ИНФОРМАТИКА, ВЫЧИСЛИТЕЛЬНАЯ ТЕХНИКА И УПРАВЛЕНИЕ

INFORMATICS, COMPUTER ENGINEERING AND CONTROL

УДК 519.6 DOI: 10.17213/0321-2653-2017-4-5-12

ТРАССИРУЮЩАЯ НОРМАЛИЗАЦИЯ НЕТИПИЗИРОВАННОГО ЛЯМБДА-ИСЧИСЛЕНИЯ

© 2017 г. Д.А. Березун

Санкт-Петербургский государственный университет, г. Санкт-Петербург, Россия

UNTYPED LAMBDA-CALCULUS NORMALIZATION BY TRAVERSALS

D.A. Berezun

Saint Petersburg University, Saint Petersburg, Russia

Березун Даниил Андреевич – аспирант, Санкт-Петербургский государственный университет, г. Санкт-Петербург, Россия. E-mail: d.berezun@2009.spbu.ru

Berezun Daniil Andreyevich – post-graduate student, Saint Petersburg University, Saint Petersburg, Russia. E-mail: d.berezun@2009.spbu.ru

Трассирующая нормализация является новым подходом к нормализации λ-термов путём обхода их абстрактного синтаксического дерева. Изначально трассирующая нормализация была обоснована лишь для некоторых типизированных подмножеств λ-исчисления, а её расширение на нетипизированное λ-исчисление до сих пор не имело формального представления и обоснования. В статье приводится первое формальное представление процедуры трассирующей нормализации для нетипизированного λ-исчисления и показывается её корректность относительно линейной редукцией.

Ключевые слова: нормализация; трасса; λ-исчисление; линейная редукция; симуляция; семантики.

Normalization by traversals is a new approach to normalization of lambda-terms by tracing their abstract syntax trees. Originally designed is context of simply-typed lambda-calculus, it was then extended to untyped lambda-calculus but this extension has no formal presentation and correctness proof. In the paper we present the first formal presentation of untyped lambda-calculus normalization by traversals procedure via transition system and show its correctness with respect to the linear reduction.

Keywords: normalization; traversal; λ-calculus; linear reduction; simulation; semantics.

1. Введение

Трассирующая нормализация представляет собой новый способ вычисления нормальной, в широком смысле[1], формы терма путём построения *обхода* его абстрактного синтаксического дерева, представленного в некотором конкретном синтаксисе. Отличительной особенностью данного подхода является то, что в то время как классическая β-редукция путём произведения подстановок изменяет исходный терм, трассирующая нормализация напротив – оставляет его нетронутым, что позволяет успешно производить компиляцию лямбда-термов путём специализации на них нормализирующей процедуры [1]. Порождённая игровой семантикой [2, 3], своё применение трассирующая нормализация нашла в проверках моделей высшего порядка [2, 4, 5].

Тем не менее все существующие подходы к трассирующей нормализации определены либо для некоторых типизированных подмножеств лямбда-исчисления, например, безопасного лямбда-исчисления [3]. В статье мы обобщим этот

[1] Под *нормальной в широком смысле* подразумевается такая форма терма, в которой он является значением – нормальной формой – с точки зрения некоторой предопределённой стратегии вычислений.

подход до нетипизированного λ-исчисления. Мы также приведём формальное представление этого обобщения в виде системы переходов и покажем его корректность относительно *полной головной линейной редукции* [6], ПГЛР, (Complete Head Linear Reduction, CHLR).

2. Контекст работы

В статье будем придерживаться соглашения Барендрегта, а термы рассматривать с точностью до α-эквивалентности.

Головная линейная редукция (ГЛР). В 2004 г. Данос и Ренье осуществили попытку связать игровую семантику лямбда-исчисления с традиционными подходами к нормализации лямбда-термов [7]. В результате было введено понятие *головной линейной редукции*, ГЛР (Head Linear Reduction, HLR), формализованное с помощью абстрактной машины Кривина [8] и основанной на ней PAM – Pointer Abstract Machine. Головная линейная редукция производит редукцию λ-терма путём линейных подстановок головного вхождения переменной «по-надобности» и завершается в так называемой *псевдо-головной нормальной форме*, ПГНФ. Тем не менее головная линейная редукция не является нормализирующей: для получения нормальной формы терма из псевдо-головной нормальной формы требуется произведение его дальнейшей головной редукции.

Полная головная линейная редукция, ПГЛР, является обобщением головной линейной редукции, интуитивно производящая рекурсивное применение последней к аргументам псевдоголовной нормальной формы терма после её достижения посредством головной линейной редукции. Полная головная линейная редукция введена и формализована с помощью системы переходов, а также показана её корректность в работе [6]. В отличие от ГЛР, ПГЛР завершается в нормальной форме терма тогда и только тогда, когда последняя существует [6].

Трассирующая нормализация. Понятие *обхода* было введено Онгом в контексте рекурсивных схем высшего порядка с целью генерации древовидных структур нулевого порядка [4]. В дальнейшем понятие обхода было распространено до простого типизированного лямбда-исчисления [2, 3], и установлено взаимнооднозначное соответствие между множеством обходов синтаксического дерева терма простого типизированного лямбда-исчисления и значением его семантической функции в *невинной модели взаимодействия* в игровой семантике – теорема о соответствии пути в η-длинной β-нормальной формы терма и его обхода [2, 5], что позволило определить новый подход к нормализации термов простого типизированного лямбда-исчисления без применения классической β-редукции – *трассирующую нормализацию* [1 – 3, 5]. Трассирующая нормализация в стиле Онга, ОПН, определена для термов простого типизированного лямбда-исчисления, находящихся в η-длинной форме [2], которая получается путём его полного η-расширения, а также замены операторов применения на операторы длинного применения $@_{long}$. *Обходом* терма называется обоснованная последовательность вершин АСД его η-длинной формы. Каждому дереву некоторого корректного просто типизированного терма соответствует некоторое непустое множество обходов Trav, каждый из которых соответствует пути в АСД η-длинной β-нормальной формы исходного терма, и однозначно определяющих последнюю.

Такой подход к нормализации термов оказался согласован с головной линейной редукцией. Заметим, что важной составляющей частью трассирующей нормализации в стиле Онга является преобразование терма в η-длинную форму, которое в свою очередь существенно опирается на типы и не может быть напрямую распространено до нетипизированного λ-исчисления, в котором, ввиду отсутствия типизации, неконтролируемое η-расширение терма, очевидно, расходится. Тем не менее, текущие работы Блюма [9] показывают, что η-*расширения на лету* позволяет определить некоторое подобие трассирующей нормализации в стиле Онга для нетипизированного лямбда-исчисления [9].

Процедура нетипизированной трассирующей нормализации. В статье [1] была предпринята попытка обобщить трассирующую нормализацию до нетипизированного λ-исчисления. В отличие от трассирующей нормализации в стиле Онга, предложенный подход определяется на стандартном представлении λ-исчисления, не требуя преобразования термов в η-длинную форму. Он получил название ПНТН – процедура нетипизированной трассирующей нормализации, а его обоснованием служит ряд последовательных стандартных преобразований[2] классической

[2] Преобразования, широко используемые при компиляции, трансляции и определении семантики различных исчислений: использование замыканий, окружений, линеаризация и т.д. (см. в [1,10–12])

естественной семантики нетипизированного лямбда-исчисления, приводящих последнюю к ПНТН. Такое представление, хоть и является наглядным, не содержит формального математического доказательства корректности, а значит, корректность ПНТН, в некотором смысле, принимается «на веру».

В следующих разделах статьи мы приведём формальное представление ПНТН в виде системы переходов и установим взаимнооднозначное соответствие между приведённой системой и системой переходов для полной головной линейной редукции, приведённой в [6], связав тем самым трассирующую нормализацию для нетипизированного λ-исчисления с линейной редукцией и формально показав её корректность, а именно то, что ПНТН является нормализирующей.

3. Головная линейная редукция и ОПНТН

Система переходов для головной линейной редукции, заимствованная из [6], приведена на рис. 1. Далее мы приведём представление ОПНТН в виде системы переходов и установим соответствие между приведёнными системами переходов для ГЛР и ОПНТН.

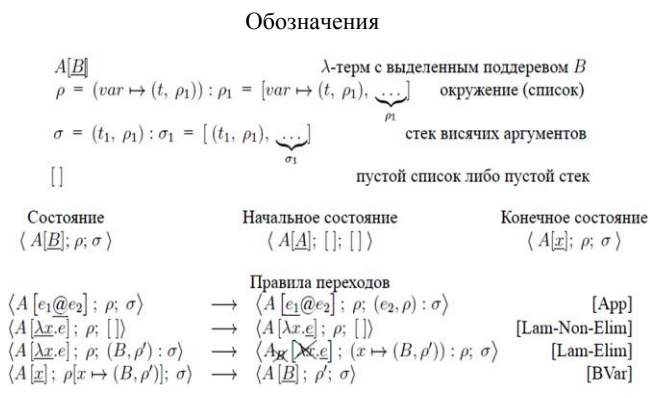


Рис. 1. Система переходов для головной линейной редукции / Fig. 1. Transition system for head linear reduction

3.1. Система переходов для ОПНТН

ОПНТН, или базовый ПНТН, является ограниченной версией ПНТН (см. раздел 4, [1]). ОПНТН соответствует ГЛР, являющейся основой для ПГЛР. Далее мы установим взаимнооднозначное соответствие между системами переходов для ОПНТН и ГЛР, определив функцию преобразования состояний одной системы переходов в состояния другой. Более того, правила переходов рассматриваемых систем переходов напрямую отображаются друг в друга. В отличие от ГЛР, состояние системы переходов для ОПНТН является обоснованной последовательностью вершин входного лямбда-терма, снабженных указателями на ранние вершины последовательности. В ОПНТН существует два нижеследующих рода указателей.

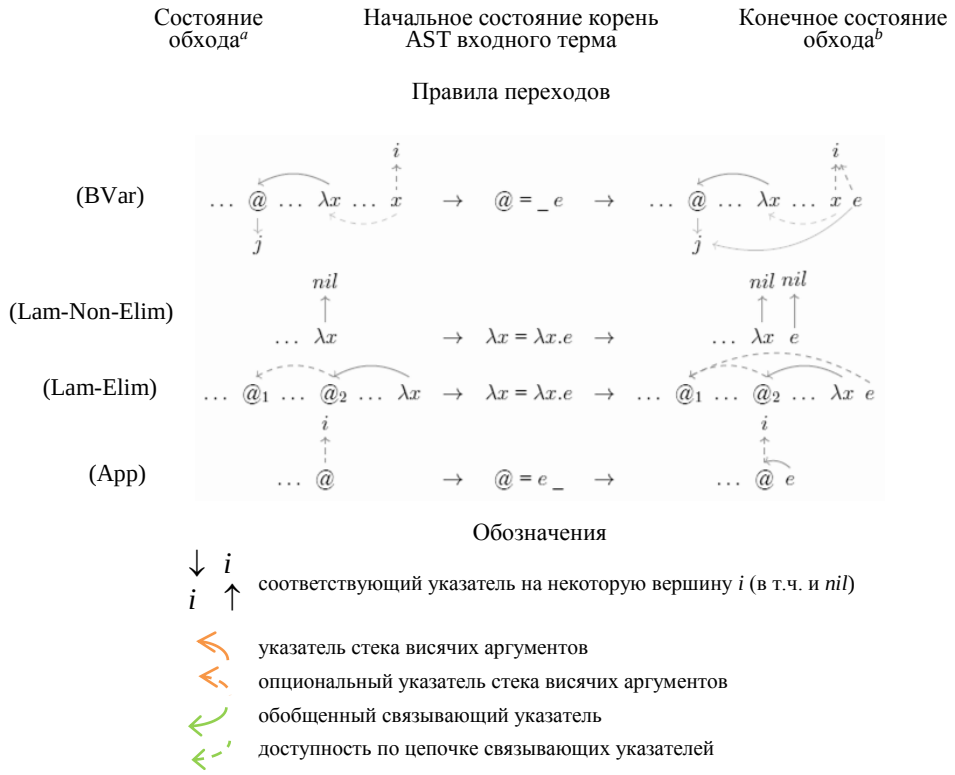
Обобщённый связывающий указатель. Связывающий указатель является ОПНТН-эквивалентом окружения: он позволяет построить соответствующее окружение на каждом шаге алгоритма. Для каждого токена он указывает на токен, представляющий последнюю абстракцию, которую необходимо добавить в окружение при его формировании, если эта абстракция участвует в образовании какого-либо простого редекса.

Указатель стека висячих аргументов. Он является ОПНТН-эквивалентом стека висячих указателей системы переходов для ГЛР. Инвариантом этого указателя является то, что он всегда указывает либо на токен, представляющий применение термов, либо отсутствует вовсе. Указатель стека висячих аргументов позволяет построить список спинальных аргументов терма на каждом шаге. А именно, для токенов-абстракций он всегда указывает на токен-применение, с аргументом которого он формирует простой редекс, а для остальных токенов – на применения, являющиеся непосредственным предком последнего спинального аргумента.

Система переходов для ОПНТН приведена на рис. 2. Для краткости изложения, а также ввиду их простоты, правила установки обобщённого связывающего указателя опущены во всех правилах, кроме (BVar). В остальных случаях применяется следующее правило: если последний токен обхода был абстракцией, то у добавляемого в обход токена обобщённый связывающий указатель выставляется на эту самую абстракцию, в противном случае – он выставляется такой же, как у предыдущего токена.

3.2. Соответствие между ГЛР и ОПНТН

Для того чтобы восстановить состояние системы переходов для ГЛР из текущего обхода, необходимо определить каждую из его компонент: текущий терм с выделенной вершиной, текущее окружение и корректный стек висячих аргументов ρ. Итак, пусть дан обход *t*, тогда терм, окружение и стек висячих аргументов могут быть восстановлены из него нижеследующим образом.



[a]Для удобства формального изложения мы будем считать, что состояние системы переходов для BUNP является тройка $\langle t, \beta, \alpha\rangle$, где $t : \Sigma\to[1\ldots|t|]$ является упорядоченной последовательностью, $|t|$ – длина обхода t, а частичные функции α, $\beta : [1\ldots|t|]\to[1\ldots|t|-1]$ определяют множества указателей стека висячих аргументов и связывающих указателей соответственно.
[b]Конечным состоянием является обход, такой, что его последний токен является токеном-переменной, не имеющим указателя висячих аргументов, $t=\_\bullet x{:}\alpha(x)=\bot$, являющейся либо свободной, $\nexists k\in\mathrm{N} : \alpha\circ\beta^k(x)=\lambda\, x$, либо связанной, чья абстракция не связана никаким простым редексом, $\nexists\, k\in\mathrm{N} : \beta^k(x)=\lambda x\Rightarrow\beta^k(x)=\lambda x\notin\mathrm{D}(\alpha)$.

Рис. 2. Система переходов для ОПНТН / Fig. 2. BUNP transition system

Терм. Обход t представляет собой подпоследовательность самого левого пути в АСД терма. Таким образом, для восстановления соответствующего первого компонента системы переходов для ГЛР необходимо снабдить токены-применения соответствующими висячими аргументами, игнорируя вхождения связанных переменных, и выделить вершину, соответствующую последнему токену обхода.

Окружение. Если последним токеном текущего обхода является абстракция, то при восстановлении окружения она не рассматривается. Пусть R – множество всех простых редексов обхода t; Λ – список λ-абстракций, доступных из последнего токена по цепочке связывающих указателей, тогда текущее окружение

$$\rho(t) = \{x \mapsto (A,\ \rho')\,|\,(\lambda x, A)\in R\}\restriction\Lambda,$$

где ρ' определено рекурсивно. Заметим, что R упорядочено:

$$\forall r1 = (\lambda x, A), r2 = (\lambda y, B)\in R : r1 > r2 \Leftrightarrow \lambda x$$

входит в обход до токена λy.

Висячие аргументы. Стек висячих аргументов получается из обхода ограничением последнего на список токенов, доступных из последнего по цепочке указателей висячих аргументов, и снабжением их соответствующими окружениями, согласно предыдущему пункту.

Теорема 1. (Согласованность ОПНТН с ГЛР). *Пусть* S и T — соответствующие состояния систем переходов для ГЛР и ОПНТН, а S' и T' получаются из S и T применением правил $rule_S$ и $rule_T$ соответственно, *тогда* правила имеют одинаковые имена, а состояние S' получается из состояния T' согласно вышеописанному алгоритму. Иными словами, диаграмма, приведённая на рис. 3, является коммутативной.

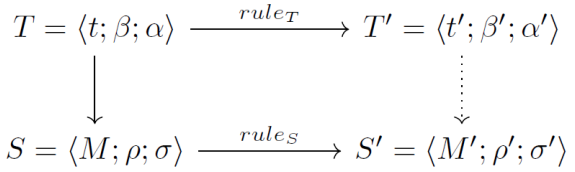


Рис. 3. Иллюстрация к теореме 1
/ Fig. 3. Illustration of the Theorem 1

Доказательство. Индукция по количеству применений правила $rule_T$, база которой тривиальна. Индукционный переход доказывается разбором случаев для правила $rule_T$. Для краткости, мы рассмотрим только правило (App), остальные случаи доказываются по аналогии.

Первой компонентой S является терм $M[\underline{@}]$. Согласно правилам системы переходов, правило [App] может быть применено к состоянию S и только оно. В обеих системах переходов правила требуют перейти к левому ребёнку применения, таким образом первый компонент S' в точности соответствует тому, который будет восстановлен из T'. Окружение не меняется, $\rho_{S'} \overset{(App)}{=} \rho_S = \rho_T \overset{[App]}{=} \rho_{T'}$, а стек висячих аргументов расширяется аргументом рассматриваемых применений, совпадающих согласно ИП.

Следствие 1.1. Первой компонентой состояния, получающегося восстановлением из конечного состояния системы переходов для ОПНТН согласно приведённым правилам, является терм в ПГНФ.

4. ПНТН и полная головная линейная редукция

Система переходов для полной головной линейной редукции, заимствованная из [6], приведена на рис. 4. Для удобства изменения по сравнению правилами для ГЛР выделены с помощью прямоугольников: $\boxed{выделение}$. По сути система переходов для ПГЛР отличается от системы переходов для ГЛР введением новых правил [FVar-*], отвечающих за рекурсивное применение к аргументам, и нового символа-разделителя $, запрещающего связывать абстракции и применения, не находящиеся на одном пути в абстрактном синтаксическом дереве терма, соответствующего текущему состоянию системы переходов.

Далее мы приведём ПНТН в виде системы переходов. Заметим, что ниже эта система переходов имеет в точности столько же правил переходов, сколько и система переходов для полной головной линейной редукции, приведённая на рис. 4. Подобно тому как система переходов для ПГЛР является обобщением системы переходов для ГЛР, так и система переходов для ПНТН является обобщением системы переходов для ОПНТН. Как и в ОПНТН, в ПНТН каждый токен может быть снабжён указателями двух родов. *Указатель стека висячих аргументов*, который бывает двух типов: *внутриуровневый,* имеющий тот же смысл, что и одноимённый указатель в ОПНТН, и *межуровневый*. В то время как межуровневый указатель всё ещё указывает на последний висячий аргумент, он запрещает формирование простого редекса. Интуитивно, он является аналогом символа-разделителя $ системы переходов для ПГЛР. *Обобщённый связывающий указатель* имеет тот же смысл, что и у ОПНТН, а правила его установки остаются неизменными. Правила переходов для ПНТН приведены на рис. 5. Для обеспечения детерминированности системы переходов, мы будем считать, что правила упорядочены и в случае, когда несколько правил может быть применено, применяется то, что приведено на рис. 5. В действительности, в конфликте могут находится лишь правило (BVar) и любое из правил (FVar-*), а наш порядок означает, что последние могут быть применены тогда и только тогда, когда не может быть применено первое, иными словами, правила для свободных переменных применяются, только если текущая переменная не связана никаким простым редексом.

Соответствие между ПГЛР и ПНТН

Для восстановления состояния системы переходов для ПГЛР из соответствующего состояния системы переходов для ПНТН мы определим способ восстановления каждой из его компонент по отдельности. Обход представляет собой обход АСД в глубину, поэтому для восстановления соответствующего терма необходимо поочерёдно добавлять токены, опуская связанные переменные, вместо которых была произведена линейная подстановка, за исключением тех случаев, когда последним токеном обхода является токен, представляющий такую переменную, и вычёркивая простые редексы – пары токенов $(@, \lambda x) : \alpha(\lambda x) = @$. Последний токен текущего обхода определяет выделенную вершину, а стек висячих аргументов является упорядоченным списком аргументов, которые необходимо подставить, как аргументы применений. Окружение восстанавливается так же, как и в

случае ОПНТН. Построение текущего окружения прекращается при появлении первого межуровневого указателя. Стек висячих аргументов восстанавливается напрямую из текущей цепочки указателей стека висячих аргументов: внутриуровневые указатели определяют текущий аргумент, окружение которого восстанавливается согласно предыдущему пункту, а межуровневые, помимо вышеописанного, соответствуют добавлению в стек символа-разделителя $.

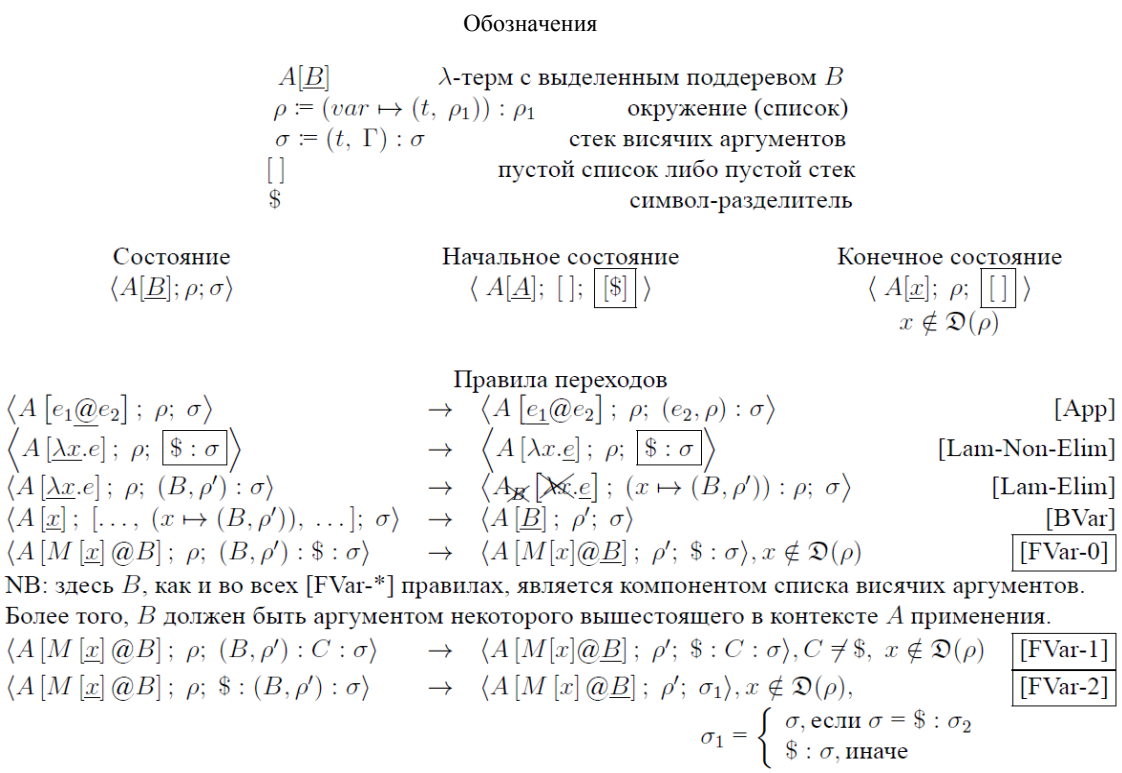


Рис. 4. Система переходов для полной головной линейной редукции
/ Fig. 4. Transition system for complete head linear reduction

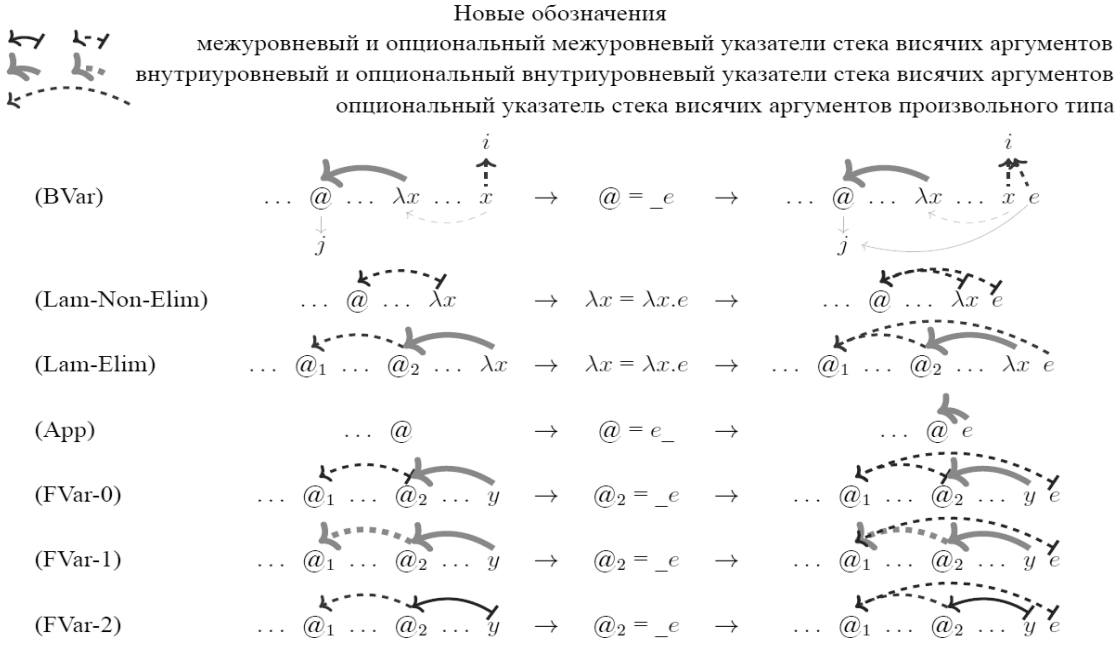


Рис. 5. Правило переходов для UNP / Fig. 5. UNP transition system rules

Теорема 2. (Согласованность ПНТН с ПГЛР)**.** *Пусть* S и T – соответствующие состояния систем переходов для ПГЛР и ПНТН, а S' и T' получаются из S и T применением правил $rule_S$ и $rule_T$ соответственно, *тогда* правила $rule_S$ и $rule_T$ имеют одинаковые имена, а состояние S' получается из состояния T' согласно вышеописанному алгоритму.

Доказательство. Аналогично Теореме 1. Мы рассмотрим только случай (FVar-0), остальные случаи доказываются аналогичным образом. Итак,

$$S \overset{\text{ИП}}{=} \langle A[M[\underline{x}]@B]; \rho; (B,\rho'):\$:\sigma\rangle \overset{[FVar-0]}{\rightarrow} S' = \langle A[M[x]@\underline{B}]; \rho'; \$:\sigma\rangle.$$

Пусть состояние, восстанавливаемое из T', $S_{T'} \rightleftharpoons \langle M_{T'}; \rho_{T'}; \sigma_{T'}\rangle$. Тогда, по ИП $M_{T'} = A[M[x]@\underline{B}]$, $\rho_{T'} = \rho'$ для токена $@_2$, $\sigma_{T'}$ отличается от $\sigma_T = (B,\rho'):\$:\sigma$ снятием вершины последнего, т.е. $\sigma_{T'} = \$:\sigma$.

Следствие 2.1. (ПНТН является нормализирующей). ПНТН завершается тогда и только тогда, когда нормальная форма входного терма существует.

Заключение

Описана трассирующая нормализация и установлена связь между трассирующей нормализацией нетипизированного лямбда-исчисления и линейной редукцией. А именно, мы ввели пару процедур трассирующей нормализации в виде систем переходов (см. рис. 5 и 2), показав их согласованность с полной головной (см. теорему 2) и головной (см. теорему 1) линейными редукциями. Показав, тем самым, что процедура нетипизированной трассирующей нормализации является 1-эффективной нормализирующей процедурой.

Пример А

Рассмотрим пример работы системы переходов процедуры трассирующей нормализации ПНТН на примере терма *NPR*, где:

$$N = \lambda h\,.\,\lambda z\,.\,h\ @\ (\lambda x\,.\,(h\ @\ (\lambda q\,.\,x)\ @\ a))\ @\ (z\ @\ a),$$
$$P = \lambda f\,.\,\lambda y\,.\,f\ @\ ((g\ @\ (\lambda b\,.\,b))\ @\ y),$$
$$R = g\ @\ (\lambda n\,.\,n).$$

Для большей наглядности введём два специальных токена = и ||, которые будут означать, что в этом месте произошла подстановка связанной переменной или произошёл переход к аргументу стека висячих аргументов соответственно. Обход терма *NPR* приведён на рис. А.

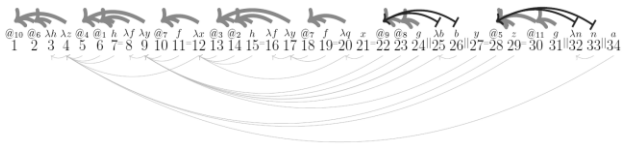


Рис. А. Обход терма *NPR* / Fig. A. *NPR*-term traversal

Итак, для восстановления нормальной формы терма из обхода необходимо удалить из него все связанные переменные, вместо которых была произведена подстановка, – токены-переменные перед токеном = – $\{11,15,19,21,27,29\}$ и все пары токенов, образующие простой редекс, $(@,\lambda x):\alpha(\lambda x) = @$, т.е. $\{(1,4),(2,3),(5,9),(6,8),(10,12),(13,17),(14,16),(18,20)\}$, тогда полученный обход, в нашем примере $\underset{22}{@_9}\ \underset{23}{@_8}\ \underset{24}{g}\ \underset{25}{\lambda b}\ \underset{26}{b}\ \underset{28}{@_5}\ \underset{30}{@_{11}}\ \underset{31}{g}\ \underset{32}{\lambda n}\ \underset{33}{n}\ \underset{34}{a}$, будет обходом в глубину нормальной формы исходного терма, а токены || обозначают конец построения его текущей ветки.

Литература

1. *Daniil Berezun, Neil D. Jones*. Compiling untyped lambda calculus to lower-level code by game semantics and partial evaluation (invited paper). In Proceedings of the 2017 ACM SIGPLAN Workshop on Partial Evaluation and Program Manipulation (PEPM 2017), ACM, New York, NY, USA, 1 – 11, 2017.
2. *Luke Ong C.-H.* Normalisation by Traversals. CoRR, abs /1511.02629. URL: http://arxiv.org/abs/1511.02629, 2015 (дата обращения 27.07.2017).
3. *William Blum*. The safe lambda calculus. PhD thesis, University of Oxford, UK, 2009.
4. *Luke Ong C.-H.* On Model-Checking Trees Generated by Higher-Order Recursion Schemes. In Logic in Computer Science, 2006 21st Annual IEEE Symposium on, pages 21 – 90, 12 – 15 Aug. 2006.
5. *Blum W., Luke Ong C.-H.* A concrete presentation of game semantics. In Galop 2008: Games for Logic and Programming Languages, 2008.
6. *Березун Д.* Полная головная линейная редукция // Науч-техн. ведомости СПбГПУ. Информатика. Телекоммуникации. Управление. Вып. 3. 2017.
7. *Vincent Danos, Laurent Regnier*. Head linear reduction. unpublished, 2004.
8. *Jean-Louis Krivine.* A call-by-name lambda-calculus machine. Higher-Order and Symbolic Computation 20(3): 199 – 207, 2007.

9. *Blum W.* Imaginary Traversals for the Untyped Lambda Calculus (ongoing work), 2017.
10. *Hendrik Pieter Barendregt*. The lambda calculus: its syntax and semantics. Studies in logic and the foundations of mathematics. North-Holland, Amsterdam, New-York, Oxford, 1981.
11. *Benjamin C. Pierce*. Types and Programming Languages (1st ed.). The MIT Press, 2002.
12. *Benjamin C. Pierce*. Advanced Topics in Types and Programming Languages. The MIT Press, 2004.

References

1. Berezun Daniil, Jones Neil D. Compiling untyped lambda calculus to lower-level code by game semantics and partial evaluation (invited paper). In Proceedings of the 2017 ACM SIGPLAN Workshop on Partial Evaluation and Program Manipulation (PEPM 2017), ACM, New York, NY, USA, 1-11, 2017.
2. Luke Ong C.-H. Normalisation by Traversals. CoRR, abs/1511.02629, 2015. http://arxiv.org/abs/1511.02629 (accessed 27.07.2017)
3. Blum William. The safe lambda calculus. PhD thesis, University of Oxford, UK, 2009.
4. Luke Ong C.-H. On Model-Checking Trees Generated by Higher-Order Recursion Schemes. In Logic in Computer Science, 2006 21st Annual IEEE Symposium on, 2006. Pp. 21–90.
5. Blum W., Luke Ong C.-H. A concrete presentation of game semantics. In Galop 2008: Games for Logic and Programming Languages, 2008.
6. Berezun D.. Polnaya golovnaya lineinaya reduktsiya [Full head linear reduction]. *Nauchno-tekhnicheskie vedomosti SPbGPU. Informatika. Telekommunikatsii. Upravlenie*, 2017, vyp. 3.
7. Vincent Danos and Laurent Regnier. Head linear reduction. unpublished, 2004.
8. Jean-Louis Krivine. A call-by-name lambda-calculus machine. Higher-Order and Symbolic Computation 20(3): 199-207, 2007.
9. Blum W. Imaginary Traversals for the Untyped Lambda Calculus (ongoing work), 2017.
10. Hendrik Pieter Barendregt. The lambda calculus: its syntax and semantics. Studies in logic and the foundations of mathematics. North-Holland, Amsterdam, New-York, Oxford, 1981.
11. Pierce Benjamin C. Types and Programming Languages (1st ed.). The MIT Press, 2002.
12. Pierce Benjamin C. Advanced Topics in Types and Programming Languages. The MIT Press, 2004.

Поступила в редакцию / Received ***30 сентября 2017 г. / September 30, 2017***