 + 1$

Заметим, что тогда в L_1 все константы можно разделить на два непересекающихся класса $\bar{c}_1$ и $\bar{c}_1$, где

$$\bar{c}_1 = \{c \in Z \mid -c_{max} \le c \le c_{max}\}$$

$$\bar{c}_2 = \{c \in Z \mid c_1 - c_1 \le c \le c_1 + c_1\} =$$

$$= \{c \in Z \mid c_{max} + 1 \le c \le c_{max} + 1 + 2c_1\} .$$

Теперь легко доказать, что если какое-нибудь из правил преобразования линейных выражений можно применить к какому-нибудь подвыражению выражения L , то это же правило можно применить и к соответствующему подвыражению L_1 и наоборот. Обозначим через $\pi(L)$ и $\pi(L_1)$ результаты применения какого-нибудь подвыражения L и L_1 , соответственно. Тогда, очевидно,

$$\pi(L(N_1))(N_1|c_1) = \pi(L_1)$$

так как для любой пары символов α и β выражения L и соответствующей пары α_1 и β_1 выражения L_1 имеет место:

$$|\alpha - \beta| \le 1 \iff |\alpha_1 - \beta_1| \le 1$$

и

$$|\alpha - \beta| > 1 \iff |\alpha_1 - \beta_1| > 1 .$$

Если бы максимальные абсолютные значения c_{max} и c_1 не увеличивались, то индукцией легко можно было бы доказать, что

$$\Lambda(L(N_1))(N_1|c_1) = \Lambda(L_1) . \qquad (4.6)$$

Однако в результате применения правила сдвига константы c_{max} и c_1 могут увеличиваться на единицу. В то же время заметим, что из конструкции алгоритма Λ следует, что если к данному терму t правило сдвига применено $|t|$ - раз (где $|t|$ - длина тела терма t), то после этого к t

будет применяться правило расширения и в результате увеличенные константы опять уменьшатся на $|t|$. Таким образом, при работе алгоритма Λ константы не могут увеличиться больше, чем на d_0, где d_0 - как и прежде наибольшая длина термов выражения $\Lambda(L)$. Следовательно, если

$$a_1 > c_{max} + c_1 + d_0,$$

то равенство (4.6) выполняется.

Теперь утверждение леммы легко можно доказать индукцией по числу заменных параметров.

Лемма доказана.

Вернемся к равенстве (4.2). Для его доказательства достаточно доказать следующие два утверждения.

УТВЕРЖДЕНИЕ 1. Если выражение E является P-отделимой и Q-изолированной для достаточно больших P и Q, то при работе Π_1 все расширенные линейные фрагменты, к которым применяются алгоритм Λ, удовлетворяют условиям леммы 4.

УТВЕРЖДЕНИЕ 2. Если выражение E является P-отделимым и Q-изолированным для достаточно больших P и Q, то при работе Π_1 полученные промежуточные выражения как минимум являются $(P-c)$-отделимыми и $(Q-c)$-изолированными, где $c = depth(E)$.

Для доказательства, во-первых, оценим максимальную возможную длину тел простых термов с переменным фактором или фактором больше c_0 среди всех термов выражений E' которые можно получить из E при работе Π_1.

Рассмотрим "достаточно большой" пример $VAL(E', \vec{c})$ выражения E (можно считать, что $VAL(E', \vec{c}_0) = VAL(E, \vec{c})$). Каждому простому терму E', удовлетворяющему вышеупомянутым требованиям, соответствует в $VAL(E', \vec{c})$ регулярная последовательность длины не меньше c_0, которая не (c_0-5)-покрывает другую регулярную последовательность с фактором не меньше c_0. Поэтому длина тела терма меньше или равна наибольшему периоду регулярных последовательностей, удовлетворяющих данному ус-

ловию. Согласно следствию I, такие регулярные последовательности не могут содержать другие регулярные подпоследовательности с фактором больше или равным c_0. Поэтому наибольшую длину периода d_0 можно оценить, если известна наибольшая длина D_0-последовательности, не содержащей ни одной регулярной подпоследовательности с фактором больше или равным c_0:

$$d_0 < \frac{D_0}{c} .$$

Индукцией по глубине выражения нетрудно получить следующую оценку для D_0

$$D_0 < (c_0|E|)^n ,$$

где $n = depth(E)$

Таким образом,

$$d_0 < c_0^{n-1}|E|^n .$$

Пусть теперь L - некоторое линейное подвыражение тела T сложного терма $\overline{T}$, внешние параметры $I_0, \dots, I_k$ которого обладают свойством $I_0 \lesssim I_1 \lesssim \dots \lesssim I_k$ для $c > 2(c_{max}+d_0)+1$. Пусть далее $L' = L(I_0|c_0, \dots, I_k|c_k)$, где $c_0, \dots, c_k$ - некоторая выборка параметров $I_0, \dots, I_k$. Тогда L и L' удовлетворяют условиям леммы 4. Если еще вспомнить определение отделимости, то получаем следующую лемму:

ЛЕММА 5. Пусть L - линейное подвыражение выражения $T(I)$, где $[T(I)]_{I=c,d}$ P-отделимый терм в данном выражении E, $P > 2d_0+1$. Пусть L' - произвольный образ подвыражения L в $VAL_1(E, \vec{a})$. Тогда $\Lambda(L)(I_0|c_0, \dots, I_k|c_k) = \Lambda(L')$, где $c_0, \dots, c_k$ - образы параметров $I_0, \dots, I_k$ в образе L'.

Таким же образом можно доказать и следующую лемму.

ЛЕММА 5А. Пусть L^* - линейное подвыражение выражения

$T(\alpha)T(\alpha+\delta)$, где $\bar{T}=[T(I_0)]_{I=\alpha\beta}$ (δ – направление терма $\bar{T}$) – P-отделимый терм в данном выражении E, $P>2(d_0+1)+1$. Пусть L^{*1} – произвольный образ подвыражения L^* в $VAL_1(E,\vec{c})$. Тогда $\Lambda(L^*)(I_1|c_1, \ldots, I_k|c_k)=\Lambda(L^{*1})$, где $c_0,\ldots,c_k$ – образы параметров $I_0,\ldots,I_k$ в образе L^{*1}.

Из этих двух лемм следует

СЛЕДСТВИЕ 3. Пусть E – произвольное P-отделимое и Q-изолированное выражение, где $P>2(d_0+1)+1$ и $Q>4d_0$. Пусть $\bar{T}$ – произвольный терм выражения E, входящий в подвыражение $L\bar{T}$, где L – линейное подвыражение E . Пусть L^{*1} – расширенный линейный фрагмент подвыражения $L\bar{T}$ и пусть L^{*1} – произвольный образ подвыражения L^* в $VAL_1(E,\vec{c})$. Тогда $\Lambda(L^*)(I_0|c_0, \ldots, I_k|c_k)=\Lambda(L^{*1})$, где $c_0,\ldots,c_k$ – образы параметров $I_0,\ldots,I_k$ в образе L^{*1}.

Этим утверждение I доказано.

Переходим к доказательству утверждения 2. Для этого сначала докажем следующую лемму.

ЛЕММА 6. Пусть $\bar{T}=[T(I)]_{I=\alpha\beta}$ – P-отделимый терм выражения E, $P>2(d_0+1)+1$, $dep_{t_1}(\bar{T})=2$ и $T(I)$ – развернутое выражение. Рассмотрим выражение $T(\alpha)T(\alpha+\delta)$. Тогда между подвыражениями $T(\alpha)$ и $T(\alpha+\delta)$ или в подвыражении $T(\alpha+\delta)$ находится изолирующая точка.

ДОКАЗАТЕЛЬСТВО. Таким же методом, как в лемме 5 , можно доказать, что на выражениях $T(\alpha)$ и $T(\alpha+\delta)$ алгоритм Λ работает одинаково. Из этого следует, что $T(\alpha)$ и $T(\alpha+\delta)$ имеют одни и те же изолирующие точки.

Рассмотрим в подвыражении $T(\alpha)$ последний терм $t(\alpha)$ среди термов с переменным или достаточно большим факторами. Имеются две возможности: либо за $t(\alpha)$ следует изолирующая точка, либо ее нет.

Рассмотрим первую возможность. В этом случае в подвыражении $T(\alpha+\delta)$ за соответствующим термом $t(\alpha+\delta)$ также следует изолирующая точка, а согласно следствию 2, изолирующая точка следует и за соответствующим термом

$t(\alpha+\delta)$ всего выражения $T(\alpha)T(\alpha+\delta)$, что и требовалось доказать.

Рассмотрим вторую возможность: за последним термом $t(\alpha)$ среди термов с переменным или достаточно большим фактором изолирующей точки нет. Рассмотрим значение $VAL(T(\alpha)T(\alpha+\delta); c_0,\dots,c_k)$, где $c_0,\dots,c_k$ – выборка параметров $I_0,\dots,I_k$ при каком-нибудь достаточно большом значении свободных параметров $N_1,\dots,N_k$ данного выражения E. Из сказанного выше следует, что в слове $A = VAL(T(\alpha)T(\alpha+1); c_0,\dots,c_k)$ подслово, начинающееся не позже, чем с образа терма $t(\alpha)$ и продолжающееся до конца A, является регулярным. Так как это регулярное подслово пересекается с регулярным подсловом, являющимся образом $t(\alpha)$, то по лемме I период равен длине тела терма $t(\alpha)$. С другой стороны, так как $t(\alpha)$ и $t(\alpha+\delta)$ имеют переменные или достаточно большие факторы, то слова $VAL(t(\alpha); c_0,\dots,c_k)$ и $VAL(t(\alpha+\delta); c_0,\dots,c_k)$ содержат достаточно длинные подслова A_1 и A_2, имеющие одинаковую длину, такие, что для соответствующих символов α_i^1 и α_i^2 этих подслов A_1 и A_2 имеет место

$$\alpha_i^1 = \alpha_i^2 \qquad \text{или} \qquad |\alpha_i^1 - \alpha_i^2| \le 1.$$

Из этих равенств можно получить противоречие таким же образом, как при доказательстве леммы I. Таким образом, второй случай также невыполним.

Этим лемма 6. доказана.

Теперь легко доказать, что при работе алгоритма Π_1 квадратные скобки любого сложного терма могут быть переставлены на число символов не больше, чем длина тела этого терма. Действительно, для термов, имеющих фактическую глубину 2, это следует из леммы 6, а для остальных термов это очевидно. Отсюда, в свою очередь, получаем, что если терм был P-отдельным и Q-изолированным, то после перестановки он в худшем случае будет $(P-1)$-отделимым и $(Q-1)$-изолированным. Отсюда следует справедли-

вость утверждения 2.

Теперь равенство (4.2), т.е.

$$VAL_1(\Pi_1(E), \vec{c}) = \Pi_2(\overline{VAL}_1(E, \vec{c}))$$

можно доказать простой индукцией по глубине выражения E. Переходим к доказательству равенства (4.3), т.е.:

$$\Pi_2(\overline{VAL}_1(E, \vec{c})) = S(VAL(E, \vec{c})) .$$

Доказательство этого утверждения также состоит из двух частей:

$$\Pi_2(\overline{VAL}_1(E, \vec{c})) = \Lambda(VAL_1(E, \vec{c})) \qquad (4.7)$$

$$\Lambda(VAL_1(E, \vec{c})) = S(VAL(E, \vec{c})) . \qquad (4.8)$$

Равенство (4.8) является свойством 3 линейного преобразователя Λ. Поэтому остается доказать лишь равенство (4.7). Для этого нам понадобится следующая лемма.

ЛЕММА 7. Пусть $\bar{T} = [\bar{T}(I_0, I_1, \dots, I_n)]_{I_0 = \alpha, \beta}$ - P-отделимый терм выражения E, где $P > 2(d_0 + 1) + 1$, $\alpha = \underline{I}_\rho + c_\rho$, $\beta = \underline{I}_\gamma + c_\gamma$. Рассмотрим выражения

$$T(a_0, a_1, \dots, a_n)\, T(a_0 + \delta, a_1, \dots, a_n)$$
$$T(\alpha_0 - \delta, a_1, \dots, a_n)\, T(\alpha_0, a_1, \dots, a_n)$$
$$T(\beta_0, a_1, \dots, a_n)\, T(\beta_0 + \delta, a_1, \dots, a_n) ,$$

где $a_0, a_1, \dots, a_n$ - произвольная выборка значений параметров $I_0, I_1, \dots, I_n$, $\alpha_0 = a_\rho + c_\rho$ и $\beta_0 = a_\gamma + c_\gamma$. Пусть $L(I_0, I_1, \dots, I_n)$ - произвольное линейное подвыражение выражения $T(I_0, I_1, \dots, I_n)\, T(I_0 + 1, I_1, \dots, I_n)$. Тогда на выражениях

$$L(\alpha_0, \alpha_1, \dots, \alpha_n)$$

$$L(\alpha_0, \alpha_1, \dots, \alpha_n)$$

$$L(\beta_0, \alpha_1, \dots, \alpha_n)$$

линейный преобразователь Λ работает одинаково.

Эту лемму можно доказать таким же образом, как и лемму 5.

СЛЕДСТВИЕ 4. В выражениях $L(\alpha_0, \alpha_1, \dots, \alpha_n)$ $L(\alpha_0, \alpha_1, \dots, \alpha_n)$ и $L(\beta_0, \alpha_1, \dots, \alpha_n)$ имеются одни и те же изолирующие точки.

Теперь можно доказать следующую лемму:

ЛЕММА 8. Пусть $\overline{B} = \overleftarrow{L}(\alpha_0 - \delta)[\overline{A}(\alpha_0) \cdot \overline{A}(\alpha_0 + \delta) \cdot \dots \cdot \overline{A}(\beta_0)]\overline{A}''(\beta_0 + \delta)$ – аннотированное подслово аннотированной развертки $\overline{VAL}_1(E, \mathfrak{S})$, где $\overline{A}(\gamma_0)$ $\gamma_0 \in \{\alpha_0, \alpha_0 + \delta, \dots, \beta_0\}$ – аннотированная развертка тела T терма $\overline{T} = [T(\Gamma)]_{\Gamma = \alpha, \beta}$, $\overleftarrow{L}(\alpha_0 - \delta)$ – правая линейная часть аннотированной развертки левого виртуального продолжения терма $\overline{T}$, $\overline{A}''(\beta_0 + \delta)$ – аннотированная развертка правого виртуального продолжения терма $\overline{T}$. Рассмотрим расширенный фрагмент слова $\overline{B}$ и переставим квадратные скобки согласно алгоритму Γ_2. Получим

$$\overline{B}' = \overleftarrow{L}(\alpha_0 - \delta)\overline{A}_1(\alpha_0)[\overline{A}_2(\alpha_0)\overline{A}_1(\alpha_0 + \delta) \cdot \overline{A}_2(\alpha_0 + \delta)\overline{A}_1(\alpha_0 + 2\delta) \cdot$$

$$\dots \cdot \overline{A}_2(\beta_0)\overline{A}_1(\beta_0 + \delta)]\overline{A}_2(\beta_0 + \delta),$$

где $A_1(\gamma_0)A_2(\gamma_0) = \overline{A}(\gamma_0)$. Тогда квадратные скобки $[,]$ и точки отделения образов тела терма $\overline{T}$: $\cdot$ – находятся в изолирующих позициях, линейного выражения

$$\overleftarrow{L}(\alpha_0 - \delta)A(\alpha_0)A(\alpha_0 + \delta) \dots A(\beta_0)A''(\beta_0 + \delta),$$

где $A(\gamma_0)$, $A''(\beta_0 + \delta)$ получается соответственно из $\overline{A}(\gamma_0)$ и $\overline{A}_2(\beta_0 + \delta)$ вычеркиванием аннотаций.

ДОКАЗАТЕЛЬСТВО. Сначала докажем, что левая квадратная скобка находится в изолирующей точке. Не очевиден лишь случай, когда терм $\bar{T}$, скобки которого мы переставляем, имеет глубину больше двух, и скобки передвигаются до левой скобки первого внутреннего сложного терма (т.е. между обеими квадратными скобками нет непустого подслова). Легко видеть, что это возможно лишь в случае, когда первый внутренний сложный терм имеет фактическую глубину 1 и объединим с термом, стоящим левее его. Но согласно определению алгоритма Π_2 тогда эти термы будут объединены. Этот процесс объединения продолжится, пока не наступит момент, когда данное условие не будет выполнено. Такой момент когда-нибудь наступит. Действительно, если терм, квадратные скобки которого мы переставляем, имеет фактическую глубину больше двух, то это очевидно. Если терм имеет фактическую глубину равную двум, то его тело содержит изолирующую точку и она "остановит" дальнейшее объединение.

Теперь из следствия 4 сразу вытекает справедливость леммы для случая, когда $depth(\bar{T}) = 2$, т.е. когда $A(\Gamma)$ не содержит квадратных скобок.

В случае, когда $depth(\bar{T}) > 2$, мы должны потребовать, чтобы все внутренние сложные термы подвыражения T были бы как минимум 2-изолированными. Тогда, пользуясь леммой 6 и следствием 2, как и в случае $depth(\bar{T}) = 2$ можно доказать справедливость леммы и в данном случае.

Лемма доказана.

Теперь легко видеть, что имеет место следующее утверждение: в выражении $\Pi_2(\overline{VAL}_1(E, \vec{a}))$ все квадратные скобки и точки отделения образов тела термов находятся в изолирующих точках выражения $VAL_1(E, \vec{a})$.

Из этого утверждения легко следует равенство (4.7) и вместе с тем (4.3):

$$\Pi_2(\overline{VAL}_1(E, \vec{a})) = S(VAL(E, \vec{a})).$$

Так как мы уже доказали (4.2):

$$VAL_1(\Pi_1(E),\vec{c}) = \Pi_2(\overline{VAL}(E,\vec{c}))\ ,$$

то доказано основное утверждение данного параграфа:

$$VAL_1(\Pi_1(E),\vec{c}) = S(VAL(E,\vec{a}))\ .$$

Вспомним, что здесь мы должны потребовать, чтобы выражение E было P-отделимым и Q-изолированным, при $P > 2(d_0+1)+1$ и $Q \geq 4d_0$, где $d_0 = c_0^{n-1}|E'|^n$ $|E'| = \max(|E_1|,|E_2|)$, E_1 и E_2 – выражения, эквивалетность которых исследуется, $c_0 = \max(|\Omega|)+5$ ($\max$ берется по простым термам выражений E_1 и E_2), $n = \max(depth(E_1), depth(E_2))$. Вспомним еще, что согласно сказанному в 3, необходимые величины значений параметров $N_1,\dots,N_k$ и $M_1,\dots,M_\ell$ могут быть оценены по P и Q.

Таким образом, доказано, что из

$$VAL(E,\vec{a}) = VAL(F,\vec{b})$$

следует

$$VAL_1(\Pi_1(E),\vec{c}) = VAL_1(\Pi_1(F),\vec{b})\ .$$

Отсюда индукцией, как показано в 2, может быть получено, что из

$$VAL(F,\vec{c}) = VAL(F,\vec{b})$$

следует

$$E \sim F\ .$$

ЛИТЕРАТУРА

1. Барздинь Я.М. Некоторые правила индуктивного вывода и их применение // Семиотика и информатика.- Вып.9 - М.: ВИНИТИ, 1982. - С.59-89.
2. Кнут Д. Искусство программирования для ЭВМ.-М.: Мир, 1976. - С.395.
3. Бразма А. Индуктивные методы синтеза итеративных программ // Проблемы теоретической кибернетики. Тезисы докладов.-Иркутск: Иркутский гос.университет им.А.А.Жданова, 1985. - С.32-33.
4. Barzdin J.M. Some rules of inductive inference and their use for program synthesis // IFIP 83 - Elsevier, North-Holland. - 1983. - P.333-338.

ИНДУКТИВНЫЙ СИНТЕЗ ГРАФИЧЕСКИХ ВЫРАЖЕНИЙ

А.Н.Бразма, И.Э.Этмане
ВЦ ЛГУ им.П.Стучки

В работе [1] введен формальный язык для описания общих закономерностей встречающихся при индуктивном синтезе программ - язык графических выражений. В работе [2] решена проблема эквивалентности графических выражений. В предлагаемой работе решена проблема синтеза графических выражений. Данное решение использует идеи изложенные в [3], а также ряд новых идей.

Переходим к уточнению необходимых понятий и формулировке основного результата работы.

Пусть $\Sigma = \Sigma' \cup \mathbb{Z} \cup \mathcal{U} \cup \mathcal{H}$ - некоторый алфавит, где Σ' - конечное множество символов, $\mathbb{Z}$ - множество целых чисел, $\mathcal{U} = \{[,], =, ', +\}$, $\mathcal{H} = \{x_1, x_2, \ldots\}$ - т.н. множество параметров $(\Sigma' \cap \mathcal{H}) = \emptyset$). Целые чис-

ла мы будем представлять в десятичной форме, при этом числа, состоящие из более чем одной цифры, будем подчеркивать и принимать за один символ.

Определим понятие терма. Пусть T - слово в алфавите Σ, в котором некоторые подслова вида $\mathcal{Y}$ или $\mathcal{Y}+c$ $(\mathcal{Y}\in\mathcal{N},\ c\in\mathcal{Z})$ могут быть подчеркнуты. Т е р м о м $\bar{T}$ называется слово вида:

$$[T]_{\underline{\mathcal{K}}=\alpha,\beta},$$

где $\mathcal{K}\in\mathcal{N}$ и α и β либо константы из $\mathcal{Z}$, либо слова вида $\underline{\mathcal{Y}}$ или $\underline{\mathcal{Y}+c}$ $(\mathcal{Y}\in\mathcal{N})$. T называется т е л о м т е р м а α,β - е г о г р а н и ц а м и, а $\mathcal{K}$ - с о б с т в е н н ы м п а р а м е т р о м т е р м а $\bar{T}$. Терм $\bar{T}$ называется к о р р е к т н ы м, если тело T по крайней мере в одном месте содержит подчеркнутое слово $\underline{\mathcal{K}}$ или $\underline{\mathcal{K}+c}$ $c\in\mathcal{Z}$. В дальнейшем, под термом мы будем понимать именно корректные термы.

Через $T(\mathcal{Y}_1/a_1,\dots,\mathcal{Y}_n/a_n)$, где $\mathcal{Y}_i\in\mathcal{N}$ и $a_i\in\mathcal{Z}$ $(i\in\{1,\dots,n\})$ обозначим слово, полученное из T следующим образом: во всех подчеркнутых местах выражения $\underline{\mathcal{Y}_i}$ и $\underline{\mathcal{Y}_i+c}$ вместо $\mathcal{Y}_i$ подставляем a_i $(i\in\{1,\dots,n\})$ и вычисляем значение, где это необходимо. Например, если $T=A\underline{\mathcal{Y}}B(\underline{\mathcal{Y}+2})$, то $T(\mathcal{Y}/3)=A3B(5)$.

Будем говорить, что границы терма α,β фиксированы, если $\alpha,\beta\in\mathcal{Z}$. Пусть $\bar{T}=[T]_{\mathcal{Y}=\alpha,\beta}$, где α,β фиксированы и $\delta=\begin{cases}+1, & \text{если } \alpha<\beta\\ -1, & \text{если } \alpha>\beta\end{cases}$ (δ будем называть направлением терма $\bar{T}$). Тогда з н а ч е н и е $Val(\bar{T})$ т е р м а $\bar{T}$ определим следующим образом:

$$Val([T]_{\mathcal{Y}=\alpha,\beta})=T(\mathcal{Y}/\alpha)T(\mathcal{Y}/\alpha+\delta)T(\mathcal{Y}/\alpha+2\delta)\dots T(\mathcal{Y}/\beta).$$

Например, если $\bar{T}=[A\underline{\mathcal{Y}}B(\underline{\mathcal{Y}+2})]_{\mathcal{Y}=1,5}$, то

$$Val(\bar{T})=A1B(3)A2B(4)A3B(5)A4B(6)A5B(7).$$

Определим теперь графическое выражение. Пусть A - некоторое слово в алфавите $\Sigma' \cup Z \cup N$ и $Y_1, Y_{21}, \ldots, Y_k \in N$ - параметры, встречающиеся в слове A (они должны быть подчеркнуты в A и входить в A в виде $\underline{Y_i}$ или $\underline{Y_i + c}$, $c \in Z$). Слово A назовем простым графическим выражением, а параметры $Y_1, \ldots, Y_k$ - свободными параметрами в A.

Определим графическое выражение с множеством свободных параметров N' индуктивно.

1. Простое графическое выражение является графическим выражением; все параметры, входящие в это выражение, являются свободными.

2. Если X и Y - графические выражения с множеством свободных параметров N_X и N_Y, то слово XY является графическим выражением с множеством свободных параметров $N_X \cup N_Y$ (X и Y будем называть подвыражениями выражения XY).

3. Пусть X - графическое выражение с множеством свободных параметров N_X. Тогда терм

$$[X]_{\gamma=\alpha,\rho}$$

является графическим выражением с множеством свободных параметров

$$(N_X \setminus \{Y\}) \cup N_\alpha \cup N_\rho,$$

где $N_\alpha = \begin{cases} \{Y_i\}, & \text{если } \alpha = \underline{Y_i} \text{ или } \underline{Y_i + c_1}, \\ \emptyset, & \text{если } \alpha = c_1, \end{cases}$

и

$$N_\rho = \begin{cases} \{Y_2\}, & \text{если } \rho = \underline{Y_2} \text{ или } \underline{Y_2 + c_2}, \\ \emptyset, & \text{если } \rho = c_2, \end{cases}$$

если $\underline{Y} \neq \underline{Y_1}$ и $\underline{Y} \neq \underline{Y_2}$

4. Других графических выражений нет.

Выражение будем считать корректным, если все входящие в него термы являются корректными и имеют различные собственные параметры. В дальнейшем, если не оговорено обратное, под выражениями будем понимать именно коррект-

ные выражения. Если множество свободных параметров графического выражения состоит из одного элемента (пусть это будет N), то такое **выражение** будем называть **унарным**.

Далее нам будет удобно считать, что кроме свободных параметров $\mathcal{Y}_1, \dots, \mathcal{Y}_k$ выражения имеют еще один фиктивный свободный параметр (обозначим его через 0), значение которого равно нулю. Будем считать, что все константы $c \in \mathbb{Z}$, которые не входят в подчеркнутые слова вида $\underline{\mathcal{Y}+c}$, записаны в форме $\underline{0+c}$.

Определим **глубину** $depth(E)$ **выражения** E индуктивно:

1. Если E – простое выражение, то $depth(E)=0$.
2. Если $E=E_1E_2$, то $depth(E)=max(depth(E_1), depth(E_2))$.
3. Если $E=[T]_{\mathcal{Y}=A,B}$, то $depth(E)=depth(T)+1$.

Если $\mathcal{Y}_1, \dots, \mathcal{Y}_k$ – некоторые свободные параметры выражения $\mathcal{W}$, то часто будем писать $\mathcal{W}(\mathcal{Y}_1, \dots, \mathcal{Y}_k)$. В таком случае вместо $\mathcal{W}(\mathcal{Y}_1/a_1, \dots, \mathcal{Y}_k/a_k)$, где $a_1, \dots, a_k \in \mathbb{Z}$ будем писать просто $\mathcal{W}(a_1, \dots, a_k)$. Так, например, значение терма $T=[T]_{\mathcal{Y}=c_1,c_2}$ можно записать $\mathcal{VAL}(T)=T(c_1)T(c_1+\delta)\dots T(c_2)$, где δ – направление терма T.

Пусть $\mathcal{W}(\mathcal{Y}_1, \dots, \mathcal{Y}_k)$ – графическое выражение, где $\mathcal{Y}_1, \dots, \mathcal{Y}_k$ – свободные параметры данного выражения. **Конкретизацией** выражения $\mathcal{W}(\mathcal{Y}_1, \dots, \mathcal{Y}_k)$ назовем выражение $\mathcal{W}(c_1, \dots, c_k)$, где $c_1, \dots, c_k \in \mathbb{Z}$. Если $\mathcal{Y}_1, \dots, \mathcal{Y}_k$ – все свободные параметры выражения $\mathcal{W}$, то полученное выражение $\mathcal{W}(c_1, \dots, c_k)$ назовем **конкретным**. Очевидно, в конкретном выражении границы всех внешних термов фиксированы.

Первую развёртку $\mathcal{VAL}^1(\mathcal{W}(c_1, \dots, c_k))$ конкретного выражения мы получаем заменяя в $\mathcal{W}(c_1, \dots, c_k)$ все внешние термы их значениями. Например,

$$\mathcal{VAL}^1([[\underline{\mathcal{Y}+1}]_{\mathcal{Y}=1,\mathcal{Y}+3}]_{\mathcal{Y}=3,1}) = [\underline{\mathcal{Y}+1}]_{\mathcal{Y}=1,5}[\underline{\mathcal{Y}+1}]_{\mathcal{Y}=1,4}[\underline{\mathcal{Y}+1}]_{\mathcal{Y}=1,3}.$$

Развертка $CAL(a)$ конкретного выражения определяется индуктивно.

1. Если $depth(a)=0$, то $CAL(a)=a$.
2. Если $depth(a)>0$, то $CAL(a)=CAL(CAL^{-1}(a))$.

Например,

$$CAL([[x+1]_{x=1,x+2}]_{x=3,1} = 23456\,2345234$$

Часто вместо $CAL(a;c_1,\dots,c_n)$ будем писать $CAL(a;c_1,\dots,c_n)$.

Рассмотрим теперь унарные графические выражения.

Унарные графические выражения E_1 и E_2 будем называть асимптотически эквивалентными, если существует такое $c\in N$, что для каждого a такого, что $a>c$, имеет место $CAL(E_1;a)=CAL(E_2;a)$.

Пусть теперь E – унарное выражение и пусть $X=CAL(E;a)$ – некоторая его развёртка $(a\in N)$. Пару $P=(X;a)$ будем называть примером выражения. Задача синтеза состоит в том, чтобы по P построить выражение E', асимптотически эквивалентное E. Для этого будут определены некоторые правила индуктивного вывода.

Систему S правил вывода назовем асимптотически полной для унарных графических выражений, если для любого унарного графического выражения E существует константа $c\in N$, такая, что какое бы значение входного параметра $a>c$ мы ни брали, система S преобразует пример $P=(CAL(E;a);a)$ в графическое выражение E', асимптотически эквивалентное E.

Основным результатом данной работы является построение некоторой системы правил индуктивного вывода и доказательство её асимптотической полноты.

Для определения требуемых правил индуктивного вывода, введем сначала необходимые понятия. Пусть X является конкретным выражением. Будем говорить, что под-

выражение (подслово) $Y = x_1^1 \ldots x_p^1 x_1^2 \ldots x_p^2 \ldots\ldots x_1^f \ldots x_p^f$ выражения X является (p,f) - р е г у л я р н ы м, если:

1) Y не содержится ни в одном терме выражения,

2) существует такой терм $T_y = [T_y]_{y=1,f}$ с $|T_y| = p$ ($|Z|$ - число символов в последовательности Z), что $Val(T_y) = Y$; терм T_y будем называть п о р о ж д а ю щ и м термом подвыражения Y, p - п е р и о д о м, а f - ф а к т о р о м.

Под п р и в л е к а т е л ь н о с т ь ю π (p,f)-регулярной последовательности с периодом p и фактором f будем называть отношение

$$\pi = \frac{f}{p}$$

Первое правило индуктивного вывода фактически будет заключаться в том, что регулярное подвыражение, имеющее максимальную привлекательность, заменяем соответствующим термом и данный терм в некотором смысле максимально расширяем вправо и смещаем влево. Например, рассмотрим одно применение правила вывода к выражению:

$$X = [\underline{Y}BA]_{\underline{Y}=1,3}\ 4A5A6A4A5A9A.$$

После замены подвыражения $4A5A6A4A5A9A$ термом, получаем

$$X' = [\underline{Y}BA]_{\underline{Y}=1,3}\ [\underline{Y}A]_{\underline{Y}=4,9},$$

после максимального смещения влево получаем

$$X'' = [\underline{Y}BA]_{\underline{Y}=1,2}\ 3B[A\underline{Y}]_{\underline{Y}=4,9}A.$$

Однако, чтобы расширение и смещение определить более точно, нам понадобится некоторые понятия. Пусть A и B - конкретные выражения. Будем говорить, что B является л е в ы м (п р а в ы м) п о д в ы р а ж е н и е м A, если существует такое выражение C,

что $РАЗ(A) = РАЗ(B)РАЗ(C)$ (соответственно, $РАЗ(A) = РАЗ(C)РАЗ(B)$). Это запишем следующим образом: $B \sqsubseteq_L A$ (соответственно $B \sqsubseteq_R A$). Любое выражение C, удовлетворяющее данному условию, будем называть левой (соответственно, правой) разницей. Например, если

$$A_1 = [a\overline{y}]_{\overline{y}=1,9}$$
$$B_1 = a6a7a8a9$$, то

$B_1 \sqsubseteq_L A_1$, а выражения

$$C_1 = [a\overline{y}]_{\overline{y}=1,5}$$ и
$$C_2 = a1a2a3a4a5$$

являются примерами левых разниц выражений A и B. Для нахождения левой (правой) разницы двух конкретных выражений A и B, очевидно, существует тривиальный алгоритм: сравниваем развертки $РАЗ(A)$ и $РАЗ(B)$, если левая (соответственно, правая) часть $РАЗ(A)$ и $РАЗ(B)$ совпадают, то остаток является разницей. Например, так как

$РАЗ(A_1) = a1a2a3a4a5a6a7a8a9$ и
$РАЗ(B_1) = a6a7a8a9$,

то $a1a2a3a4a5$ является разницей между A и B. Однако для построения правил индуктивного вывода нам понадобится более сложный алгоритм нахождения разниц. Содержательно это означает следующее: если B является левым (соответственно, правым) подвыражением выражения A, то разницу C между A и B мы будем искать в такой форме, чтобы по мере возможности она совпадала с левой (соответственно, правой) частью выражения A. Например, в рассмотренном выше примере подходит разница $C_1 = [a\overline{y}]_{\overline{y}=1,5}$. Такую разницу будем называть канонической. Для более точного описания алгоритма нахождения канонической разницы нам понадобится понятие т.н. аннотированной развертки выражения $РАЗ(E; a_{i_1}, \ldots, a_{i_k})$

Пусть дана развертка $X = \overline{CAL}(E; a_1, \ldots, a_n)$. Очевидно, для каждого символа из X можно указать соответствующий символ из выражения E, и наоборот: для каждого символа из E можно указать соответствующие символы из X. Символы из X назовем **отображениями** соответствующих символов из E, а символы из E назовем их **прообразами**. Аналогично можно определить и отображения любого подвыражения из E. Таким образом, в X можно указать все подслова, являющиеся образами тел различных термов. Укажем это следующим образом. Левые и правые границы отображений всего терма отметим символами '$[$' и '$]$', соответственно, а отображения отдельных тел термов отделим символом '$.$'. Например, если

$$A = [[a\underline{x}]_{\underline{x}=1,\underline{x}}]_{\underline{x}=1,6}a \quad \text{и}$$

$$B = a\,[[[\underline{x}a]_{\underline{x}=1,\underline{x}}]_{\underline{x}=1,3}[\underline{x}a]_{\underline{x}=1,2}, \quad \text{то}$$

$$\overline{CAL}(A) = [[a1]\cdot[a1\cdot a2]]\cdot[a1\cdot a2\cdot a3]\cdot[a1\cdot a2\cdot a3\cdot a4]\cdot$$
$$[a1\cdot a2\cdot a3\cdot a4\cdot a5]\cdot[a1\cdot a2\cdot a3\cdot a4\cdot a5\cdot a6]]a,$$

$$\overline{CAL}(B) = a\,[[[1a]\cdot[1a\cdot 2a]\cdot[1a\cdot 2a\cdot 3a]]\,[1a\cdot 2a]].$$

Такие развертки, в которых указаны все границы термов и тел термов, а также для каждого символа его образ, будем называть **аннотированными развертками**. Отметим, что по аннотированной развертке выражения можно легко восстановить само выражение. Теперь можно описать алгоритм построения левой (правой) канонической разницы Δ^*_L (соответственно, Δ^*_R) для конкретных выражений. Опишем его с помощью примера.

Очевидно, для выражений A и B, приведенных выше, имеет место $B \underset{L}{\subseteq} A$. Покажем, как найти их левую каноническую разницу.

Рассмотрим развертки $\overline{CAL}(A)$ и $\overline{CAL}(B)$. Так как $B \underset{L}{\subseteq} A$, то между символами $\overline{CAL}(B)$ и левой частью $\overline{CAL}(A)$ можно установить однозначное соот-

ветствие. Имея ввиду это соответствие преобразуем аннотированную развертку $\overline{RAL}(A)$ следующим образом:

$$(\overline{RAL}(A))' = [[a1]\cdot[a1\cdot a2]\cdot[a1\cdot a2\cdot a3]]\cdot[a1\cdot a2]a3[a4]\cdot$$
$$[[a1\cdot a2\cdot a3\cdot a4\cdot a5]\cdot[a1\cdot a2\cdot a3\cdot a4\cdot a5\cdot a6]]a,$$

т.е. так, чтобы его левая часть совпала с $\overline{RAL}(B)$. Далее, соответствующим образом преобразуем и само A в A':

$$A' = [[a\underline{Y}]_{\underline{Y}=1,\underline{Y}}]_{\underline{Y}=1,3}[a\underline{Y}]_{\underline{Y}=1,2}a3[a\underline{Y}]_{\underline{Y}=4,4}[[a\underline{Y}]_{\underline{Y}=1,\underline{Y}}]_{\underline{Y}=5,6}a$$

A' теперь можно представить в виде: $A' = B'\cdot C'$, где

$$B' = [[a\underline{Y}]_{\underline{Y}=1,\underline{Y}}]_{\underline{Y}=1,3}[a\underline{Y}]_{\underline{Y}=1,2}a$$

$$C' = 3[a\underline{Y}]_{\underline{Y}=4,4}[[a\underline{Y}]_{\underline{Y}=1,\underline{Y}}]_{\underline{Y}=5,6}$$

Так как $\overline{RAL}(B) = \overline{RAL}(B)$, то C' является разницей. Разницу, полученную описанным алгоритмом, назовем канонической и обозначим через $\Delta_L^*(A,B)$, правую разницу, найденную аналогичным образом, обозначим через $\Delta_R^*(A,B)$.

Рассмотрим теперь конкретное выражение $\mathcal{U} = X_1[T(Y)]_{Y=a}^{b}X_2$, где $T = [T(Y)]_{Y=a}^{b}$ некоторый внешний терм выражения $a, b \in \mathcal{Z}$, X_1, X_2 – подвыражения. Пусть $\delta = 1$, если $a \le b$, и $\delta = -1$, если $a > b$. Тогда будем говорить, что терм T можно расширить вправо в выражении $\mathcal{U}$, если существует такое $c \in \mathcal{Z}$, что $c > b$ (соответственно, $c < b$) и имеет место

$$[T(Y)]_{\underline{Y}=b+\delta,c} \subseteq_L X_2.$$

Под расширением терма T тогда будем

понимать замену выражения W^{γ} на

$$W' = X_1[T(y)]_{y=a,c}\ \Delta_L^*(X_2, [T(y)]_{y=b,\delta,c})$$

Например, если

$$W_1 = [[a^{\gamma}]_{\gamma=1,\gamma}]_{\gamma=6,9}\ a_2 a_1 a_1 a_1 a$$

$$\bar{T}_1 = [[a^{\gamma}]_{\gamma=1,\gamma}]_{\gamma=6,9}$$

то, так как $\Delta_L^*(a_2 a_1 a_1 a_1 a, [[a^{\gamma}]_{\gamma=1,\gamma}]_{\gamma=2,1} = a$

получаем $W_1' = [[a^{\gamma}]_{\gamma=1,\gamma}]_{\gamma=6,1}\ a$.

Если $W_2 = [a^{\gamma}]_{\gamma=1,\gamma}[[a^{\gamma}]_{\gamma=\gamma,8}\ 6^{\gamma}]_{\gamma=8,1}$

$$\bar{T}_2 = [a^{\gamma}]_{\gamma=1,\gamma}$$

то, так как

$$\Delta_L^*([[a^{\gamma}]_{\gamma=\gamma,8}\ 6^{\gamma}]_{\gamma=8,1}, [a^{\gamma}]_{\gamma=1,8}) =$$

$$= 68[[a^{\gamma}]_{\gamma=\gamma,8}\ 6^{\gamma}]_{\gamma=8,1}$$

получаем

$$W_2' = [a^{\gamma}]_{\gamma=1,8}\ 68[[a^{\gamma}]_{\gamma=\gamma,8}\ 6^{\gamma}]_{\gamma=8,1}$$

Если

$$W_3 = [[a^{\gamma}]_{\gamma=1,\gamma}]_{\gamma=1,5}\ a[[\chi a]_{\chi=1,L}]_{L=6,9}$$

$$\bar{T}_3 = [[a^{\gamma}]_{\gamma=1,\gamma}]_{\gamma=1,5}$$

то, так как

$$\Delta_L^*(a[[\chi a]_{\chi=1,L}]_{L=6,9}, [[a^{\gamma}]_{\gamma=1,\gamma}]_{\gamma=6,9}) = a,$$

получаем

$$W_3' = [[a^{\gamma}]_{\gamma=1,\gamma}]_{\gamma=1,9}\ a$$

Будем говорить, что терм $\bar{T}$ расширен максимально, если не существует $c \in \mathcal{X}$ такое, что имеет место

$[J(y)]_{y=1+\delta, c+\delta} \underset{L}{\subset} x_2$.

Будем говорить, что терм $\bar{T}$ можно с м е с т и т ь в л е в о в выражении N', если существуют такие выражения $J_1(y)$ и $J_2(y)$, что $J(y) = J_1(y) J_2(y)$ и имеет место:

$J_2(y/a-\delta) \underset{R}{\subset} x_1$.

Под с м е щ е н и е м т е р м а $\bar{T}$ в выражении N' будем понимать замену выражения N' на выражение

$N'' = A_R^*(x_1, J_2(y/a-\delta))[J_2(y-\delta) J_1(y)]_{y=a,b} J_2(y/c) x_2$

Пусть, например, $N_1' = 1[a\,y\,a\,y]_{y=2,5}$

$\bar{T}(y) = [a\,y\,a\,y]_{y=2,5}$

берем $J_1(y) = a\,y\,a$, $J_2(y) = y$. Так как $J_2(y/1) \underset{R}{\subset} 1$, то

$N_1'' = [y\,1\,a\,y\,a]_{y=2,5}$

Если в выражении N'' вновь полученный терм $\bar{T}$ не возможно больше сместить, то будем говорить, что т е р м $\bar{T}$ с м е щ е н м а к с и м а л ь н о.

Определим теперь правила индуктивного вывода.

П р а в и л о A. Пусть y (p,f) - регулярное подвыражение конкретного выражения x с максимальной привлекательностью и фактором $f > 2$ (если таких y несколько, то берем самое левое). Тогда заменяем y на порождающий терм $\bar{T}y$. Далее максимально смещаем и расширяем. Полученное таким образом выражение x' из x будем называть выражением, полученным из x по правилу A.

П р а в и л о B. Пусть дана пара (N, a), где N - конкретное выражение $a \in N$. Пусть x_i - те символы из N, которые принадлежат x и больше $a/2$

Заменяем все такие x_i на $N+c_i$, где $c_i = x_i - a_i$. Остальные символы в W не меняем. Полученное таким образом выражение W' из W будем называть выражением, полученным из W по правилу $\mathcal{B}$.

Пусть теперь $E(X)$ - унарное графическое выражение и пара $P=(X;a)$ - его пример. Тогда правила индуктивного вывода применяются следующим образом. Применяем правило $\mathcal{A}$ к X, в результате получаем выражение X'. К полученному X' опять применяем правило и т.д. Пока правило $\mathcal{A}$ уже больше не применимо. В результате получаем некоторое конкретное графическое выражение W'. Далее к паре $(W';a)$ применяем правило $\mathcal{B}$. Правило $\mathcal{B}$ повторно применять не разрешается.

Основным результатом данной работы является следующая

ТЕОРЕМА I. Система правил вывода $(\mathcal{A},\mathcal{B})$ является асимптотически полной для унарных графических выражений.

Фактически мы докажем более сильную теорему I', которая является обобщением теоремы I для выражений с произвольным числом свободных параметров. Для этого сначала обобщим саму систему правил вывода. С этой целью введем еще несколько понятий. В дальнейшем мы будем считать, что на возможные значения свободных параметров $N_1,\dots,N_k$ выражения E наложены ограничения, т.е. фактически мы будем рассматривать пары $(E(N_1,\dots,N_k);\mathcal{P})$, где $\mathcal{P}$ - некоторый предикат, которому должны удовлетворять значения свободных параметров выражения. Иными словами, рассматриваются только такие значения $a_1,\dots,a_k$ параметров $N_1,\dots,N_k$, для которых условие $\mathcal{P}(a_1,\dots,a_k)$ выполнено. В частности, мы будем рассматривать предикаты следующего вида. Пусть $c\in\mathbb{Z}$ и пусть существует такая перестановка $i_1,\dots,i_k$ индексов $1,\dots,k$, что для значений a_{i_j} и $a_{i_{j+1}}$ параметров N_{i_j} и $N_{i_{j+1}}$ имеет место:

$$a_{j+1} - a_j > c, \quad j = 1, 2, \ldots, k-1 \qquad (1)$$

Такое условие обозначим через

$$\lambda_c(N_1, \ldots, N_k),$$

и, если данный набор $a_1, \ldots, a_k$ значений параметров $N_1, \ldots, N_k$ удовлетворяет (I), то будем писать:

$$\lambda_c(a_1, \ldots, a_k).$$

Нам также понадобится следующий предикат:

$$\lambda(N_1, \ldots, N_k) \overset{\text{def}}{\equiv} \exists c \in \mathbb{N}: \lambda_c(N_1, \ldots, N_k).$$

Назовем его упорядочением параметров. Если для свободных параметров $N_1, \ldots, N_k$ выражения E имеет место $\lambda(N_1, \ldots, N_k)$, то будем говорить, что они упорядочены.

Пусть даны выражения $E(N_1, \ldots, N_k)$ и $F(N_1, \ldots, N_k)$ и имеет место упорядочение $\lambda(N_1, \ldots, N_k)$. Будем говорить, что выражения E и F c-эквивалентны $(c \in N)$ при данном упорядочении λ, если для любых наборов значений $a_1, \ldots, a_k$, для которых выполняется $\lambda_c(a_1, \ldots, a_k)$ имеет место

$$\mathcal{VAL}(E; a_1, \ldots, a_k) = \mathcal{VAL}(F; a_1, \ldots, a_k).$$

Будем говорить, что $E(N_1, \ldots, N_k)$ и $F(N_1, \ldots, N_k)$ почти асимптотически эквивалентны при данном упорядочении λ, если существует такое c, что они c - эквивалентны при этом же упорядочении.

Пусть дано выражение $E(N_1, \ldots, N_k)$ и упорядочение $\lambda(N_1, \ldots, N_k)$. Будем говорить, что набор значений $a_1, \ldots, a_k$ параметров $N_1, \ldots, N_k$ с-выразительно

тельный ($c \in N$), если имеет место $\lambda_c(a_1, \dots, a_k)$ и для всех $i, j \in \{1, \dots, k\}$

$$\frac{a_i}{a_j} \leq c.$$

Любой набор $(\mathfrak{M}(E; a_1, \dots, a_k); c_1, \dots, c_k)$ назовем примером выражения E. Пример выражения назовем с-выразительным, если набор $a_1, \dots, a_k$ с-выразительный.

Обобщим теперь понятие асимптотической полноты системы правил вывода для выражений с произвольным числом свободных параметров. Систему S правил вывода назовем почти асимптотически полной для любой пары $(E(N_1, \dots, N_k))$, $\lambda(N_1, \dots, N_k)$, если существует такое $c_0 \in N$, что для любого $c > c_0$, имеет место следующее: какой бы с-выразительный пример $P = (\lambda; a_1, \dots, a_k)$ выражения E мы ни брали система S преобразует пример в выражение E', ассимптотически эквивалентное E при данном упорядочении λ.

Для унарных выражений почти асимптотическая полнота совпадает с асимптотической полнотой.

Обобщим теперь систему правил (A, B) так, чтобы она была почти асимптотически полной для выражений с произвольным числом параметров в упомянутом выше смысле. Правило A при этом не изменится, а вместо правила B возьмем следующее

Правило B'. Пусть дана пара $(\mathfrak{M}; a_1, \dots, a_k)$, где $\mathfrak{M}$ конкретное выражение, $a_i \in N$, $a_1, \dots, a_k$ - попарно различные. Пусть $a_1 < a_2 < \dots < a_k$. Пусть λ_i - те символы из $\mathfrak{M}$, которые принадлежат Σ. Заменяем все такие λ_i на $\underline{\lambda_i + c_i}$, где j такое, что

$$|a_j - \lambda_i| < |a_l - \lambda_i| \quad \text{для } l \neq j, \quad c_j = a_j - \lambda_i.$$

Остальные символы в $\mathfrak{M}$ не меняем.

ТЕОРЕМА I'. Система правил вывода (A,B') является почти асимптотически полной.

Очевидно, что из теоремы I' следует теорема I.

Переходим к доказательству теоремы I . Будем считать, что данное выражение E не содержит термы константной длины (т.е. термы вида $T=[T(A)]_{y=\overline{y+c_1,y+c_2}}$), так как их всегда можно заменить развертками.

В дальнейшем, кроме полной развертки $RAL(E;N_1,\dots,N_k)$ выражения E , мы будем пользоваться и различными частичными развертками, в которых некоторые подвыражения могут остаться не развернутыми.

Так, если

$$E(N)=[[\overline{y}A]_{\overline{y}=1,\overline{y}}\ \overline{y}B]_{\overline{y}=\overline{1,N}}$$

и дано $N=4$, то можно рассматривать, например, следующие частичные развертки:

$$[\overline{y}A]_{\overline{y}=1,1}\ 1B[\overline{y}A]_{\overline{y}=1,2}\ 2B[\overline{y}A]_{\overline{y}=1,3}\ 3B[\overline{y}A]_{\overline{y}=1,4}\ 4B$$

$$[\overline{y}A]_{\overline{y}=1,1}\ 1B[\overline{y}A]_{\overline{y}=1,2}\ 2B1A2A3A3B[\overline{y}A]_{\overline{y}=1,4}\ 4B$$

$$1A1B1A2A2B[\overline{y}A]_{\overline{y}=1,3}\ 3B1A2A3A4A4B.$$

Можно и некоторые термы заменить развертками не полностью. Так, например, возможны следующие частичные развертки выражения $E(N)$:

$$[\overline{y}A]_{\overline{y}=1,1}\ 1B[[\overline{y}A]_{\overline{y}=1,\overline{y}}\ \overline{y}B]_{\overline{y}=2,N}.$$

$$1A1B1A2A2B1A2A3A3B1A[\overline{y}A]_{\overline{y}=2,4}\ 4B[[\overline{y}A]_{\overline{y}=1,\overline{y}}\ \overline{y}B]_{\overline{y}=5,N}.$$

Таким же образом, как для полной развертки, можно определить понятие о б р а з а и п р о о б р а з а некоторого символа или подвыражения для частичной развертки.

Пусть $\bar{T}$ - некоторый терм выражения E. Тогда будем различать **развернутые** и не **развернутые** отображения терма $\bar{T}$ в данной развертке A. Для развернутых отображений можно определить **фактор этого отображения**.

Если F некоторое подвыражение выражения E, то через $NAL_F(E; a_{11}, \dots, a_{1k})$ будем обозначать такую развертку E, где подвыражение F остается не развернутым. Будем ее называть развёрткой до данного подвыражения F.

Пусть $X = NAL(E; \vec{a})$, где $\vec{a} = (a_{11}, \dots, a_{1k})$ - набор значений параметров $N_1, \dots, N_k$ выражения E. Рассмотрим последовательность слов (конкретных выражений) $X_0, X_1, \dots, X_N$, где $X_0 = X$, а X_{i+1} получается из X_i применением правила синтеза A. Слова $X_0, \dots, X_N$ назовем **промежуточными результатами синтеза**.

Рассмотрим некоторый промежуточный результат синтеза X_m $(m < n)$. Пусть $X_m = X_m^1 Y X_m^2$, где Y - регулярное подслово X_m, имеющее наибольшую привлекательность. Пусть $X_{m+1} = X_{m+1}^1 \bar{T}^* X_m^2$, где внешний терм $\bar{T}^*$ получен заменой подпоследовательности Y порождающим термом $\bar{T}_Y$ и максимальным расширением и смещением терма $\bar{T}_Y$ влево. Тогда подслово Y назовем **основой синтеза нулевого порядка терма** $\bar{T}^*$ и обозначим через $fond_0(\bar{T}^*)$. Фактор регулярного слова Y назовем **фактором основы синтеза нулевого порядка** и обозначим через $\varphi_0^{\bar{T}^*}$.

Рассмотрим теперь все внешние термы $\bar{T}_1^*, \dots, \bar{T}_k^*$ подслова $Y = fond_0(\bar{T}^*)$. Для каждого из них определена основа синтеза нулевого порядка: $Y_1 = fond_0(\bar{T}_1^*), \dots, Y_k = fond_0(\bar{T}_k^*)$. Множество подвыражений $\{Y_1, \dots, Y_k\}$ назовем **основой синтеза первого порядка терма** $\bar{T}^*$ и обозначим через $fond_1(\bar{T}^*)$. Минимальный из факторов слов $Y_1, \dots, Y_k$

назовем фактором основы синтеза первого порядка $\varphi_1^{\bar{T}^*}$ терма $\bar{T}^*$. Таким образом, индукцией можно определить основу синтеза n-того порядка $fond_n(\bar{T}^*)$ терма $\bar{T}^*$ и фактор основы синтеза n-того порядка $fond_n(\bar{T}^*)$, если $n \le depth(\bar{T}^*)-1$. Фактор основы синтеза $\varphi^{\bar{T}^*}$ терма $\bar{T}^*$ определим как $\min\{\varphi_n^{\bar{T}^*} / n \le depth(\bar{T}^*)-1\}$.

Часто нам будет удобнее вместо основы синтеза данного порядка $fond_n(\bar{T}^*)$ пользоваться основой синтеза данного уровня $fond^m(\bar{T}^*)$, которая определяется следующим образом:

$$fond^m(\bar{T}^*) = fond_{depth(\bar{T}^*)-m}(\bar{T}^*)$$

Основу синтеза нулевого порядка часто будем называть основой синтеза верхнего уровня.

Основу синтеза можно определить не только для терма, но также и для любого подвыражения.

В дальнейшем мы часто будем пользоваться разверткой выражения E и данных $a_1, \dots, a_n$, где развернуты все подвыражения, кроме основы синтеза данного уровня некоторого терма или подвыражения. Обозначим эту развертку через $\mathcal{RAL}_{fond^k(E_i)}(E; a_1, \dots, a_n)$.

Теперь можно сформулировать следующую лемму:

ЛЕММА I. Пусть дано выражение E и пусть $\chi = \mathcal{RAL}(E; \vec{a})$, где $\vec{a} = (a_1, \dots, a_m)$ – набор значений параметров E. Пусть χ_M – некоторый промежуточный результат синтеза, и пусть $\bar{T}^*$ – некоторый внешний терм из χ_M, $\varphi^{\bar{T}^*}$ – фактор основы синтеза этого терма. Тогда существует такое $\varphi_0 \in N$, зависящее только от E (а не от $\vec{a}$), что, если $\varphi^{\bar{T}^*} > \varphi_0$ и $\vec{a}$ с-выразительно при достаточно большом c, то E можно

преобразовать в E', симптотически эквивалентное E, такое, что существует частичная развертка A' выражения E', что

$$PAL_{\bar{J}^*}(X_M) - A',$$

где $PAL_{\bar{J}^*}(X_M)$ - развертка выражения X_M, до терма $\bar{J}^*$.

ДОКАЗАТЕЛЬСТВО. Докажем, что для всех $K \le depth(\bar{J}^*)$ имеет место следующее утверждение: выражение E можно преобразовать в выражение E_K, асимптотически эквивалентное E, такое, что для E_K существует развертка A_K, для которой имеет место:

$$A_K = PAL_{fond^K(\bar{J}^*)}(X_M),$$

где $PAL_{fond^K(\bar{J}^*)}(X_M)$ - развертка выражения X_M до основы синтеза K-того уровня терма $\bar{J}^*$.

Для $K=0$ справедливость этого утверждения очевидна. Допустим, что утверждение верно для некоторого $K<n$ и докажем, что тогда оно верно и для $K+1$.

Рассмотрим некоторый внешний терм $\bar{J}_i^*$ выражения $PAL_{fond^{K+1}(\bar{J}^*)}(X_M)$ и образ $A^{\bar{J}_i^*}$ этого терма в развертке $PAL_{fond^K(\bar{J}^*)}(X_M)$. Заметим, что подслово $A^{\bar{J}_i^*}$ является регулярным. Пусть его фактор равен f_i^*, а период p_i^*. Тогда $A^{\bar{J}_i^*}$ содержит по крайней мере f_i^* различных чисел. Поэтому, если f_i^* достаточно большое, то $A^{\bar{J}_i^*}$ должно содержать подслово $A^{\bar{J}_i'}$ являющегося развернутым образом некоторого терма $\bar{J}_i'$ выражения E_K, имеющим фактор f_i' такой, что $f_i' \ge f_i^* - Ac_{max}$. Индукцией по глубине терма $\bar{J}_i'$ легко доказать, что $A^{\bar{J}_i'}$ должен содержать регулярную подпоследовательность $A^{\bar{J}_i}$ с фактором $f_i = f_i'/c$, где c - некоторая константа $(c \ge 1)$ зависящая только от $\bar{J}_i'$. Пусть период $A^{\bar{J}_i}$ равен p_i.

В [2] доказана следующая лемма.

ЛЕММА 2. Пусть подпоследовательность X - (ρ_1, f_1) - регулярная, а подпоследовательность Y - (ρ_2, f_2) - регулярная. Пусть X и Y содержат общую подпоследовательность Z. Тогда, если $|Z| > \max(\rho_1, \rho_2) + 4\min(\rho_1, \rho_2)$, то $\rho_1 = \rho_2$.

Из леммы легко получить

СЛЕДСТВИЕ I. Пусть подпоследовательность X - (ρ_1, f_1) - регулярная, а подпоследовательность Y (ρ_2, f_2) - регулярная. Пусть X и Y содержат общую подпоследовательность Z. Пусть $\rho_1 \neq \rho_2$. Тогда

$$\max(\rho_1, \rho_2) > |Z| - 5\min(\rho_1, \rho_2).$$

Из следствия I вытекает, что для периодов ρ_1^* и ρ_1 последовательностей $A^{\bar{\pi}_1^*}$ и $A^{\bar{\pi}_1}$, соответственно, имеет место либо

$$\rho_1 = \rho_1^*,$$

либо

$$\rho_1^* > f_1 \cdot \rho_1 - 5 \cdot \rho_1 \geq c' \cdot f_1' \cdot \rho_1$$

Допустим, что $\rho_1 \neq \rho_1^*$.

Так как $f_1' \geq f_1 - 2c_{max} > \varphi_0 - 2c_{max}$, то, если φ_0 достаточно большое, $\rho_1^* > \rho_1$. Пусть π_1^* и π_1 - привлекательности последовательностей $A^{\bar{\pi}_1^*}$ и $A^{\bar{\pi}_1}$, соответственно. Так как из правила синтеза следует, что $\pi_1^* > \pi_1$, то $f_1^* > f_1$. Поэтому

$$\rho_1^* > c' \cdot f_1' \cdot \rho_1 \geq c'' \cdot f_1^*,$$

где $c'' = c' \cdot \rho_1$ зависит только от E_K. Тогда

$$\pi_1^* = \frac{f_1^*}{\rho_1^*} < \frac{f_1^*}{c'' \cdot f_1^*} = const$$

$$\pi_1 = \frac{f_1}{\rho_1} > c \cdot f_1^* \geq c \cdot \varphi_0.$$

Если γ_0 достаточно большое, то $\bar{T}_1 > \bar{T}_1^*$, а это является противоречием, поскольку, в таком случае следуя правилу $\mathcal{A}$, мы должны были заменять порождающим термом последовательность $\mathcal{A}^{\bar{T}_1}$.

Таким образом, мы доказали, что $\rho_1 = \rho_1^*$.

Докажем теперь, что тогда тело терма $\bar{T}_1^*$ совпадает с телом терма $\bar{T}_1$ с точностью до смещения, т.е. тело $\bar{T}_1^*(y)$ можно представить в форме:

$$\bar{T}_1^*(y) = \bar{T}_1'(y)\bar{T}_1'(y+\delta),$$

где $\bar{T}_1'(y)\bar{T}_1''(y) = \bar{T}_1(y)$, $\delta = \pm 1$.

Для этого достаточно заметить, что любой развернутый образ любого подтерма $\bar{T}_1'$ терма $\bar{T}_1$ имеет фактор меньше некоторой константы γ_0, зависящей только от самого выражения. Действительно, из свойства с-выразительности набора $a_1, \dots, a_k$, следует, что для всех a_i, a_j $(i,j \in \{1,\dots,k\})$ имеет место $a_i < c \cdot a_j$. Если теперь фактор $\bar{T}_1$ не был бы ограничен, то при достаточно больших $a_1, \dots, a_k$ образ $\bar{T}_1'$ содержал бы регулярную подпоследовательность с привлекательностью больше привлекательности $\mathcal{A}^{\bar{T}_1}$, что противоречило бы применению правила $\mathcal{A}$.

Таким образом, мы доказали, что существует такая развертка A_k' выражения E_k, в которой каждому внешнему терму $\bar{T}_i^*$ выражения $\mathcal{KHL}_{\mathrm{cond}^{k+1}(\bar{T}^*)}(x_m)$ соответствует некоторый терм $\bar{T}_i$, который совпадает с $\bar{T}_i^*$ с точностью до смещения. Очевидно, если в A_k' все термы теперь максимально расширить и сместить в таком же порядке в каком они были просинтезированы в $\mathrm{cond}^{k+1}(\bar{T}^*)$, то для полученного выражения A_k'' будет иметь место

$$A_k'' = \mathcal{KHL}_{\mathrm{cond}^{k+1}(\bar{T}^*)}(x_m)$$

Остается показать, что E_k можно преобразовать в E_{k+1} такое, что существует развертка A_{k+1} выражения E_{k+1}, для которой имеет место

$$A_{k+1} = A'_k .$$

Алгоритм таких преобразований фактически является обобщением алгоритма Π_1 из [2]. Опишем сначала идею алгоритма.

В начале выражение E_k преобразуется в некоторое выражение $\widetilde{E}_k$, асимптотически эквивалентное E_k, термы которой обладают описанным ниже свойством P-отделимости и Q-изолированности $(P,Q \in N)$. Из алгоритма преобразования будет следовать, что эти преобразования фактически достаточно провести только в самом начале, т.е. при $k=0$, так как, если P и Q достаточно большие, то свойство P-отделимости и Q-изолированности при преобразованиях сохраняется.

Дальше будут описаны правила смещения и расширения термов выражения. Из построения этих правил будет следовать, что термом выражения $\widetilde{E}_k$ можно приписать различные номера так, чтобы имело место следующее свойство: если при преобразовании A'_k какие-то образы термов $\bar{J}_i$ и $\bar{J}_j$ выражения $\widetilde{E}_k$ "контактируются", то меньший номер имеет тот терм, образ которого в $RAL_{forall}^{k+1}(\bar{J}^*)(X_m)$ "просинтезирован" раньше.

После этого применим к термам выражения упомянутые правила расширения и смещения в порядке возрастания номеров. Будет доказано, что полученное таким образом выражение E_{k+1} асимптотически эквивалентное E_k, и для него существует частичная развертка A_{k+1} такая, что

$$A_{k+1} = RAL_{forall}^{k+1}(\bar{J}^*)(X_m) .$$

Переходим теперь к более точному описанию алгоритма преобразования. Сначала опишем более точное предварительное преобразование E_K в $\bar{E}_K$.

Пусть дано выражение E, пусть $N_1,\dots,N_l$ – свободные параметры E, и пусть дано $\lambda'(N_1,\dots,N_l)$. Пусть $\bar{T}$ – некоторый терм выражения E, и $Y_1,\dots,Y_k$ – свободные параметры тела T терма $\bar{T}$. Будем говорить, что терм $\bar{T}$ является P – **отделимым**, если существует упорядочение λ и число $c\in N$, такие что для любых значений $a_1,\dots,a_l$ параметров $N_1,\dots,N_l$ для которых имеет место $\lambda_c'(a_1,\dots,a_l)$ выполняется $\lambda_{P\cdot c^T_{max}}(\beta_1,\dots,\gamma_k)$, где c^T_{max} – максимальное абсолютное значение констант из T. Будем говорить, что **выражение** E P – **отделимое**, если все его термы P-отделимые.

Пусть $\bar{T}=[T/M]_{Y=\alpha,\beta}$. Пусть $\alpha=\overline{Y_1+c_1}$, $\beta=\overline{Y_2+c_2}$ $(c_1,c_2\in Z)$. Будем писать $\alpha \underset{c}{\leftarrow} \beta$, если имеет место $\lambda_{c-c_1+c_2}(Y_1,Y_2)$. Назовем **терм** $\bar{T}$ **односторонним**, если $\alpha \underset{c}{\leftarrow} \beta$ или $\beta \underset{c}{\leftarrow} \alpha$. Будем говорить, что **направление** терма $\bar{T}$ равно $+1$, если $\alpha \underset{c}{\rightarrow} \beta$, и равно -1, если $\beta \underset{c}{\rightarrow} \alpha$.

Очевидно, что если выражение E P-отделимое для достаточно большого P, то все его термы односторонние.

Определим понятие левого и правого подвыражения.

Пусть даны выражения $F(N_1,\dots,N_k)$ и $G(N_1,\dots,N_k)$ и пусть дано, что имеет место упорядочение $\lambda(N_1,\dots,N_k)$. Будем говорить, что G является **левым (правым) подвыражением** F, если существует такое $c\in N$, что для любого набора $(a_1,\dots,a_k)$ значений параметров $N_1,\dots,N_k$, для которого выполняется $\lambda_c(a_1,\dots,a_k)$ имеет место

$$F(a_1,\dots,a_k)\underset{L}{\sqsubset} G(a_1,\dots,a_k)$$

(соответственно, $F(a_1,\dots,a_k)\underset{R}{\sqsubset} G(a_1,\dots,a_k)$).

В этом случае будем писать $F \underset{L}{\subseteq} G$ (соответственно, $F \underset{R}{\subseteq} G$).

Пусть теперь $\bar{T} = [T(Y)]_{\gamma+\alpha,\beta}$ - односторонний терм выражения E . Будем говорить, что терм $\bar{T}$ Q-и з о л и р о в а н в E , если он входит в подвыражение $A\bar{T}B$, где

$[T(Y)]_{\gamma=\alpha-\delta Q,\alpha-\delta} \underset{R}{\subseteq} A$ и $[T(Y)]_{\gamma=\beta+\delta,\beta+\delta Q} \underset{L}{\subseteq} B$.

Любое подвыражение A' (B') , для которого выполняется $A' \underset{R}{\subseteq} [T(Y)]_{\gamma=\alpha-\delta n,\alpha-\delta}$ (соответственно, $B' \underset{L}{\subseteq} [T(Y)]_{\gamma=\beta+\delta,\beta+\delta n}$), $n \le Q$, назовем л е в ы м (п р а в ы м) п р о д о л ж е н и е м т е р м а $\bar{T}$ д л и н ы n

Будем говорить, что в ы р а ж е н и е E Q-и з о л и р о в а н н о , если все его термы Q-изолированы.

В [2] доказана следующая

ЛЕММА 3. Пусть дано $E(N_1,\dots,N_k)$ и $\lambda_c(N_1,\dots,N_k)$ Тогда, если c достаточно большое, то существует алгоритм, который для любых P и Q преобразует выражение E в P-отделимое и Q-изолированное выражение $\tilde{E}$, асимптотически эквивалентное E .

Фактически, мы будем использовать выражения, в которых P-отделимы и Q-изолированы только те термы, которые имеют уровень больше некоторого $\kappa \in N$ $(\kappa \le depth(E))$, где под уровнем терма понимается уровень вложения скобок терма.

Определим теперь правила смещения и расширения для термов выражения. Для этого сначала определим левую и правую разницу двух выражений F и G . Пусть $\Phi(N_1,\dots,N_k) \underset{L}{\subseteq} G(N_1,\dots,N_k)$ $(F(N_1,\dots,N_k) \underset{R}{\subseteq} G(N_1,\dots,N_k))$ и пусть дано $\lambda(N_1,\dots,N_k)$. Тогда существует такое выражение $H(N_1,\dots,N_k)$, что для любых значений $a_1,\dots,a_k$ таких, что $\lambda_c(a_1,\dots,a_k)$ для достаточно большого $c \in N$, имеет место

$VAL(F; a_1, \dots, a_n) VAL(H; a_1, \dots, a_n) = VAL(G; a_1, \dots, a_n)$

(соответственно, $VAL(H; a_1, \dots, a_n) VAL(F; a_1, \dots, a_n) = VAL(G; a_1, \dots, a_n)$).

Выражение H будем называть **левой (правой) разницей выражений** F и G.

Построим теперь алгоритм Δ_L (Δ_R), который выясняет, верно ли, что $F \underset{L}{\subseteq} G$ (соответственно, $F \underset{R}{\subseteq} G$) и, если да, то находит левую (соответственно, правую), разницу этих выражений. Результат работы алгоритма обозначим через $\Delta_L(G, F)$ (соответственно, $\Delta_R(G; F)$) и назовем **канонической разницей** выражений F и G.

Допустим, что для некоторого C-выразительного $\vec{a} = (a_1, \dots, a_n)$ при достаточно большом C имеет место $F(\vec{a}) \underset{L}{\subseteq} G(\vec{a})$ (соответственно, $F(\vec{a}) \underset{R}{\subseteq} G(\vec{a})$). Запустим на выражениях $F(\vec{a})$ и $G(\vec{a})$ алгоритм нахождения канонической разницы для конкретных выражений Δ_L^* (соответственно, Δ_R^*). Как было сказано выше, алгоритм Δ_L^* (соответственно, Δ_R^*) постепенно преобразует выражение $G(\vec{a})$ в $G'(\vec{a})$. Параллельно будем преобразовывать таким же образом и выражение $G(\vec{x})$ в $G'(\vec{x})$. Полученное $G'(\vec{x})$ можно будет записать в форме $G'(\vec{x}) = -G_1'(\vec{x}) G_2'(\vec{x})$, где $G_1'(\vec{x})$ (соответственно, $G_2'(\vec{x})$) обладает свойством: $VAL(G_1'; \vec{a}) = VAL(F; \vec{a})$ (соответственно, $VAL(G_2'; \vec{a}) = VAL(F; \vec{a})$). Так как $\vec{a}$ C-выразительный, то, если C – достаточно большое, из теоремы I работы [2] следует, что $G_1' \sim F$ (соответственно, $G_2' \sim F$). Далее, по определению левой (соответственно, правой) разницы G_1' (соответственно, G_2') является левой (соответственно, правой) разницей. Назовем его левой (соответственно, правой) канонической разницей выражений F и G.

Сформулируем некоторые свойства канонической разницы.

СВОЙСТВО 1. Пусть даны выражения $F(N_1,\dots,N_k)$ и $G(N_1,\dots,N_k)$ и пусть при упорядочении $\lambda(N_1,\dots,N_k)$ имеет место $F \underset{L}{\subseteq} G$ (соответственно, $F \underset{R}{\subseteq} G$). Если для $a_1,\dots,a_k$ имеет место $\lambda_c(a_1,\dots,a_k)$ при достаточно большом c, то

$$\Delta_L^*(F(a_1,\dots,a_k), G(a_1,\dots,a_k)) = \Delta_L(F,G)(a_1,\dots,a_k),$$

(соответственно, $\Delta_R^*(F(a_1,\dots,a_k), G(a_1,\dots,a_k)) = \Delta_R(F,G)(a_1,\dots,a_k)$).

СВОЙСТВО 2. Пусть даны выражения F и G и пусть $F \underset{L}{\subseteq} G$ ($F \underset{R}{\subseteq} G$). Тогда, если F и G P-отделимые и Q-изолированные при достаточно больших P и Q, то G имеет форму $G = G_1 G_2$, где $G_1 \sim F$ (соответственно, $G_2 \sim F$).

Пусть теперь дано выражение $E = E_1 \bar{T} E_2$, где $\bar{T}$ внешний терм. Пусть для всех свободных параметров $N_1,\dots,N_k$ выражения E дано $\lambda(N_1,\dots,N_k)$. Тогда для терма $\bar{T}$ можно определить правила расширения и смещения таким же образом, как и для конкретного выражения, только вместо разниц Δ_L^* и Δ_R^* необходимо брать разницы Δ_L и Δ_R. Из свойства I и этого определения следует:

СВОЙСТВО 3. Пусть $E = E_1 \bar{T}_1 \dots E_k \bar{T}_k E_{k+1}$, где $\bar{T}_1,\dots,\bar{T}_k$ – некоторые внешние термы выражения E. Пусть для всех свободных параметров E имеет место $\lambda(N_1,\dots,N_k)$. Через $\Pi_{(i_1,\dots,i_k)}(E)$ обозначим выражение, которое получается из E, если к термам $\bar{T}_{i_1},\dots,\bar{T}_{i_k}$ применить правила максимального расширения и смещения в указанном порядке. Тогда, если $a_1,\dots,a_k$ удовлетворяют $\lambda_c(a_1,\dots,a_k)$ при достаточно большом c, то

$$\Pi_{(i_1,\dots,i_k)}(E(a_1,\dots,a_k)) = (\Pi_{(i_1,\dots,i_k)}(E))(a_1,\dots,a_k).$$

Как уже отмечалось, для преобразования E некоторые его термы в определенном порядке перенумеруем и затем к ним применим правила преобразования (т.е. расширения и смещения). Эти термы будем называть **основными** термами данного преобразования, а те термы, которые не имеют номеров и не являются подтермами основных термов, назовем **термами высшего порядка**. При преобразовании E_K также будет необходимо смещать термы высшего порядка. В данном случае под смещением будем понимать не только смещение влево, но и смещение вправо (очевидно, смещение вправо можно определить таким же образом как и влево.)

Пусть даны выражения $F(N_1,\dots,N_k)$ и $G(N_1,\dots,N_k)$, и пусть имеет место упорядочение $\lambda(N_1,\dots,N_k)$. . Будем говорить, что F является **истинным подвыражением** G (запишем это $F \sqsubset G$), если существуют такие (возможно и пустые) выражения $H_1(N_1,\dots,N_k)$ и $H_2(N_1,\dots,N_k)$, что для любого набора $\vec{a}=(a_1,\dots,a_k)$, для которого имеет место $\lambda_c(a_1,\dots,a_k)$, верно:

$$VAL(G;\vec{a}) = VAL(H_1;\vec{a}) \cdot VAL(F;\vec{a}) \cdot VAL(H_2;\vec{a}).$$

Будем говорить, что F и G **пересекаются**, если они имеют непустое истинное подвыражение.

Рассмотрим выражение $F=F_1\bar{T}F_2F_3F_4$ $(F=F_1F_3F_2\bar{T}F_4)$, где $\bar{T}$ – некоторый внешний терм а F_1,F_2,F_3 и F_4 – любые подслова. Применим правила преобразования к терму $\bar{T}$. В результате получим выражение $F'=F_1'\bar{T}'F_2'$ (соответственно, $F'=F_2'\bar{T}'F_1'$), где $\bar{T}'$ – терм, полученный из $\bar{T}$ в результате преобразований. Будем говорить, что терм $\bar{T}$ при преобразованиях **поглощает** подвыражение F_3, если $F_1F_3 \sqsubset \bar{T}$ (соответственно, $F_3F_4 \sqsubset \bar{T}'$). Будем говорить, что терм $\bar{T}$ **достигает** подвыражения F_3, если $F_3 \sqsubset \bar{T}'$, а F_3 и $\bar{T}$ имеет непустое пересечение.

Рассмотрим теперь выражение $F = F_1 \bar{\bar{J}}_1 F_2 \bar{\bar{J}}_2 F_3$ $(F = F_3 \bar{\bar{J}}_2 F_2 \bar{\bar{J}}_1 F_1)$. Пусть терм $\bar{\bar{J}}_1$ является основным термом в данном преобразовании, а $\bar{\bar{J}}_2$ является термом высшего порядка. Пусть при преобразованиях терм $\bar{\bar{J}}_1$ достигает терма $\bar{\bar{J}}_2$. Тогда имеются две возможности: либо терм $\bar{\bar{J}}_1$ поглощает терм $\bar{\bar{J}}_2$, либо нет. В каждом из этих случаев выражение F будем преобразовывать по разному. В первом случае терм $\bar{\bar{J}}_1$ будем просто расширять, пока это возможно или пока он не достигнет терма высшего порядка, которого он не поглощает. Во втором случае терм $\bar{\bar{J}}_2$ будем смещать вправо (соответственно, влево) так, чтобы терм $\bar{\bar{J}}_1$ его не достигал. Докажем, что если F Q-изолированное для достаточно большого Q, и для параметров $N_1, \dots, N_k$ выражения F имеет место $\lambda_c(N_1, \dots, N_k)$, то это возможно.

Во-первых, заметим, что из свойства 2 следует, что этот случай возможен только, если $\bar{\bar{J}}_1$ принадлежит продолжению терма $\bar{\bar{J}}_2$. Докажем, что в этом случае, терм $\bar{\bar{J}}_2$ достаточно сместить в пределах его продолжения первого порядка, т.е. что в наихудшем случае подвыражение $[\bar{\bar{J}}_1(J)]_{J=A+\beta} F_3$ (соответственно, $F_3 [\bar{\bar{J}}_1(J)]_{J=A+\beta}$) достаточно заменить на $\bar{\bar{J}}_1(A)[\bar{\bar{J}}_1(J)]_{J=A+\delta,\beta+\delta} A_L(F_3 \bar{\bar{J}}_2(\beta-\delta))$ (соответственно, $A_R(F_3 \bar{\bar{J}}_2(A-\delta))[\bar{\bar{J}}_1(J)]_{J=A-\delta,\beta-\delta} \bar{\bar{J}}_1(\beta)$), где δ направление терма $\bar{\bar{J}}_2$. Утверждение легко вытекает из следующей леммы.

ЛЕММА 4. Пусть дано выражение $F_1 = F_1' \bar{\bar{J}}_1 F_1''$, где $\bar{\bar{J}}_1$ некоторый внешний терм. Пусть для свободных параметров выражения имеет место условие $\lambda_c(N_1, \dots, N_k)$. Пусть выражение $F_2 = F_2' \bar{\bar{J}}_2 F_2''$ получено из F_1 подстановкой выражения $N_i + \delta$ вместо некоторого параметра N_i, где $\delta = 1$ или $\delta = -1$, т.е. $F_2' = F_1'(N_i | N_i + \delta)$ $\bar{\bar{J}}_2 = \bar{\bar{J}}_1(N_i | N_i + \delta)$ и $F_2'' = F_1''(N_i | N_i + \delta)$. Рассмотрим выражение $F_1(N_i | F_1(N_i + \delta)) = F_1 F_2 =$ $= F_1' \bar{\bar{J}}_1 F_1'' F_1' \bar{\bar{J}}_2 F_2''$. Тогда, если c достаточно большое, то при расширении и смещении термы $\bar{\bar{J}}_1$ и $\bar{\bar{J}}_2$ не дости-

гают друг друга.

Эта лемма является обобщением леммы 6 из [2] и доказывать её здесь мы не будем.

Таким образом, мы показали как преобразовать произвольный внешний терм выражения. Пусть теперь данный терм $\bar{T}$ не является внешним в выражении $\bar{E}_K$, а является подтермом какого-то терма $\bar{T}'$. Тогда $\bar{T}'$ можно представить в форме $\bar{T}' = [H_1(\gamma)\bar{T}(\gamma)H_2(\gamma)]_{\gamma\cdot\Delta\rho}$. В этом случае возможно, что в некотором образе $T'(c)$ тело терма $\bar{T}'$, где $c \in \mathcal{X}$ - некоторая из возможных значений параметра γ, образ $\bar{T}(c)$ терма $\bar{T}(\gamma)$ при преобразовании достигает подвыражение $T'(c,\delta)$ (или $T'(c\cdot\delta)$), где δ - направление терма $\bar{T}'$. Иными словами говоря возможно, что образ $\bar{T}(c)$ достигает следующий образ тела T' терма $\bar{T}'$. В этом случае выражение $\bar{E}_K$ будем преобразовывать следующим образом. Так как выражение G - изолированное, то терм $\bar{T}'$ входит в подвыражение $F[T'(\gamma)]_{\gamma\cdot\Delta\rho}G$, где $T'(\lambda\cdot\delta) \sqsubseteq F$ и $T'(\rho\cdot\delta) \sqsubseteq G$. Рассмотрим выражение $T'(\rho)G$ (соответственно, $FT'(\lambda)$) и применим к терму $\bar{T}(\rho)$ (соответственно, $\bar{T}(\lambda)$) правила расширения и смещения. Пусть x_i последний символ подвыражения G (соответственно, F) которого достигает терм $\bar{T}(\rho)$ (соответственно, $\bar{T}(\lambda)$), а x_{i+1} следующий символ за ним (из леммы 4 вытекает, что такие x_i и x_{i+1} существуют). Тогда смещаем в выражении $\bar{E}_K$ терм $\bar{T}'$ до символа соответствующего символу x_{i+1}. После этого применяем правила расширения и смещения к терму $\bar{T}(\gamma)$ выражения $\bar{E}_K$. При этом очевидно $\bar{T}(\gamma)$ не достигает квадратных скобок терма $\bar{T}'$, а из свойства 3 следует, что образы преобразованного терма $\bar{T}(\gamma)$ совпадают с преобразованными образами $\bar{T}(c)$ для всех возможных значений c параметра γ.

Остается показать, что описанное смещение терма высшего порядка не противоречит смещению этого же терма

при преобразовании подвыражений, находящихся левее или правее данного терма. Это прямо вытекает из следующей леммы.

ЛЕММА 5. Пусть $\bar{T}' = [T'(\gamma)]_{\gamma_0=\alpha,\beta}$ P - отделимый терм некоторого выражения E при достаточно большом P. Рассмотрим выражения:

$$F_1 = T'(\gamma_0)\,T'(\gamma_0+\delta)$$
$$F_2 = T'(\alpha-\delta)\,T'(\alpha)$$
$$F_3 = T'(\beta)\,T'(\beta+\delta)$$

Пусть $\bar{T}$ - произвольный внешний подтерм терма $\bar{T}'$. Тогда, если при расширении и сокращении терм $\bar{T}$ достигает (не достигает) некоторый символ в каком-то из выражений F_1, F_2 или F_3, то терм $\bar{T}$ достигает (соответственно, не достигает) соответствующий символ в любом другом из выражений F_1, F_2 или F_3.

Справедливость леммы легко вытекает из свойства 3 правил преобразования.

Таким образом, для завершения доказательства леммы I остается показать, что термы выражения $\tilde{E}_K$ можно упорядочить так, как было отмечено выше, т.е. если при преобразованиях развертки A'_K какой-то терм $\bar{\bar{T}}_1'$, являющийся образом терма $\bar{\bar{T}}_1$, достигает терм $\bar{\bar{T}}_2'$, являющийся образом терма $\bar{\bar{T}}_2$, то наименьший номер имеет тот из термов $\bar{\bar{T}}_1$ и $\bar{\bar{T}}_2$, соответствующий образ которого в $\mathfrak{M}_{load^{k+1}(T^{*})}(\gamma_0)$ при синтезе получен первым.

Для этого вспомним, что в P -отделимом и G - изолированном выражении при достаточно больших P и Q терм $\bar{T}_1$, достигающий терма $\bar{T}_2$, может его не поглощать только в том случае, если $\bar{T}_1$ принадлежит продолжению $\bar{T}_2$. Тогда существование упомянутого выше упорядочения термов легко вытекает из следующей леммы.

ЛЕММА 6. Пусть дано выражение $F = F_1 \bar{T}_1 \dots F_k \bar{T}_k F_{k+1}$, где $\bar{T}_1, \dots, \bar{T}_k$ – внешние термы. Пусть $N_1, \dots, N_k$ – свободные параметры выражения F и пусть имеет место упорядочение $\lambda(N_1, \dots, N_k)$. Рассмотрим развертки

$$A = VAL^1(F; \vec{a}) \quad \text{и} \quad B = VAL^1(F; \vec{b}),$$

где $\vec{a} = (a_1, \dots, a_k)$, $\vec{b} = (b_1, \dots, b_k)$. Обозначим привлекательность регулярного подслова, соответствующего образу подтерма $\bar{T}_i$ в A через $atr_i(\vec{a})$, а в B через $atr_i(\vec{b})$. Тогда существует такое $c \in N$, что для любых $\vec{a} = (a_1, \dots, a_k)$ и $\vec{b} = (b_1, \dots, b_k)$, для которых выполняется $\lambda_c(\vec{a})$ и $\lambda_c(\vec{b})$, имеет место:

$$atr_i(\vec{a}) > atr_j(\vec{a}) \equiv atr_i(\vec{b}) > atr_j(\vec{b}),$$

для любых подтермов $\bar{T}_i$ и $\bar{T}_j$.

Доказательство леммы очевидно.

Этим доказательство леммы I завершено. Пусть теперь A и B – два конкретных выражения таких, что $CAL(A) = CAL(B)$. Пусть $A = A_1 \bar{T}_A A_2$ и $B = B_1 \bar{T}_B B_2$, где $\bar{T}_A$ и $\bar{T}_B$ – внешние термы. Будем говорить, что терм $\bar{T}_A$ совпадает с термом $\bar{T}_B$, если $T_A = T_B$, $CAL(A_1) = CAL(B_1)$ и $CAL(A_2) = CAL(B_2)$.

Пусть E – выражение, $\vec{a}$ – некоторый набор значений этого выражения и A – некоторая частичная развертка выражения E при данном $\vec{a}$. Пусть $\bar{T}_i$ – некоторый терм из A. Через $\bar{T}_i^0$ обозначим прообраз терма $\bar{T}_i$ в выражении E. Пусть далее в некоторое конкретное выражение такое, что $CAL(A) = CAL(B)$. Будем говорить, что развертку A можно эквивалентно преобразовать в B, если для каждого внешнего терма $\bar{T}_i$ развертки A выражение E можно преобразовать в такое асимптотически эквивалентное E', что терм

$\bar{T}_i^{\circ}$ преобразуется в терм $\bar{T}_i^{\circ 1}$ такой, что его образ, соответствующий образу $\bar{T}_i$, совпадает с некоторым термом из $\mathcal{B}$.

Теперь можно сформулировать важное следствие, вытекающее из леммы I.

СЛЕДСТВИЕ 2. Пусть E - выражение $\bar{a}=(a_0,\dots,a_n)$ некоторый c -выразительный набор значений параметров E, $X=\mathcal{AAL}(E;\bar{a})$ - развертка E, а X_N - промежуточный результат синтеза на X такой, что при синтезе $X_0 \to X_1 \to \dots \to X_N$ все привлекательности заменённых регулярных подслов больше некоторого $\varphi_0 \in \mathcal{N}$. Тогда, если φ_0 и c достаточно большие, то существует такая частичная развертка A выражения E, что A можно эквивалентно преобразовать в X_N.

Для завершения доказательства теоремы I, кроме следствия 2, нам понадобится ещё следующая

ЛЕММА 7. Пусть дано выражение $F=\bar{T}=[T(Y_0)]_{\bar{a}=\bar{a},\rho}$, где $\bar{T}$ - P -отделимый терм, и пусть $N_1,\dots,N_k$ - свободные параметры выражения F. Пусть далее, $X=\mathcal{AAL}(\bar{T};\bar{a})$, где $\bar{a}=(a_1,\dots,a_k)$ - набор значений параметров $N_1,\dots,N_k$, и пусть X_M - некоторый промежуточный результат синтеза на X такой, что выражение $A=\mathcal{AAL}'(\bar{T};\bar{a})$ можно эквивалентно преобразовать в A' так, чтобы $A'=X_M$. Тогда, если P - достаточно большое, то и само выражение F можно преобразовать в асимптотически эквивалентное выражение $F'=F_1'\bar{T}'F_2'$ такое, что

$$F_1'(\bar{a})\,\mathcal{AAL}'(\bar{T};\bar{a})\,F_2'(\bar{a})=X_N,$$

где $F_1' \underset{L}{\subset} T(L)$ и $F_2' \underset{R}{\subset} T(\rho)$.

Лемма доказывается индукцией по уровням терма $\bar{T}$, т.е. доказывается, что для каждого $n < depth(\bar{T})$ выражение $F=\bar{T}$ можно преобразовать в $F^n=F_1^n\bar{T}^{(n)}F_2^n$ так, чтобы существовала развертка $\mathcal{A}^n$ выражения F^n, для которой имеет место:

$A^n = RAL_{found^n(X_m)}(X_m)$

Справедливость этого утверждения для $n=0$ очевидна, а индуктивный переход фактически является повторением второй части доказательства леммы I.

Пусть теперь $X = RAL(E;\vec{a})$ и $depth(E) = n$. Рассмотрим промежуточные результаты синтеза на X: $X_1, X_2, \dots, X_p$. Пользуясь леммой 7 покажем, что если $\vec{a}$ достаточно выразительное, то существует такой $m \le p$, что результат синтеза X_l содержит терм $\bar{T}^*$, имеющий глубину n и произвольно большой фактор основы синтеза γ_0.

Для этого рассмотрим промежуточный результат синтеза X_k такой, что в процессе синтеза $X \to X_1 \to \dots X_k$ все замененные регулярные подпоследовательности имеют привлекательность больше или равно γ_0, а в X_k такой регулярной подпоследовательности больше нет. Из следствия 2 вытекает, что для выражения E существует развертка A такая, что A можно эквивалентно преобразовать в X_k.

Допустим теперь, что X_k не содержит терм глубины n. Имея ввиду, что X_k не содержит регулярного подслова с привлекательностью большей или равной $\cdot \gamma_0$, то, если $\vec{a}$ c-выразительно при достаточно большим c от противного легко доказать, что X_k содержит терм $\bar{T} = [T[c_0, x_1, \dots, x_k]]_{x_i = x_i + c_i, y_i = c_i}$, обладающий следующим свойством:

а) существуют значения $b_{11}, \dots, b_{1k}$ параметров $y_1, \dots, y_k$ такие, что $(b_j + c_j) - (b_i + c_i) > \gamma_0$ и существует подслово

$$B = T(b_i + c_i, b_{1i}, \dots, b_{ki}) T(b_i + c_i + \delta, b_{1i}, \dots, b_{ki}) \dots T(b_j - c_j, b_{1j}, \dots, b_{kj})$$

развертки A, которое можно эквивалентно преобразовать в B' так, чтобы имело место:

$$X_h = X_h^1 \mathcal{B} X_h^2$$

где X_h^1 и X_h^2 - некоторые подвыражения X_h ;

б) выражение $F - \bar{F}$ не возможно преобразовать в асимптотически эквивалентное F' такое, чтобы при значениях $b_1, \ldots, b_k$ параметров F' существовала развертка C выражения F' такая, что

$$X_h = X_h'' C X_h'^2.$$

Но существование такого терма противоречит лемме 7. Таким образом, имея ввиду, что фактор регулярной подпоследовательности больше или равен её привлекательности, мы доказали, что если глубина выражения E равна n, то существует такой промежуточный результат синтеза что в X_m существует терм $\bar{F}^*$ глубины $n - depth(E)$ с фактором основы синтеза больше или равно γ_0.

Теперь по лемме I можно преобразовать в E' такое, что

$$E' = E_1' \bar{F} E_2',$$

где

$$X_m = X_{m1}^1 \bar{F}^* X_m^2,$$
$$MAL(X_m^1) = MAL(E_1'; \vec{a}),$$
$$MAL(X_m^2) = MAL(E_2'; \vec{a}),$$
$$\bar{F}^* = \bar{F}(\vec{a}).$$

Рассмотрим дальше термы максимальной глубины из E_1' и E_2'. К ним можно применить аналогичные рассуждения. Таким образом, индукцией получаем, что E можно преобразовать в E^* асимптотически эквивалентное E, так, чтобы

$$X_m = E^*(\vec{a}).$$

Легко видеть, что если $\vec{a}$ достаточно выразительное, то все дальнейшие применения правила синтеза A к $X_m = E^*(\vec{a})$ можно применить и к самому E^*, сохраняя при этом асимптотическую эквивалентность, т.е. для любого $z > m$ существует такое $E^{**} \sim E^*$, что

$$X_z = E^{**}(\vec{a}).$$

Теперь, так как $\vec{a} = (a_1, \ldots, a_k)$ достаточно выразительное, то очевидно, после применения правила синтеза A' из $X_z = E^{**}(\vec{a})$ получим выражение E^{**}, которое как доказано, асимптотически эквивалентно E.

Теорема I' доказана.

ЛИТЕРАТУРА

I. Барздинь Я.М. Некоторые правила индуктивного вывода и их применение // Семиотика и информатика. - М.: ВИНИТИ, 1982. - Вып.9 - С.59-89.

2. Бразма А.Н. Разрешимость проблемы эквивалентности для графических выражений // Теория алгоритмов и программ - Рига: ЛГУ им.П.Стучки, 1986. - С.103-156.

3. Этмане И.Э. Об одной формализации понятия многоточия // Семиотика и информатика. - М.: ВИНИТИ, 1986. - Вып. 27 - С.29-43.

ABSTRACTS

On probabilistic and deterministic Turing machines with input and output

R.Freivalds

Turing machines are considered which begin the processing input by reading the input letter-by-letter at linear speed. After reading the last input symbol additional time for the processing is allowed. A language is allowed. A language D is constructed such that 1) for arbitrary $\varepsilon > 0$ there is a probabilistic Turing machine recognizing D in real time with probability 1/3, and 2) every deterministic Turing machine needs at least $n^2/\log n$ time to recognize D .

A counterpart of the Perikh's theorem for languages accepted by nondeterministic one-way multitape finite automata

D.Geidmanis

Let n -tape finite automaton process n -tuples of words in an l -letter alphabet. Let $\psi^i(z_1,\dots,z_l)$ be an l -tuple of integers denoting the number of various symbols in the word z_i . The theorem asserts that for arbitrary language H accepted by a nondeterministic one-way n -tape finite automaton the set of sets $(\psi^1(H), \psi^2(H),\dots,\psi^n(H))$ is semilinear.

On recognition of ω -languages by pushdown automata

D.Taimina

If all the ω -words form an ω -language $L \subset \{0,1\}^{\omega}$ contain no more than one symbol 1 each, and L is recognized by a nondeterministic (deterministic) pushdown automaton then L can be recognized by a deterministic finite ω -automaton as well. The counterpart of this theorem for ω -languages with two symbols 1 in an ω -word fails.

On the number of states in finite automata for identification of ω -words

D.Taimina

The number of states in deterministic finite automata recognizing the (finitary) language $L_k = \{y2y \mid y \in \{0,1\}^k\}$ and the ω -language $M_k = \{y_1 2 y_2 2 \ldots \mid (\forall n)(y_n = y_{n+1}) \& (\forall n)(y_n \in \{0,1\}^k)\}$ is compared. No less than 2^k states are needed to recognize L_k but only $2k+2$ states suffice to recognize M_k.

Bounds to the length of the words for the simulation of finite deterministic automata

J.Buls

The simulation and the simulation with a delay action es studied. A concept of automata complexity is introduced.

On the equivalence problem for multitape automata, one of which is arbitrary

A.Kālis

The decidability of equivalence for deterministic one-way mutitape automata with a bounded number of changes of the pairs of tape is proved. The same holds if one of the automata is an arbitrary deterministic one.

An estimation for the number of repetitions and inference of simple regular languages

K.Čerāns

It is proved, that in the sequence of the length n there are no more than $6n$ repetitions. There exists an algorithm inferring simple regular languages by dialogue using no more than $6n$ questions.

A model of data base with indeterministic transitions

J.Cīrulis

An algebraic model of a relational data base is briefly discussed in this note.

Recognition of tally languages by probabilistic pushdown automata

J.Kaņeps

If a language in a one letter alphabet is recognized by a probabilistic pushdown automaton with isolated cutpoint then the language is regular.

On computational power of alternating one-way mutitape finite automata

M.Alberts

Relations between the classes of languages which are accepted by $\Pi_0, \Sigma_1, \Pi_1, \Sigma_2, \Pi_2$ alternating one-way multitape finite automata are studied.

The decidability of the equivalence for the graphical expresions

A.Brāzma

A formal language of the so called graphical expressions designed for description of general regularities is introduced. The language is based upon a formalization of the 'dots' notion and is convenient for description of the programs with for loops. The equivalence for the expressions of the language is defined and the decidability is proved.

The inductive synthesis of the graphical expressions

A.Brāzma, I.Ztmane

The language of the graphical expressions is considered and a formal example of the expression is defined. The rules of an inductive inference for the synthesis of general expressions by a sufficiently large example is defined. The completeness of the given rules is proved.

ЗАКЛЮЧЕНИЕ

Полученные результаты, опубликованные в данном сборнике, составляют основу для разработки автоматизированных систем тестирования и синтеза программ, которые позволят снизить затраты на создание программного обеспечения ЭВМ. Результаты получены в рамках НИР "Индуктивные и вероятностные методы в теории программ" и "Разработка вопросов технологии программирования", выполняемых в соответствии с Комплексной программой "Кибернетика" и Комплексной программой "Математические и физические основы развития вычислительной техники и информатики" (раздел I.I2.2. "Системное математическое и программное обеспечение") согласно Постановлению № I470/I46 от 29.I2.60 АН СССР и Минвуза СССР.

Депонированных работ в сборнике не содержится.

СОДЕРЖАНИЕ

ТЕОРИЯ АЛГОРИТМОВ И ПРОГРАММ

Сборник научных трудов

Рецензенты: И.Страздиньш, канд. физ.-мат. наук, доцент РПИ им.А.Пельше;

М.Шнепс, доктор техн. наук, зам. дир. НИИ ВЭУ;

Э.Икауниекс, канд. физ.-мат. наук, доцент каф. дискретной математики и программирования ЛГУ им. П. Стучки

Редакторы: Я.Барздинь, Р.Павлова
Технический редактор А.Яковича
Корректор И.Балоде

Подписано к печати 01.08.86 ЯТ05345 Ф/б 60x84/16.
Бум. №3. 12,8 физ.печ.л. 11,9 усл.печ.л. 9,7 уч.-изд.л.
Тираж 500 экз. Зак.№ 968 Цена 1 р. 50 к.

Латвийский государственный университет им. П.Стучки
226098 Рига, б. Райниса, 19
Отпечатано в типографии, 226050 Рига, ул.Вейденбаума,5
Латвийский государственный университет им. П.Стучки

УДК 519.95

Фрейвалд Р.В. О вероятностных и детерминированных машинах Тьюринга со входом и выходом // Теория алгоритмов и программ. - Рига: ЛГУ им.П.Стучки, 1986. - С. 4-22.

Рассматриваются машины Тьюринга, которые в начале работы читают входную информацию буква за буквой с линейной скоростью. После прочтения последней буквы со входа машина имеет возможность работать как машина без входа. Построен такой язык D ; что: 1) для любого $\varepsilon > 0$ существует вероятностная машина Тьюринга, распознающая D в реальное время с вероятностью 1/3, 2) для распознавания D на любой детерминированной машине Тьюринга требуется не меньше времени чем $n^2/\log n$.
Библиограф.6 назв.

УДК 519.95

Гейдманис Д.Г. Аналог теоремы Парика для языков, принимаемых многоленточным односторонним конечным недетерминированным автоматом // Теория алгоритмов и программ. - Рига: ЛГУ им.П.Стучки, 1986. - С.23-39.

Пусть n -ленточный конечный автомат перерабатывает n -ки слов в l-буквенном алфавите. Пусть l-ка целых чисел $\psi^i(z_1,\dots,z_n)$ указывает число различных символов в слове z_i . Доказана теорема, показывающая, что для любого языка M n-ок слов (если этот язык принимается недетерминированным односторонним n -ленточным конечным автоматом) множество множеств $(\psi^1(M), \psi^2(M), \dots, \psi^n(M))$ полулинейно.
Библиограф.4 назв.

УДК 519. 95

Тайминя Д.Я. О распознавании ω-языков автоматами с магазинной памятью // Теория алгоритмов и программ. - Рига: ЛГУ им.П.Стучки, 1986. - С. 40-45.

Доказано, что если в ω-языке $L \subset \{0,1\}^{\omega}$ каждое ω-слово содержит не более одного символа 1, то из распознаваемости языка L недетерминированным (или детерминированным) ω-автоматом с магазинной памятью вытекает распознаваемость L детерминированным конечным ω-автоматом. Аналог этого утверждения для ω-языков с двумя символами 1 не имеет места.
Библиогр. 3 назв.

УДК 519.95
Тайминя Д.Я. О числе состояний конечных автоматов, распознающих равенство ω-слов. // Теория алгоритмов и программ. - Рига: ЛГУ им.П.Стучки, 1986. - С.40-45.

Сравнивается число состояний детерминированных конечных автоматов, распознающих, соответственно, финитный язык $L_k = \{y2y \mid y \in \{0,1\}^k\}$ и ω-язык $M_k = \{y_1 2 y_2 2 \ldots \mid (\forall^{\infty} n)(y_n = y_{n+1}) \,\&\, (\forall k)(y_n \in \{0,1\}^k)\}$. Для распознавания языка L_k требуется не меньше чем 2^k состояний, а для распознавания M_k достаточно $2k+2$ состояний.
Библиогр. 1 назв.

УДК 519.95
Булс Я.А. Оценка длины слова при моделировании конечных детерминированных автоматов // Теория алгоритмов и программ. - Рига: ЛГУ им.П.Стучки, 1986. - С.50-63.

Уточняется понятие моделирования. Для этих уточнений найдены длины слов. Введена некоторая мера сложности слов.
Библиограв. 1 назв.

УДК 519.95
Калис А.А.О проблеме эквивалентности многоленточных автоматов, один из которых произвольный // Теория алгоритмов и программ. - Рига: ЛГУ им.П.Стучки, 1986. - С.4-22.

Доказывается алгоритмическая разрешимость проблемы эквивалентности многоленточных детерминированных конечных автоматов с ограниченным числом переходов на пары лент. Проблема эквивалентности остается разрешимой, если один

автомат является произвольным детерминированным.
Библиограф. 5 назв.

УДК 519.95
Черанс К.Х. Линейность оценки числа повторений и синтез простых регулярных языков // Теория алгоритмов и программ. - Рига: ЛГУ им.П.Стучки, 1986 - С.4-22.

Доказано, что последовательность длины n, не содержит более $6n$ повторений. Существует алгоритм, который синтезирует простые регулярные языки в диалоге, требуя не более $6n$ вопросов.
Библиограф. 6 назв.

УДК 519.689.2
Цирулис Я.П. Модель базы данных с недетерминированными переходами // Теория алгоритмов и программ. - Рига: ЛГУ им.П.Стучки, 1986. - С.87-95.

Предлагается алгебраическая модель реляционной базы данных, представляющая собой автомат вида (S, Σ, Q, A), где S - множество состояний базы, Σ - система операторов управления ею, Q - алгебра запросов, A - подходящая алгебра отношений и кроме того, заданы операции $\delta: S\times\Sigma\to\mathcal{P}(S)$, $\alpha: S\times Q\to A$, $\diamond: \Sigma\times Q\to Q$, подчиняющиеся определенным условиям. Рассматриваются несколько элементарных свойств такой модели, и отмечаются некоторые естественно возникающие задачи, требующие, в частности, привлечения идей динамической логики.
Библиограф. 7 назв.

УДК 519.95
Канепс Я.Я. Распознавание языков в однобуквенном алфавите вероятностными автоматами с магазинной памятью // Теория алгоритмов и программ. - Рига: ЛГУ им.П.Стучки, 1986. - С. 96-97.

Для языков в однобуквенном алфавите доказано, что из распознаваемости с изолированной точкой сечения этого языка на вероятностном автомате с магазинной памятью вытекает распознаваемость этого языка на детерминированном конечном

автомате.
Библиограф. 2 назв.

УДК. 519.95

АлбертсМ.Я.О вычислительной силе двухленточных конечных альтернирующих автоматов // Теория алгоритмов и программ. - Рига: ЛГУ им.П.Стучки, 1986. - С.98-102.

Изучены отношения между классами языков, которые принимаются альтернирующими односторонними многоленточными конечными автоматами Π_0, Σ_1, Π_1, Σ_2, Π_2.
Библиограф. 2 назв.

УДК 519.71

Бразма А.Н. Разрешимость проблемы эквивалентности для графических выражений // Теория алгоритмов и программ. - Рига: ЛГУ им.П.Стучки, 1986. - С.103-156.

Предлагается формальный язык т.н. графических выражений. Предлагаемый язык основан на формализации понятия многоточия и является удобным средством описания програми с for - циклами. Введено понятие эквивалентности для графических выражений и доказана его алгоритмическая разрешимость.
Библиограф. 4 назв.

УДК 519.71

Бразма А.Н. Этмане И.Э. Индуктивный синтез графических выражений //Теория алгоритмов и программ. - Рига: ЛГУ им.П.Стучки, 1986 - С. 156-189.

Рассмотрен язык графических выражений. Определено понятие формального примера графического выражения. Даны правила индуктивного вывода для синтеза общих выражений по их достаточно выразительным примерам. Доказана полнота предложенной системы правил вывода.
Библиограф. 3 назв.

1 р. 50 к.