На правах рукописи

Березун Даниил Андреевич

ТРАССИРУЮЩАЯ НОРМАЛИЗАЦИЯ

Специальность 05.13.11 —
«Математическое и программное обеспечение вычислительных машин, комплексов и компьютерных сетей»

Автореферат

диссертации на соискание учёной степени

кандидата физико-математических наук

Санкт-Петербург
2017

Работа выполнена на кафедре системного программирования Санкт-Петербургского государственного университета

Научный руководитель: **КОЗНОВ Дмитрий Владимирович**, доктор технических наук, доцент, профессор кафедры системного программирования

Официальные оппоненты: **НЕПЕЙВОДА Николай Николаевич**,
доктор физико-математических наук, профессор,
Федеральное государственное бюджетное учреждение науки Институт программных систем имени А.К. Айламазяна Российской академии наук, главный научный сотрудник

МИРОНОВ Андрей Михайлович,
кандидат физико-математических наук,
Федеральное государственное бюджетное образовательное учреждение высшего образования "Московский государственный университет имени М.В.Ломоносова", доцент

Ведущая организация: Федеральное государственное учреждение "Федеральный исследовательский центр Институт прикладной математики им. М.В. Келдыша Российской академии наук" (ИПМ им. М.В. Келдыша РАН)

Защита состоится 29 марта 2018 г. в 14 часов на заседании диссертационного совета Д 212.232.51 на базе Санкт-Петербургского государственного университета по адресу: 198504, Санкт-Петербург, Старый Петергоф, Университетский пр. 28, математико-механический факультет СПбГУ, ауд. 405.

С диссертацией можно ознакомиться в библиотеке Санкт-Петербургского государственного университетапо адресу: 199034, Санкт-Петербург, Университетская наб., д. 7/9, а также на сайте: https://disser.spbu.ru/disser/soiskatelyu-uchjonoj-stepeni/dis-list/form/14/1545.html.

Автореферат разослан __ _____ 20__ года.

Ученый секретарь
диссертационного совета
Д 212.232.51, д.ф.-м.н., профессор Демьянович Юрий Казимирович

Общая характеристика работы

Актуальность темы исследования

Лямбда-исчисление, наряду с машинами Тьюринга, теорией частично-рекурсивных функций, формальными алгорифмами Маркова и Поста, является одной из основных формальных моделей вычислений. Первая модель лямбда-исчисления, предложенная A. Church в 30-х годах 20 века, оказалась противоречивой: в 1935 году S. Kleene и J. B. Rosser выявили в ней парадокс, названный в их честь. В 1936 году A. Church предложил первую непротиворечивую систему, ныне известную как *нетипизированное*, *бестиповое* или *чистое* лямбда-исчисление, а в 1940 году он же предложил модель простейшего типизированного исчисления — *простое типизированное лямбда-исчисление*. С тех пор лямбда-исчисление играет особую роль в теории вычислимости и стало основой для целой парадигмы программирования — функционального программирования.

Тем не менее, различные языки, основанные на лямбда-исчислении, долгое время не имели *полностью абстрактных моделей* вычислений, т.е. таких денотационных моделей, что существует изоморфизм между термами языков и их представлениями в этих моделях. Впервые задача построения полностью абстрактной модели вычислений была поставлена в 1975 году (G. Plotkin) для языка программирования вычислимых функций PCF (Programming Computable Functions). Первым подходом, предложившим решение поставленной задачи, стала *игровая семантика*, на основе которой были построены полностью абстрактные модели для целого спектра языков программирования.

Игровая семантика определяет как *денотационную*, так и *операционную* модели языка как некоторое взаимодействие, игру, между программой и её окружением. Недавние работы (C.-H. L. Ong и W. Blum) показали, что игровая семантика способна нормализовывать лямбда-термы простого типизированного лямбда-исчисления без использования стандартных подходов, таких как β-редукции и окружения. Стратегия вычислений, основанная на данном подходе, получила название *головная линейная редукция* (Head Linear Reduction), а сам подход — *трассирующая нормализация* (Traversal-Based Normalizarion). Головная линейная редукция играет особую роль в различных подходах к вычислениям, таких как оптимальные редукции, геометрия взаимодействия, сети доказательств и др.

Тем не менее, до сих пор оставался открытым вопрос расширения подхода трассирующей нормализации до исчисления, полного по Тьюрингу. Такое расширение определило бы свежий взгляд на функциональные языки программирования, позволяя исследовать их свойства с нового ракурса.

Степень разработанности темы исследования

Существует множество работ, посвящённых лямбда-исчислению и его свойствам, начиная с классических работ A. Church, A. M. Turing и H. B. Curry. Эти работы были посвящены основам математики и математической логики, и, в частности, теориям сложности и вычислимости. Значительный вклад в теорию лямбда-исчисления внёс голландский математик H. P. Barendregt, предложив так называемый лямбда-куб. Множество отечественных и зарубежных учёных изучали свойства языков, основанных на лямбда-исчислении, в том числе D. S. Scott и C. S. Strachey построили денотационные модели для целого ряда языков программирования. Тем не менее денотационные модели некоторых языков программирования не обладали свойством полной абстракции (Full-Abstraction Property), задачу построения которых сформулировал G. Plotkin.

В 50-х годах 20 века P. Lorenzen предложил так называемую игровую семантику для логики, которая получила развитие в работах немецкого философа K. Lorenz. Дальнейшее развитие этого подхода такими учёными как S. Abramsky, C.-H. L. Ong, P. Malacaria, R. Jagadeesan и G. McCusker позволило в начале 90-х годов решить задачу, сформулированную G. Plotkin. Дальнейшее развитие игровая семантика получила в работах J. M. E. Hyland, A. D. Ker, H. Nickau, W. Blum и др.

В последние годы исследование операционных особенностей игрой семантики стало актуальной темой исследований, о чём свидетельствуют работы D. R. Ghica, L. Ong и W. Blum. V. Danos и L. Regnier описывали игровую семантику программ с помощью абстрактных машин KAM (Krivine Abstract Machine) и PAM (Pointer Abstract Machine), а также предложили связать её с линейной редукцией. L. Ong впервые определил трассирующую нормализацию для простого типизированного лямбда-исчисления, а W. Blum распространил этот подход до безопасного лямбда-исчисления (Safe Lambda-Calculus). Тем не менее, вопрос распространения подхода трассирующей нормализации до полного по Тьюрингу исчисления оставался открытым.

Целью данной работы является обобщение подхода трассирующей нормализации до полного по Тьюрингу исчисления, обоснование корректности трассирующей нормализации и исследование возможности представления различных языковых конструкций в рамках предложенного подхода при компиляции функциональных языков программирования.

Для достижения вышеупомянутой цели были поставлены следующие **задачи**.

1. Разработать в рамках стратегии вычисления нормального порядка редукций алгоритм трассирующей нормализации для нетипизированного лямбда-исчисления.
2. Формально доказать корректность предложенного алгоритма.
3. Исследовать возможность адаптации трассирующей нормализации для различных стратегий вычислений.
4. Исследовать возможность представления различных языковых конструкций в рамках предложенного подхода с точки зрения компиляции основных конструкций функциональных языков программирования.

Постановка цели и задач исследования соответствует следующим пунктам паспорта специальности 05.13.11: методы, модели и алгоритмы проектирования и анализа программ и программных систем, их эквивалентных преобразований, верификации и тестирования (пункт 1); языки программирования и системы программирования, семантика программ (пункт 2).

Методология и методы исследования

Методология исследования основана на идеях и подходах информатики к описанию и анализу понятия вычислимости. Используются также операционная и денотационная концепции описания семантик языков программирования.

В работе использовались методы синтаксического преобразования лямбда-термов. Для формализации моделей использовался подход к описанию семантик в виде систем переходов, а для доказательства их корректности — метод симуляции одной системы переходов другой посредством определения отношения, связывающего соответствующие состояния этих систем переходов. Были использованы стратегии вычислений вызова по имени, нормального порядка, вызова по значению, аппликативного порядка, головной редукции, линейной редукции и вызова по необходимости. Для автоматического преобра-

зования программ и семантик использовались методы частичных вычислений и дефункционализации по Reynolds. Программная реализация предложенных в диссертации результатов была выполнена с помощью языков программирования Haskell и Racket.

Основные положения, выносимые на защиту

1. Разработан алгоритм трассирующей нормализации для нетипизированного лямбда-исчисления, соответствующий нормальному порядку редукций.
2. Представлена модель полной головной линейной редукции, являющаяся расширением известной модели головной линейной редукции. Предложенная модель формализована в виде системы переходов, доказана корректность этой модели относительно головной редукции.
3. Доказана корректность представленного алгоритма трассирующей нормализации относительно предложенной модели полной головной линейной редукции путём его формализации в виде системы переходов и дальнейшей симуляции системы переходов для полной головной линейной редукции. Таким образом, было доказано, что предложенная процедура трассирующей нормализации действительно является нормализирующей.
4. Предложенный алгоритм адаптирован для других, отличных от нормального порядка, стратегий вычислений: аппликативного порядка редукций и вызова по необходимости.
5. Предложен и реализован на примере нетипизированного лямбда-исчисления новый метод компиляции функциональных языков программирования в низкоуровневое представление путём специализации представленного алгоритма трассирующей нормализации на входной терм.

Научная новизна полученных в ходе исследования результатов заключается в следующем.

1. Впервые было введено и формально описано понятие полной головной линейной редукции и доказана её корректность.
2. Алгоритм трассирующей нормализации, представленный в работе, отличается от аналогов (работы L. Ong и W. Blum) тем, что он не накладывает каких-либо ограничений на лямбда-термы, что позволяет

корректно нормализовывать произвольный лямбда-терм. Кроме того, алгоритмы, предложенные L. Ong и W. Blum, применяются лишь к частным случаям, а именно, к простому типизированному лямбда-исчислению и безопасному лямбда-исчислению.

3. Предложен новый подход к компиляции функциональных языков программирования путём специализации процедуры трассирующей нормализации на входной терм, и исследованы её свойства.

Теоретическая значимость диссертационного исследования заключается в формализации понятий головной и полной головной линейной редукций, разработке формального алгоритма, решающего задачу трассирующей нормализации, соответствующей полной головной линейной редукции, для бестипового лямбда-исчисления, а также в формальном доказательстве корректности представленного алгоритма.

Практическая значимость

Трассирующая нормализация представляет собой новый подход к вычислениям, позволяя исследовать ряд свойства языков программирования с нового ракурса. Как показано в диссертации, применение частичных вычислений к предложенному в диссертации алгоритму трассирующей нормализации позволяет производить трансляцию лямбда-термов в низкоуровневое представление, сохраняя исходную семантику программы. Такой подход к компиляции является перспективным направлением в области автоматической генерации компиляторов на основе семантики (Semantics-Directed Compiler Generation). Описание известных языковых конструкций и программных трансформаций при помощи нового подхода открывает перспективы к проектированию языка промежуточного представления, являющегося базой для описания семантик широкого класса языков программирования и позволяющего автоматически генерировать компиляторы соответствующих языков.

Степень достоверности и апробация результатов

Достоверность и обоснованность результатов исследования опирается на использование формальных методов исследуемой области, выполнение формальных доказательств и инженерные эксперименты.

Основные результаты работы были представлены на следующих научных конференциях и семинарах: семинар “Математические вопросы информатики” (27 октября 2017, МГУ им. М.В. Ломоносова, мех.-мат. ф-т, Москва, Рос-

сия), семинар в ИПС им. А.К. Айламазяна РАН (26 октября 2017, Переславль-Залесский, Россия), семинар в ИПМ им. М.В. Келдыша РАН (25 октября 2017, Москва, Россия), конференция “Языки программирования и компиляторы” (PLC 2017, 3–5 апреля 2017, Ростов-на-Дону, Россия), Games for Logic and Programming Languages XII (GaLoP 2017, 22-23 апреля 2017, Уппсала, Швеция), PEPM 2017 Workshop on Partial Evaluation and Program Manipulation (16 – 17 января, 2017, Париж, Франция), внутрениие семинары Department of Computer Science of Oxford University (февраль 2016, апрель 2016, март 2017, Оксфорд, Великобритания), Fifth International Valentin Turchin Workshop on Metacomputation (Meta 2016, 27 июня – 1 июля, 2016, Переславль-Залесский, Россия), Games for Logic and Programming Languages XI (GaLoP 2016, 2-3 апреля 2016, Эйндховен, Нидерланды) и внутренние семинары Department of Computer Science of DIKU (октябрь-ноябрь 2015 и ноябрь 2017, Копенгаген, Дания).

Публикации по теме диссертации

Все результаты диссертации опубликованы в пяти печатных работах, зарегистрированных в РИНЦ. Две единоличные статьи изданы в журналах из “Перечня российских рецензируемых научных журналов, в которых должны быть опубликованы основные научные результаты диссертаций на соискание учёных степеней доктора и кандидата наук”. Две статьи опубликованы в изданиях, входящих в базы цитирования SCOPUS и Web of Science.

Вклад автора в публикациях, написанных в соавторстве, распределён следующим образом. В работе [3] автору принадлежит формализация метода трассирующей нормализации, реализация предложенных семантических преобразований, идея и формализация метода трассирующей нормализации для языка PCF. Соавторы предложили схему представления нормализирующей процедуры, описали применение частичных вычислений к процедуре нормализации с целью компиляции термов лямбда-исчисления, сформулировали направления дальнейших исследований. В работе [4] автору принадлежит реализация инструментальных средств, разработка и реализация модели представления программной кучи и проведение экспериментов.

Объем и структура работы

Диссертация состоит из введения, 6 глав, заключения и 2 приложений. Полный объем диссертации составляет 130 страниц, включая 34 рисунка и 1 таблицу. Список литературы содержит 107 наименований.

Содержание работы

Во **введении** обосновывается актуальность исследований, выполненных в рамках данной диссертационной работы, приводится обзор научной литературы по изучаемой проблеме, формулируется цель, ставятся задачи работы, формулируются научная новизна и практическая значимость.

В **первой главе** приводится обзор области исследования. Рассматриваются существующие подходы к нормализации лямбда-термов и, в том числе, к трассирующей нормализации. Описывается игровая семантика и алгоритмы трассирующей нормализации для простого типизированного лямбда-исчисления и безопасного лямбда-исчисления, предложенные C.-H. L. Ong и W. Blum соответственно. Также показано, что предложенные подходы трассирующей нормализации соответствуют известной модели головной линейной редукции, предложенной L. Regnier и V. Danos. На основе проделанного обзора сделаны следующие выводы.

- Существующие алгоритмы трассирующей нормализации определены лишь для некоторых неполных по Тьюрингу исчислений. При этом ориентированность этих алгоритмов на системы типов и статические ограничения на структуру исходного терма осложняют их обобщение для поддержки исчислений, полных по Тьюрингу.
- Задача построения алгоритмов трассирующей нормализации для нетипизированного лямбда-исчисления для стратегий вычислений нормального и аппликативного порядков, а также стратегии вызова по необходимости, является актуальной задачей в области анализа языков программирования.
- Алгоритмы трассирующей нормализации требуют формального операционного доказательства корректности.

Вторая глава содержит описание предложенной автором модели полной головной линейной редукции. Она является расширением известной модели головной линейной редукции, предложенной V. Danos и L. Regnier. Обе модели основаны на понятиях простого редекса и линейной редукции.

Определение. *Хребетными* или *спинальными* подтермами (Spine Sub-Terms) лямбда-терма T называются сам терм T , а также все спинальные подтермы терма U , если $T = UV$ или $T = \lambda x.U$ соответственно. Заметим, что любой лямбда-терм имеет ровно один спинальный подтерм-переменную. Такое

Терм T	Условие	$\lambda_h(T)$	$pr(T)$
x		$[\,]$	$[\,]$
UV	$\lambda_h(U) = [\,]$	$[\,]$	$pr(U)$
UV	$\lambda_h(U) = \lambda x : l$	l	$(\lambda x, V) : pr(U)$
$\lambda x.U$		$\lambda x : \lambda_h(U)$	$pr(U)$

Рис. 1. Определение списков головных абстракций и простых редексов.

вхождение переменной, так же как и сам подтерм, называется головным (Head Occurrence, HOC).

Определение. Списки *головных абстракций* (Head λ List, $\lambda_h(T)$) и *простых редексов* (Prime Redexes) определяются совместно индукцией по структуре терма T. *Простой редекс* представляет собой пару $(\lambda x, N)$, первый и второй элементы которой называются абстракцией и аргументом простого редекса соответственно. Индуктивное определение списков простых редексов и головных абстракций приведены на рис. 1, где $pr(T)$ обозначает список простых редексов терма T.

Редексом головного вхождения переменной (HOC Redex) терма T называется простой редекс $(\lambda x, V)$, где x является головной переменной, а V — соответствующий аргумент, если таковой редекс существует.

Интуитивно, головная линейная редукция линеаризует последовательность подстановок путём замены на каждом шаге лишь одного вхождения переменной — головного — оставляя сам головной редекс нетронутым. Если же он не определён, головная линейная редукция завершается в так называемой *псевдоголовной нормальной форме* (Quasi-Head-Normal Form, QHN).

В данной главе предлагается модель *полной головной линейной редукции*, являющаяся обобщением модели головной линейной редукции. Приведены формальное представление обеих моделей в виде систем переходов, причём системы переходов для головной линейной редукции является частным случаем системы переходов для полной головной линейной редукции. Доказаны следующие теоремы о корректности моделей.

Теорема 2 устанавливает соответствие между представленной моделью головной линейной редукцией и головной редукцией: 1) если головная линейная редукция завершается, то её результат находится в псевдо-головной нормальной форме; 2) головная линейная редукция завершается тогда и только тогда, когда завершается головная редукция.

Теорема 3 устанавливает соответствие между предложенной моделью полной головной линейной редукцией и головной редукцией: 1) если полная головная линейная редукция завершается, то её результат находится в нормальной форме; 2) полная головная линейная редукция завершается тогда и только тогда, когда завершается головная редукция.

Обе теоремы доказаны методом симуляции соответствующей системой переходов головной редукции лямбда-термов.

В **третьей главе** приводится разработанный автором алгоритм трассирующей нормализации для нетипизированного лямбда-исчисления. Предлагаемый алгоритм реализует стратегию нормального порядка редукции. Отличительной особенностью трассирующей нормализации является то, что в то время как классическая β-редукция путём произведения подстановок изменяет исходный терм, трассирующая нормализация, напротив — оставляет его нетронутым. В главе приведено обобщение трассирующей нормализации до нетипизированного лямбда-исчисления, являющегося истинным надмножеством вышеуказанных исчислений.

Трассирующая нормализация нормализует лямбда-терм путём построения *обхода* его абстрактного синтаксического дерева (АСД).

Определение. *Обоснованной последовательностью* (Justified Sequence) называется упорядоченная последовательность, каждый элемент которой может быть снабжён одним или несколькими указателями на более ранние элементы.

Определение. *Обходом* (Traversal) АСД лямбда-терма называется обоснованная последовательность его вершин, именуемых *токенами*.

Обратные указатели используются для определения следующего токена при построении обхода. Предложенный автором алгоритм трассирующей нормализации манипулирует двумя видами указателей: указателем стека висячих аргументов, изображаемым над обходом на диаграммах, и обобщённым связывающим указателем, изображаемым на диаграммах под обходом. Интуитивно, цепочка обобщённых связывающих указателей определяет на каждом шаге множество связанных переменных, в то время как указатели стека висячих аргументов позволяют определять простые редексы. На рис. 2 приведён пример АСД терма и его обхода.

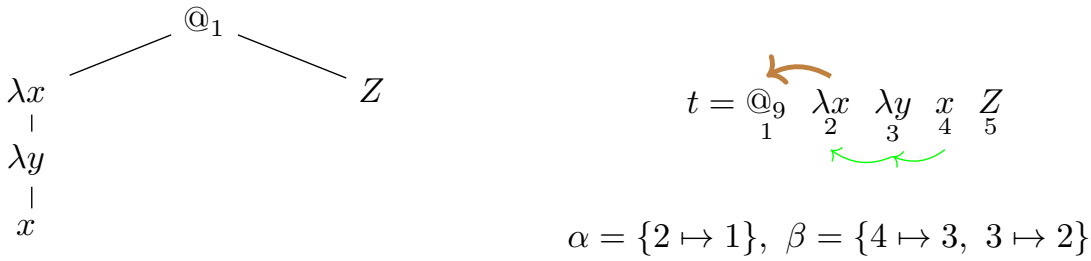

Рис. 2. АСД терма $(\lambda x.\lambda y.x)@_1 Z$ и его обход.

Алгоритм построения обхода АСД терма является детерминированным. По его завершению из конечного обхода может быть однозначно восстановлена нормальная форма исходного терма. А именно, после удаления незначимых токенов из обхода оставшаяся последовательность токенов является обходом в глубину АСД нормальной формы исходного терма.

В **четвёртой главе** доказана корректность трассирующей нормализации. Для этого приведено формальное представление процедуры нетипизированной трассирующей нормализации (UNP) и её ограниченной версии (BUNP — ограниченная процедура нетипизированной трассирующей нормализации) в виде систем переходов.

Пусть $\mathbb{V} \coloneqq \{a,b,c,\dots\}$ — бесконечный алфавит имён переменных, Σ — следующий алфавит символов:

$$\Sigma \coloneqq \{@_i \mid i \in \mathbb{N}\} \cup \{x \mid x \in \mathbb{V}\} \cup \{\lambda x \mid x \in \mathbb{V}\}.$$

Состоянием системы переходов для трассирующей нормализации является текущий обход, формально, тройка $\langle t;\ \alpha;\ \beta\rangle$, где $t : \Sigma \to [1..|t|]$ является упорядоченной последовательностью, $|t|$ — длина последовательности t, а частичные функции $\alpha, \beta : [1..|t|] \rightharpoonup [1..|t|-1]$ определяют множества обратных указателей.

Корректность обеих процедур относительно головной и полной головной линейных редукций соответственно доказана в теоремах 5 и 6 путём симуляций системы переходов для головной линейной редукции системой переходов для BUNP и системы переходов для полной головной линейной редукции системой переходов для UNP.

В **пятой главе** приведён новый метод компиляции термов нетипизированного лямбда-исчисления в низкоуровневое представление, обладающий свойством полу-композициональности: каждый аргумент семантической функции является подтермом вычисляемого лямбда-терма. Таким образом, аргумент

семантической функции может принимать конечное заранее заданное множество значений (всевозможных подтермов исходного терма). Полукомпозициональность позволяет успешно специализировать процедуру трассирующей нормализации на входной лямбда-терм.

Пусть процедура нормализации реализована в виде программы NP на языке L. Тогда, применяя вторую проекцию Футамуры–Ершова–Турчина, можно преобразовать программу NP в программу на том же языке L, производящую трансляцию термов из лямбда-исчисления в некоторый язык LLL, являющийся подмножеством языка L.

$$\text{Если NP} \in \boxed{\begin{matrix}\text{LC}\\ \text{L}\end{matrix}}\text{, то } [\![spec]\!](spec, \text{NP}) \in \boxed{\begin{matrix}\text{LC} \rightarrow \text{LLL}\\ \text{L}\end{matrix}}.$$

Здесь LC — некоторое лямбда-исчисление (например, чистое или простое типизированное), $spec$ – частичный вычислитель, написанный на языке L, LLL $\subset$ L — язык низкоуровневого представления.

Шестая глава посвящена распространению подхода трассирующей нормализации для стратегий вычислений аппликативного порядка и вызова по необходимости. Показано, как трассирующая нормализация может быть использована для компиляции конструкций языка PCF.

В **заключении** приводятся основные результаты, полученные в рамках диссертационного исследования.

1. Разработан алгоритм трассирующей нормализации для нетипизированного лямбда-исчисления, соответствующий нормальному порядку редукций.
2. Представлена модель полной головной линейной редукции, являющаяся расширением известной модели головной линейной редукции. Предложенная модель формализована в виде системы переходов, доказана её корректность относительно головной редукции.
3. Доказана корректность представленного алгоритма трассирующей нормализации относительно предложенной модели полной головной линейной редукции. Для этого сам алгоритм трассирующей нормализации формализован в виде системы переходов, а также приведено отношение симуляции, с помощью которого поведение системы переходов для алгоритма трассирующей нормализации может быть симулировано системой переходов для полной головной линейной ре-

дукции. Таким образом, доказано, что процедура трассирующей нормализации является нормализирующей.

4. Предложенный алгоритм адаптирован для других, отличных от нормального порядка, стратегий вычислений: аппликативного порядка редукций и вызова по необходимости.
5. Предложен новый метод компиляции функциональных языков программирования в низкоуровневое представление путём специализации представленного алгоритма трассирующей нормализации на входной терм.
6. Разработана экспериментальная реализация алгоритма трассирующей нормализации нетипизированного лямбда-исчисления[1], а также генерирующего расширения, на ней основанного, для компиляции лямбда-термов в низкоуровневое представление, на языках Haskell и Racket.

В качестве **рекомендации по применению результатов работы** в индустрии и научных исследованиях указывается, что предложенный алгоритм трассирующей нормализации не преобразует исходный терм, а только посещает его в конечном числе точек, благодаря чему он успешно поддается специализации с помощью хорошо известных методов специализации программ. В частности, применяя вторую проекцию Футамуры–Ершова–Турчина (специализируя специализатор на алгоритм трассирующей нормализации) становится возможной генерация компиляторов для языков программирования.

В качестве **перспектив дальнейшей разработки тематики** может быть отмечено следующее: (1) описание известных языковых конструкций и программных трансформаций с помощью подхода трассирующей нормализации; (2) разработка, на основе предложенного подхода, универсального языка промежуточного представления, который может служить базой для описания семантик широкого класса языков программирования, позволяя автоматически генерировать компиляторы соответствующих языков, а также (3) исследование свойств вычислимости и сложности программ в рамках подхода трассирующей нормализации.

[1] Исходный код доступен по ссылке: https://github.com/danyaberezun/traversal-Based-Normalization.

Публикации автора по теме диссертации

Ниже приведён перечень публикаций, где были представлены основные результаты, представленные автором в диссертации.

Публикации из "Перечня рецензируемых научных изданий, в которых должны быть опубликованы основные научные результаты диссертаций на соискание ученой степени кандидата наук, на соискание ученой степени доктора наук", сформированного согласно требованиям, установленным Министерством образования и науки Российской Федерации

1. **Березун, Д.А. Полная головная линейная редукция / Д.А. Березун // НТВ СПбГПУ. Информатика. Телекоммуникации. Управление. – 2017. – № 3. – С. 59–82.**
2. **Березун, Д.А. Трассирующая нормализация нетипизированного лямбда-исчисления. / Д.А. Березун // Известия вузов. Северо Кавказский регион. Технические науки. – 2017. – № 4. – С. 5–12.**

Публикации в изданиях, входящих в базы цитирования Web of Science и SCOPUS

3. Berezun, D.A. Compiling Untyped Lambda Calculus to Lower-Level Code by Game Semantics and Partial Evaluation (invited paper) / D.A. Berezun, N.D. Jones // In Proceedings of the 2017 ACM SIGPLAN Workshop on Partial Evaluation and Program Manipulation (PEPM 2017). ACM, New York, NY, USA. – 2017. – С. 1–11.
4. Berezun, D.A. Precise Garbage Collection for C++ with a Non-Cooperative Compiler / D.A. Berezun, D.Y. Boulytchev // Proceedings of the 10th Central and Eastern European Software Engineering Conference in Russia. –2014. С. 15:1–15:8.

Публикации по теме диссертации в других изданиях

5. Berezun, D.A. Working Notes: Compiling ULC to Lower-level Code by Game Semantics and Partial Evaluation. / D.A. Berezun, N.D. Jones // META 2016 Fifth International Valentin Turchin Workshop on Metacomputation. – 2016. – С. 11–23.