УДК 519.68:510

С. А. Аветисян

О представлении бестиповых λ-термов помеченными бинарными деревьями

(Представлено академиком Н.У. Аракеляном 2/III 2004)

1. *Используемые определения* (см. [1, 2]). Зафиксируем счетное множество переменных V. Множество термов Λ является наименьшим множеством, удовлетворяющим следующим условиям:

1) если $x \in V$, то $x \in \Lambda$;

2) если $t_1, t_2 \in \Lambda$, то $(t_1 t_2) \in \Lambda$;

3) если $x \in V$ и $t \in \Lambda$, то $(\lambda x t) \in \Lambda$.

Введем сокращенную запись термов: терм $(\ldots(t_1 t_2)\ldots t_k)$, где $t_i \in \Lambda$, $i=1,\ldots,k$, $k > 1$, условимся обозначать $t_1 t_2 \ldots t_k$; терм $(\lambda x_1(\lambda x_2(\ldots(\lambda x_m t)\ldots)$, где $x_j \in V$, $t \in \Lambda$, условимся обозначать $\lambda x_1 x_2 \ldots x_m.t$, $j=1,\ldots,m$, $m > 0$.

Традиционным образом вводятся понятия свободного и связанного вхождения переменной в терм, понятие свободной переменной терма.

Через $t[x_1,\ldots,x_m]$ условимся обозначать терм t с указанием интересующих нас попарно различных переменных $x_1,\ldots,x_m$, $m \geq 1$. Через $t[t_1,\ldots,t_m]$ обозначим терм, полученный в результате одновременной подстановки термов $t_1,\ldots,t_m$ в терм t вместо всех свободных вхождений переменных $x_1,\ldots,x_m$ соответственно.

Термы t_1 и t_2 назовем конгруэнтными (обозначим $t_1 \equiv t_2$), если один терм можно получить из другого переименованием связанных переменных. Далее мы не будем отличать конгруэнтные термы.

Напомним понятие β-редукции:

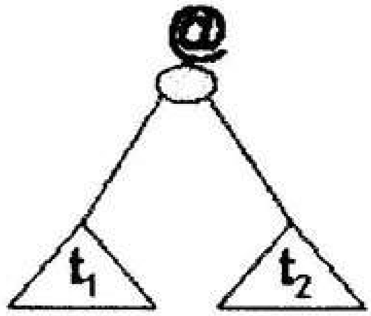
$$\beta = \{((\lambda x.t[x])t', t[t'])|t, t' \in \Lambda, x \in V\}.$$

Одношаговая β-редукция ($\rightarrow_\beta$) определяется обычным образом. Напомним, что терм $(\lambda x.t[x])t'$ называется β-редексом (далее просто редексом), а терм $t[t']$ - его сверткой. Терм, не содержащий редексов, называется β-нормальной формой (далее просто нормальной формой). Терм вида $\lambda x_1\ldots x_k.xt_1\ldots t_m$, где $x, x_i \in V$, $t_j \in \Lambda$, $i=1,\ldots,k$, $j=1,\ldots,m$, $k \geq 0$, $m \geq 0$, называется головной нормальной формой.

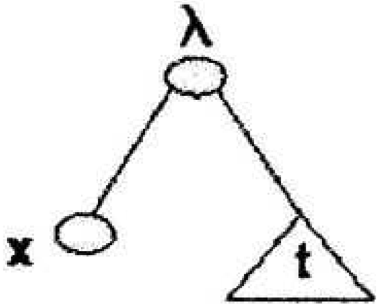
2. *Представление терма.* Любой терм $t \in \Lambda$ можно представить в виде помеченного бинарного дерева, нетерминальные узлы которого представляют собой операции аппликации или абстракции, терминальные узлы (листья) - переменные терма. Под t-деревом будем понимать дерево, соответствующее терму t, которое будем отождествлять с самим термом t. Дадим рекурсивное определение t-дерева.

- Любой переменной $x \in V$ соответствует дерево, состоящее из одного узла. Отметим ее переменной x. Такой узел назовем v-узлом.

- Если t_1, $t_2 \in \Lambda$, то терму $(t_1 t_2)$ соответствует дерево, корень которого помечен символом @. Левое поддерево является t_1-деревом, правое поддерево - t_2-деревом. Корень $(t_1 t_2)$-дерева будем называть @-узлом, t_1-дерево - левым поддеревом @-узла, t_2-дерево - правым поддеревом @-узла. $(t_1 t_2)$-дерево имеет следующий вид:



- Если $t \in \Lambda$, $x \in V$, то терму $(\lambda x.t)$ соответствует дерево, корень которого помечается символом λ и называется λ-узлом. Левое поддерево представляется листом, помеченным переменной x. Левое поддерево будем называть переменной λ-узла. Правое поддерево является t-деревом. Правое поддерево будем называть телом λ-узла. $(\lambda x.t)$-дерево имеет следующий вид:



Из определения следует, что t-дерево содержит узлы трех типов: v-узел, @-узел, λ-узел. Например, дерево, соответствующее терму $((x_1 x_2)(\lambda x_3.x_3 x_4))$, представлено на рис. 1.

Каждому терму t, а, следовательно, и t-дереву, соответствует множество свободных переменных. В описанном выше примере (рис. 1) переменные x_1, x_2, x_4 являются свободными.

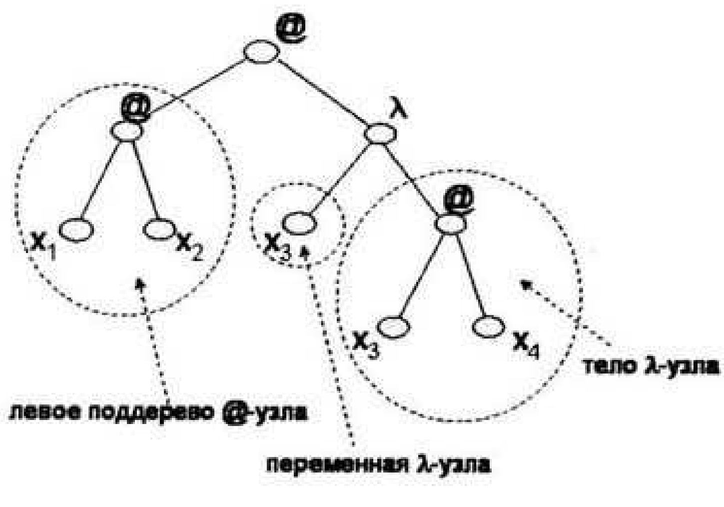


Рис.1. Представление терма $((x_1x_2)(\lambda x_3.x_3x_4))$ в виде помеченного бинарного дерева

3. *Представление головной нормальной формы.* Любой терм t имеет следующий вид:

$$\lambda x_1x_2\ldots x_n.t't_1t_2\ldots t_k,$$

где t' - переменная, если k=0, и t' - переменная, либо терм вида $\lambda x.t''$, если $k > 0$, $x, x_i \in V$, $i=1,\ldots,n$, $t', t'', t_j \in \Lambda$, $j=1,\ldots,k$, $k \geq 0$, $n \geq 0$. Соответствующее t-дерево представлено на рис. 2.

Под левым гребнем t-дерева будем понимать путь, состоящий из @-узлов, встречающихся при спуске по левым поддеревьям @-узлов начиная с первого (рис. 2). Если в представлении t-дерева t' представляет собой v-узел, то такому t-дереву (рис. 3) соответствует головная нормальная форма, а соответствующий терм имеет следующий вид:

$$\lambda x_1x_2\ldots x_n.xt_1t_2\ldots t_k,$$

где $x, x_i \in V$, $t_j \in \Lambda$, $i=1,\ldots,n$, $j=1,\ldots,k$, $k,n \geq 0$.

t-дерево, соответствующее головной нормальной форме, характеризуется v-узлом, которому соответствует переменная головной нормальной формы x, и @-узлами, правые поддеревья которых являются t_1-деревом, t_2-деревом, ..., t_k-деревом соответственно (рис. 3). Описанные @-узлы являются узлами левого гребня t-дерева. Если n=0, то отсутствуют лидирующие (корневые) λ-узлы, если k=0, то отсутствуют @-узлы левого гребня t-дерева.

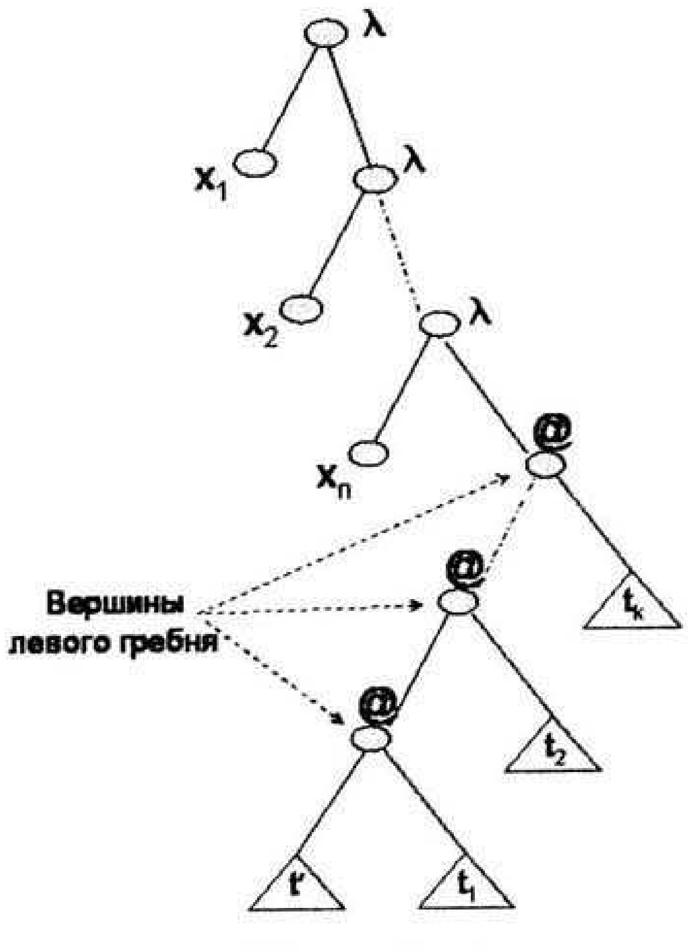


Рис.2. Общий вид t-дерева

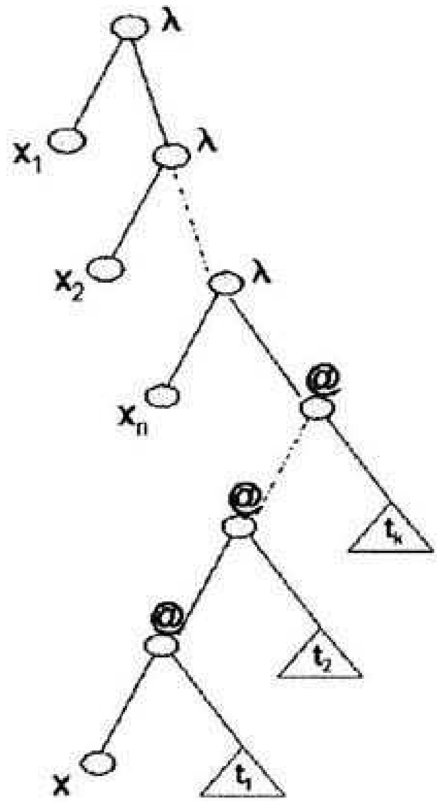
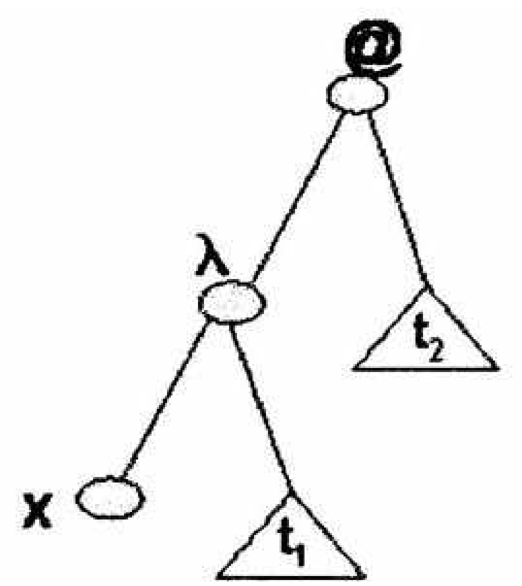


Рис.3. t-дерево, соответствующее головной нормальной форме $\lambda x_1 x_2 \ldots x_n.xt_1t_2\ldots t_k$.

4. *Преобразование* t-*дерева в* t′-*дерево, соответствующее одношаговой* β-*редукции.* Рассмотрим сначала получение свертки редекса. Редексом является терм $(\lambda x.t_1[x])t_2$, где t_1 - тело редекса, t_2 - аргумент редекса, x - переменная редекса. Дерево редекса имеет следующий вид:



где t_2-дерево соответствует аргументу редекса, t_1-дерево соответствует телу редекса. Под корнем редекса будем понимать корневую вершину дерева редекса.

Алгоритм *Red1* преобразования дерева редекса в дерево свертки состоит из следующих шагов:

1) выделить t_2-дерево, соответствующее аргументу редекса;

2) заменить каждый лист, представляющий свободное вхождение переменной x в t_1-дереве, на t_2-дерево. Преобразованное t_1-дерево соответствует свертке;

3) за корневую вершину дерева свертки принять корневую вершину преобразованного t_1-дерева;

4) удалить все не используемые в преобразованном t_1-дереве узлы.

При замене переменной в теле редекса на аргумент редекса свободная переменная аргумента редекса может связаться. При обнаружении связывания осуществляется переименование связанной переменной тела редекса на новую переменную. Пример получения свертки $(\lambda x_1.(x_1 x_3))x_4$ представлен на рис. 4.

Алгоритм *Red* преобразования t-дерева в t′-дерево, согласно одношаговой β-редукции, состоит из следующих шагов:

1. найти @-узел, который является корнем редекса. Если соответствующий @-узел найден, то отметить его;
2. рассмотреть t″-дерево, корнем которого является отмеченный @-узел;
3. преобразовать t″-дерево в дерево-свертку согласно алгоритму Red1;
4. заменить поддерево, корнем которого является отмеченный @-узел, на преобразованное t″-дерево-свертку.

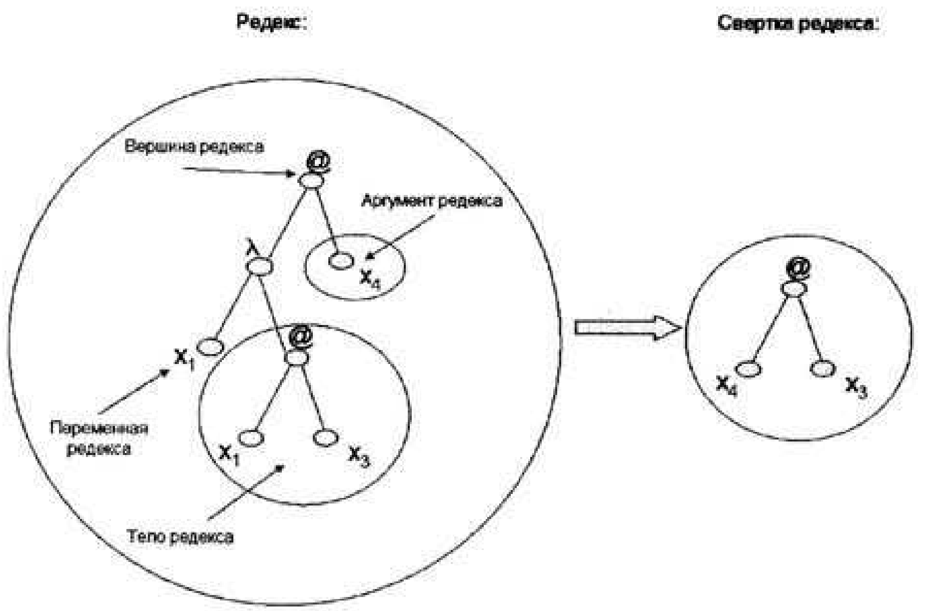


Рис.4. Преобразование дерева-редекса в дерево-свертку для редекса $(\lambda x_1.(x_1\ x_3))x_4$.

Утверждение 1. Пусть терм t не является нормальной формой, тогда алгоритм Red переводит t-дерево в t′-дерево так, что $t \rightarrow_\beta t'$.

5. *Преобразование* t-*дерева в* t′-*дерево, соответствующее подстановке.* Пусть $t \equiv t[x_1,\ldots,x_m]$, где $x_1,\ldots,x_m$ попарно различные переменные, $m \geq 1$. Пусть $t' \equiv t[t_1,\ldots,t_m]$, где $t_1,\ldots,t_m$ - термы.

Алгоритм *Sub* преобразования t-дерева в t′-дерево, соответствующее подстановке, состоит из следующих шагов:

1. в t-дереве найти и отметить v-узлы, соответствующие свободным вхождениям переменных $x_1,\ldots,x_m$;
2. заменить каждый отмеченный v-узел, соответствующий свободному вхождению переменной x_i, на t_i-дерево, $i=1,\ldots,m$.

Очевидно, что при подстановке возникает проблема связывания свободных переменных. При обнаружении связывания происходит переименование переменных λ-узлов. Переименование переменных λ-узлов заключается в переименовании всех свободных вхождений данной переменной в теле λ-узла.

Утверждение 2. Пусть $t,t_1,\ldots,t_m$ - термы, $x_1,\ldots,x_m$ - переменные и $t \equiv t[x_1,\ldots,x_m]$, $m \geq 1$. Тогда алгоритм Sub переводит t-дерево в t′-дерево так, что $t' \equiv t[t_1,\ldots,t_m]$.

Ереванский научно-исследовательский институт
автоматизированных систем управления

Литература

1. *Barendregt H.P.* The lambda calculus, its syntax and semantics. Amsterdam. North Holland. 1981 (рус. пер. Барендрегт Х. Ламбда-исчисление, его синтаксис и семантика. М. Мир. 1985. 606 с.).
2. *Нигиян С.А., Аветисян С.А.* - Программирование. 2002. N3. С. 5-14 (англ. пер. Nigiyan S.A., Avetisyan S.A. - Programming and Computer Software. 2002. V. 28. N3. P. 119-126).

Ս. Ա. Ավետիսյան

Առանց տիպերի λ-տերմերի ներկայացումը նշած բինար ծառերի օգնությամբ

Նկարագրված է առանց տիպերի λ-տերմերի ներկայացման եղանակը նշած բինար ծառերի միջոցով: Այս ծառերի տերևները նշած են փոփոխականներով, իսկ ներքին գագաթները աբստրակցիայի և ապլիկացիայի գործողությունների սիմվոլներով: Առաջարկված են ծառերի ձևափոխության ալգորիթմներ, որոնք համապատասխանում են միաքայլ β-ռեդուկցիային և տեղադրմանը: