Предложения как типы

Филип Уодлер

Эдинбургский университет

wadler@inf.ed.ac.uk

1. Введение

Мощные озарения возникают в результате объединения двух областей исследования, которые ранее считались отдельными. Примеры включают систему координат Декарта, которая связывает геометрию с алгеброй, Квантовую теорию Планка, которая связывает частицы с волнами, и Теорию информации Шеннона, которая связывает термодинамику с коммуникацией. Такой синтез предлагается принципом предложений как типов, который связывает логику с вычислениями. На первый взгляд это кажется простым совпадением — почти каламбуром, — но оно оказывается на удивление надежным, вдохновляя на разработку автоматизированных помощников по проверке и языков программирования, и продолжая оказывать влияние на передовые направления вычислительной техники.

Предложения как типы - это понятие со многими названиями и множеством истоков. Это тесно связано с интерпретацией ВНК, взглядом на логику, разработанным интуиционистами Брауэром, Хейтингом и Кол-Могоровым в 1930-х годах. Его часто называют изоморфизмом Карри-Ховарда , ссылаясь на соответствие, наблюдаемое Карри в 1934 году и уточненное Говардом в 1969 году (хотя и не публиковалось до 1980 года в Festschrift, посвященном Карри). Другие привлекают внимание к значительному вкладу автомата де Брейна и Теории типов Мартина-Лофа в 1970-х годах. В литературе встречается множество вариантов названий, включая Формулы как типы, соответствие Карри-Ховарда-де Брейна, Изречение Брауэра и другие.

Предложения как типы - понятие глубокое. Оно описывает соответствие между данной логикой и данным языком программирования . На первый взгляд, это говорит о том, что для каждого предложения в логике существует соответствующий тип в языке программирования — и наоборот. Таким образом, мы имеем

предложения как типы.

Это происходит глубже, поскольку для каждого доказательства данного предложения существует программа соответствующего типа — и наоборот. Таким образом, у нас также есть

доказательства как программы.

И это еще глубже, поскольку для каждого способа упрощения доказательства существует соответствующий способ оценки программы — и наоборот . Таким образом, мы далее имеем

упрощение доказательств как оценка программ.

Следовательно, мы имеем не просто неглубокую биекцию между предложениями и типами, но истинный изоморфизм, сохраняющий глубокую структуру доказательств и программ, упрощения и оценки.

Предложения как типы - это понятие, обладающее широтой. Это применимо к целому ряду логик, включая пропозициональную, предикатную, второго порядка, интуиционистскую, классическую, модальную и линейную. Он лежит в основе основ функционального программирования, объясняя особенности, включая функции, записи, варианты, параметрический полиморфизм, структуру данных, продолжения, линейные типы и типы сеансов. В него встроены автоматизированные помощники проверки и языки программирования, включая Agda, Automath, Coq, Epigram, F#, F, Haskell, LF, ML, NuPRL, Scala, Singularity и Trellys.

Предложения как типы - это понятие, обладающее тайной. Почему должно быть так, что интуиционистская естественная дедукция, разработанная Гентцен в 1930-х годах и простотипизированное лямбда-исчисление, разработанное Черчем примерно в то же время для несвязанных целей, должны ли быть обнаружены тридцать лет спустя как по существу идентичные? И почему должно быть так, что одно и то же соответствие возникает снова и снова? Логик Хиндли и специалист по информатике Милнер независимо разработали однотипную систему, теперь получившую название Хиндли-Милнер. Логик Джирард и специалист по информатике Рейнольдс независимо разработали одно и то же математическое моделирование, теперь получившее название Джирард-Рейнольдс. Карри-Ховард - это двуствольное имя, которое гарантирует существование других двуствольных имен. Тем из нас, кто разрабатывает и использует языки программирования, часто может казаться, что они произвольны, но предложения в виде типов гарантируют нам абсолютность некоторых аспектов программирования.

Онлайн-приложение содержит этот документ полностью с дополнительными деталями и ссылками, а также историческую заметку, предоставленную Уильямом Говардом. (Версия, которую вы читаете, является онлайновым приложением.)

Этот документ служит кратким введением в пропозиции как типы. Для тех, кто заинтересован узнать больше, доступны варианты изложения в учебниках [23, 59, 56].

2. Черч и теория вычислений

Истоки логики восходят к Аристотелю и стоикам в классической Греции, Оккаму и схоластам в средние века, а также к видению Лейбницем математического аппарата на заре просвещения. Наш интерес к этому предмету связан с формальной логикой, которая возникла благодаря вкладам Буля, Де Моргана, Фреге, Пирса, Пеано и других в 19 веке.

На заре 20-го века Принципы Уайтхеда и Рассела Mathematica [66] продемонстрировал, что формальная логика может выражать большую часть математики. Вдохновленные этим видением, Гильберт и его коллеги из Геттингена стали ведущими сторонниками формальной логики, стремясь поставить ее на прочную основу.

Одной из целей программы Гильберта было решить проблему Entschei-dungsproblem (проблема принятия решения), то есть разработать “эффективную вычисляемую” процедуру для определения истинности или ложности любого утверждения. Проблема предполагает полноту: для любого утверждения либо оно, либо его отрицание обладают доказательством. В своем обращении к Математическому конгрессу 1930 года в Кенигсберге Гильберт подтвердил свою веру в этот принцип, заключив “Wir mussen wis-sen, wir werden wissen” (“Мы должны знать, мы узнаем”), слова, позже выгравированные на его надгробии. Возможно, надгробие - это подходящее место для этих слов, учитывая, что любые основания для оптимизма Гильберта были подорваны днем ранее, когда на той же конференции Гедель [24] объявил о своем доказательстве того, что арифметика неполная.

Хотя цель состояла в том, чтобы удовлетворить программе Гильберта, точного определения понятия “эффективно вычисляемый” не требовалось. Было бы ясно, эффективна ли данная процедура или нет, как, например, Джастис Стью-характеристика непристойности Арта: “Я узнаю это, когда вижу”. Но чтобы показать неразрешимость Entscheidungsproblem, требовалось формальное определение “эффективно вычисляемого”.

Можно найти упоминания о концепции алгоритма в работах Евклида и, соответственно, аль-Хорезми, но эта концепция была формализована только в 20 веке, а затем одновременно получила три независимых определения от логиков. Как автобусы: вы ждете две тысячи лет определения понятия “эффективно вычисляемый”, а затем появляются сразу три. Все три были лямбда-вычислениями с помощью опубликованная в 1936 году Алонзо Черчем [9], рекурсивные функции, предложенные Геделем на лекциях в Принстоне в 1934 году и опубликованные в 1936 году Стивеном Клином [35], и машины Тьюринга, опубликованные в 1937 году Аланом Тьюрингом [60].

Лямбда-исчисление было введено Черчем в Принстоне и далее развито его учениками Россером и Клином. В это время Принстон соперничал с Геттингеном как центр изучения логики. Институт перспективных исследований располагался совместно с математическим факультетом в Файн-Холле. В 1933 году Эйнштейн и фон Нейман присоединились к Институту, и Гедель приехал их навестить.

Логики уже давно озабочены идеей функции. Лямбда-исчисление предоставляет краткое обозначение функций, включая “первоклассные” функции, которые могут отображаться как аргументы или результаты других функций. Он удивительно компактен и содержит всего три конструкции: переменные, абстракцию функции и приложение функции . Черч [7] сначала представил лямбда-исчисление как способ определения точных обозначений для логических формул (почти как макроязык) в новом представлении логики. Все формы связанной переменной могут быть отнесены к лямбда-привязке. (Например, вместо ∃x. A[x], Черч написал Σ(λx. A[x]).) Однако позже Клини и Россер обнаружили [38], что система Церкви была непоследовательной. К этому времени Черч и его ученики поняли, что эта система представляет самостоятельный интерес. Черч предвидел такую возможность в своей первой статье на эту тему, где он писал: “Действительно, у системы могут быть другие применения, кроме ее использования в качестве логики”.

Черч открыл способ кодирования чисел в виде терминов лямбда-исчисления. Число n представлено функцией, которая принимает функцию f и значение x и применяет функцию к значению n раз. (Например, три - это λf. λx. f (f (f (x))).) С помощью этого представления легко кодировать лямбда-термины, которые могут складываться или умножаться, но было неясно, как кодировать функцию-предшественницу , которая находит число на единицу меньше заданного числа. Однажды днем в кабинете дантиста Клини внезапно увидела, как определить predecessor [34]. Когда Клини представил результат своему руководителю, Черч признался, что он почти убедил себя в том, что представление предшественника в лямбда-исчислении невозможно. Как только это препятствие было преодолено, Черч и его ученики вскоре убедились, что любая “эффективно вычисляемая” функция чисел может быть представлена членом в лямбда-исчислении.

Черч предложил λ-определяемость в качестве определения “эффективно вычисляемого”, что мы теперь знаем как тезис Черча, и продемонстрировал, что существует проблема, решение которой не является λ-определяемым, проблема определения того, имеет ли данный λ-член нормальную форму, то, что мы теперь знаем как проблему остановки [9]. Год спустя, он продемонстрировал, что не существует λ-определяемого решения проблемы Энчея-данга [8].

В 1933 году Гедель прибыл с визитом в Принстон. Его не-убедило утверждение Черча о том, что каждая эффективно вычисляемая функция является λ-определяемой. Черч ответил, предложив, что если Гедель предложит другое определение, то Черч “возьмется доказать, что оно включено в λ-определимость”. В серии лекций в Принстоне в 1934 году, основанных на предложении Хербранда, Гедель предложил то, что стало известно как “общие рекурсивные функции” в качестве своего кандидата на эффективную вычислимость. Клини сделала заметки и опубликовала определение [35]. Черч и его ученики вскоре определили, что эти два определения эквивалентны: каждая общая повторяемая функция курсива является λ-определяемой, и наоборот. Доказательство было выдвинуто Черчем [8] и подробно опубликовано Клином [36]. Скорее этот результат не успокоил Геделя, а заставил его усомниться в правильности его собственного определения! Ситуация зашла в тупик.

Тем временем в Кембридже Алан Тьюринг, ученик Макса Ньюмена, независимо сформулировал свое собственное понятие “эффективно вычисляемого” в форме того, что мы сейчас называем машиной Тьюринга, и использовал это, чтобы показать неразрешимость задачи Entscheidungsproblem..... До того, как статья была опубликована, Ньюман был встревожен, обнаружив, что Тьюринг был похищен Черчем. Однако подход Тьюринга достаточно отличался от подхода Черча, чтобы заслуживать независимой публикации . Тьюринг поспешно добавил приложение, описывающее эквивалентность определимости λ для своих машин, и его статья [60] появилась в печати через год после статьи Черча, когда Тьюрингу было 23 года. Ньюман распорядился, чтобы Тьюринг поехал в Принстон, где он защитил докторскую степень под руководством Черча.

Самое существенное отличие Тьюринга от Черча заключалось не в логике или математике, а в философии. В то время как Черч просто представил определение λ-определимости и прямо заявил, что оно соответствует эффективной вычисляемости, Тьюринг предпринял анализ возможностей “компьютера” - в то время этот термин относился к человеку, выполняющему вычисления с помощью бумаги и карандаша. Тьюринг утверждал, что количество символов должно быть конечным (поскольку если бы они были бесконечными, некоторые символы были бы сколь угодно близки друг к другу и неразличимы), что количество состояний сознания должно быть конечным (по той же причине) и что количество символов, рассматриваемых в данный момент, должно быть ограничено (“Мы не можем с первого взгляда сказать, являются ли 999999999999999999 одинаковыми”). Позже Ганди [18] укажет, что аргумент Тьюринга сводится к теореме, утверждающей, что любое вычисление, которое может выполнить человек с бумагой и карандашом, также может быть выполнено машиной Тьюринга . Именно аргумент Тьюринга окончательно убедил Геделя; поскольку была доказана эквивалентность λ-определимости, рекурсивных функций и машин Тьюринга, теперь он признал, что все три определения “эффективно вычисляемы”.

Как уже упоминалось, первым использованием Черчем лямбда-исчисления было кодирование логических формул, но от этого пришлось отказаться, поскольку это приводило к непоследовательности. Сбой возник по причине, связанной с парадоксом Рассела, а именно с тем, что система позволяла предикату воздействовать на себя, и поэтому Черч адаптировал решение, аналогичное Расселовскому, - классификацию терминов по типам. Простотипизированное лямбда-исчисление Черча исключало самостоятельное применение, позволяя лямбда-исчислению поддерживать непротиворечивую логическую формулировку [10].

В то время как самоприменимость в логике Рассела приводит к парадоксу, самоприменимость в нетипизированном лямбда-исчислении Черча приводит к не завершающимся вычислениям. И наоборот, просто типизированное лямбда-исчисление Черча гарантирует, что каждый член имеет нормальную форму, то есть соответствует вычислению, которое останавливается.

Два приложения лямбда-исчисления - для представления вычислений и для представления логики - в некотором смысле взаимоисключают друг друга. Если понятие вычисления достаточно мощно, чтобы представить любую эффективно вычисляемую процедуру, то это понятие недостаточно мощно, чтобы решить свою собственную проблему остановки: не существует эффективно вычисляемой процедуры для определения, завершается ли данная эффективно вычисляемая процедура. Однако последовательность логики Черча, основанной на простом типизированном лямбда-исчислении, зависит от того, что каждый член имеет нормальную форму.

Нетипизированное лямбда-исчисление или типизированное лямбда-исчисление с структурой для общей рекурсии (иногда называемой оператором фиксированной точки) допускает определение любой эффективно вычислимой функции, но имеет проблему остановки, которая неразрешима. Типизированные лямбда—вычисления без конструкции для общей рекурсии имеют проблему остановки, которая тривиальна — каждая программа останавливается! - но не может определить некоторые эффективно вычислимые функции.

Оба вида математического анализа имеют свое применение, в зависимости от предполагаемого применения.

Помимо фундаментального вклада в языки программирования
, Черч также внес ранний вклад в верификацию оборудования
и проверку моделей, как описано Варди [62].

3. Гентцен и теория доказательства

Второй целью программы Гильберта было установить непротиворечивость различных логик. Если логика непоследовательна, то она может вывести любую формулу, что сделает ее бесполезной.

В 1935 году, в возрасте 25 лет, Герхард Генцен [20] представил не одну, а две новые формулировки логики: естественную дедукцию и последовательное количественное исчисление
которые утвердились как две основные системы формулирования логики и остаются таковыми по сей день. Он показал, как нормализовать доказательства, чтобы гарантировать, что они не были “обходными”, что дало новое доказательство непротиворечивости системы Гильберта. И, в довершение всего, чтобы соответствовать использованию символа ∃ для экзистенциальной квантификации, введенной Пеано, Гентцен ввел символ ∀ для обозначения универсальной квантификации. Он записал импликацию как A ⊃ B (если выполняется A, то выполняется B), конъюнкцию как A & B (выполняются как A, так и B), а дизъюнкцию как A ∨ B (выполняется хотя бы одно из A или B).

Идея Гентцена заключалась в том, что правила доказательства должны быть составлены парами, чего не было в более ранних системах, таких как система Гильберта. В естественном выводе это вводные и исключающие пары. Вводное правило вывода определяет, при каких обстоятельствах можно утверждать формулу с логической связкой (например, чтобы доказать A ⊃ B, можно предположить A, а затем необходимо доказать B), в то время как соответствующее правило исключения показывает, как использовать эту логическую связку (например, из доказательства A ⊃ B и доказательства A можно вывести B, свойство, получившее название modus ponens в средние века). Как отмечает Гентцен, “Введения представляют собой, так сказать, ”определения рассматриваемых символов", а исключения, в конечном счете, являются не более чем следствиями этих определений".

Следствием этого понимания стало то, что любое доказательство может быть нормализовано до такого, которое не является “обходным путем”, где “в доказательство не входят никакие концепции, отличные от тех, которые содержатся в конечном результате”. Например, в нормализованном доказательстве формулы A & B, единственными формулами, которые могут появиться, являются она сама и ее подформулы, A и B, а также сами подформулы A и B. Никакая другая формула, такая как (B & A) ⊃ (A & B) или A ∨ B, не может отображаться; это называется свойством подформулы. Непосредственным следствием была последовательность. Доказать ложь - это противоречие, написанное
. единственный способ вывести противоречие - доказать, скажем, как A ⊃
, так и A для некоторой формулы A. Но, учитывая такое доказательство, можно было бы нормализовать его к тому, которое содержит только подформулы его заключения,
. Но не имеет подформул! Это как старая поговорка: “Какую часть "нет" ты не понимаешь?" Логики заинтересовались нормализацией доказательств из-за ее роли в установлении согласованности.

Гентцен предпочитал систему естественной дедукции, потому что она была, по его мнению, более естественной. Он представил последовательное исчисление в основном как техническое устройство для доказательства свойства подформулы, хотя оно представляет самостоятельный интерес.

Последовательное исчисление обладает двумя ключевыми свойствами. Во-первых, каждое доказательство в Естественном выводе может быть преобразовано в доказательство в последовательном исчислении, и наоборот, так что эти две системы эквивалентны. Во-вторых, в отличие от естественного вывода, каждое правило, кроме одного, обладает тем свойством, что его гипотезы включают только подформулы из тех, которые фигурируют в его заключении. Единственное исключение, правило отсечения, всегда может быть удалено с помощью процесса, называемого устранением отсечения. Следовательно, каждое доказательство имело эквивалентную нормальную форму, удовлетворяющую свойству подформулы. Основным интересом Генцена к последовательному исчислению было доказательство подформулы Свойства, хотя у последовательного исчисления есть особенности, представляющие независимый интерес, такие как обеспечение более симметричного представления классической логики, и сегодня исследователи часто используют формулировки, более близкие к последовательному исчислению, чем к естественному выводу.

Ирония заключается в том, что Гентцену потребовалось ввести последовательное исчисление, чтобы доказать свойство подформулы для естественного Дедукция. Ему нужно было обходное доказательство, чтобы показать отсутствие обходных доказательств! Позже, в 1965 году, Правиц показал, как доказать свойство субформулы напрямую, представив способ упрощения доказательств методом естественной дедукции; и это заложило основу для работы Говарда, описанной в следующем разделе.

4. Предложения как типы

В 1934 году Карри наблюдал любопытный факт, связывающий теорию функций с теорией импликации [13].....
tions Каждый тип функции (A → B) может быть прочитан как предложение (A ⊃ B), и при таком прочтении тип любой данной функции всегда будет соответствовать доказуемому предложению. И наоборот, для каждого доказуемого предложения существовала функция с соответствующим типом. Впоследствии Карри и Фейс [14] расширили соответствие не только с типов и предложений, но и включили термин и доказательства, а также намекнули на связь между оценкой терминов и упрощением доказательств.

В 1969 году Говард распространил ксерокопированную рукопись [32]. Она не публиковалась до 1980 года, когда появилась в праздничном выпуске, посвященном Карри. Мотивированный наблюдением Карри, Ховард указал , что существует аналогичное соответствие между естественным выводом , с одной стороны, и просто типизированным лямбда-исчислением, с другой, и он четко обозначил третий и самый глубокий уровень соответствия, ответ, как описано во введении, что упрощение доказательств соответствует оценке программ. Говард показал, что соответствие распространяется и на другие логические связки, конъюнкцию и дизъюнкцию, расширив свой лямбда-исчисление с помощью структур, представляющих пары и непересекающиеся суммы. Точно так же, как правила проверки представлены парами введения и исключения, так и правила ввода: правила введения соответствуют способам определения или конструирования значения данного типа, а правила исключения соответствуют способам использования или деконструкции значений данного типа.

Мы можем описать наблюдение Говарда следующим образом:

- Соединение A и B соответствует декартову произведению A×B, то есть записи с двумя полями, также известной как пара. Доказательство предложения A & B состоит из доказательства A и доказательства B. Аналогично, значение типа A × B состоит из значения типа A и значения типа B.
- Дизъюнкция A ∨ B соответствует непересекающейся сумме A + B, то есть варианту с двумя альтернативами. Доказательство предложения A ∨ B состоит либо из доказательства A, либо из доказательства B, включая указание на то, какое из двух было доказано. Аналогично, значение типа A + B состоит либо из значения типа A, либо из значения типа B, включая указание на то, является ли это левым или правым слагаемым.
- Импликация A ⊃ B соответствует функциональному пространству A → B. Доказательство предложения A ⊃ B состоит из процедуры, которая при условии доказательства A дает доказательство B. Аналогично, значение типа A → B состоит из функции, которая при применении к значению типа A возвращает значение типа B.

Это прочтение доказательств восходит к интуиционистам и часто называется интерпретацией BHK, названной в честь Брауэра, Хейтинга и Колмогорова. Брауэр основал интуиционизм [28], а Хейтинг [29] , Колмогоров [39] формализовали интуиционистскую логику и разработали приведенную выше интерпретацию в 1920-х и 1930-х годах. Реализуемость, введенная Клином [37] в 1940-х годах, основана на аналогичной интерпретации.

Учитывая интуиционистское прочтение доказательств, вряд ли кажется удивительным, что интуиционистская естественная дедукция и лямбда-исчисление должны так близко соответствовать друг другу. Но только после Говарда переписка была изложена четко, таким образом, чтобы позволить работающим логикам и компьютерщикам использовать ее.

Бумага Говарда делится на две половины. В первой половине объясняется соответствие между двумя хорошо понятными концепциями, пропозициональными связками &, ∨, ⊃, с одной стороны, и вычислительными типами ×, +, → с другой стороны. Вторая половина расширяет эту аналогию и для хорошо понятных понятий из логики предлагает новые понятия для типов, которые им соответствуют. В частности, Ховард предполагает, что кванторы предикатов ∀ и ∃ соответствуют новым типам, которые мы теперь называем зависимыми типами.

С введением зависимых типов каждое доказательство в предикатной логике может быть представлено термином подходящего типизированного лямбда-исчисления. Математики и компьютерщики предложили числовые системы, основанные на этой концепции, в том числе автомат де Брейна [17], теорию типов Мартина-Лофа [43], PRL Бейтса и Констебля и NuPRL [3], а также конструктивный анализ Кокванда и Хьюэта [11], который превратился в помощника по проверке Coq.

Приложения включают CompCert, сертифицированный компилятор для языка программирования C, проверенный в Coq [41]; проверенное компьютером доказательство теоремы о четырех цветах, также проверенное в Coq [25]; части распределенной системы Ensemble, проверенные в NuPRL [27, 40]; и двадцать тысяч строк подключаемых модулей браузера, проверенных в F [57].

работа де Брюйна была независимой от работы Говарда, но Говард непосредственно вдохновил Мартина Лофа и все другие работы, перечисленные выше. Говард (по праву!) гордился своей статьей, называя ее одним из двух величайших достижений своей карьеры [55].

5. Интуиционистская логика

В "Гондольерах" Гилберта и Салливана Касильде рассказывают, что в младенчестве она была замужем за наследником короля Батавии, но что из-за путаницы никто не знает, кто из двух человек, Марко или Джузеппе, является наследником. Встревоженная, она причитает: “Значит, ты хочешь сказать, что я замужем за одним из двух гондольеров, но невозможно сказать, за каким именно?” На что последовал ответ: “Без каких-либо сомнений какого бы то ни было рода”.

Логика бывает многих разновидностей, и одно различие проводится между классической и интуиционистской. Интуиционисты, обеспокоенные бесцеремонными предположениями некоторых логиков о природе бесконечности, настаивают на конструкционистском понятии истины. В частности, они настаивают на том, что доказательство A ∨ B должно показывать, какое из A или B имеет место, и, следовательно, они отвергнут утверждение о том, что Касильда замужем за Марко или Джузеппе, пока один из них не будет идентифицирован как ее муж. Возможно, Гилберт и Салливан предвосхитили интуиционизм, поскольку итог их истории таков, что наследником оказывается третий человек, Луис, в которого Касильда, к счастью, уже влюблена.

Интуиционисты также отвергают закон исключенного среднего, который утверждает A ∨ A для каждого A, поскольку закон не дает подсказки относительно того, какой из них A или A выполняется. Хейтинг формализовал вариант классической логики Гильберта, который отражает интуиционистское понятие доказуемости. В частности, закон исключенного среднего доказуем в логике Гильберта, но не в логике Хейтинга. Далее, если закон исключенного среднего добавить в качестве аксиомы к логике Хейтинга, то он становится эквивалентным логике Гильберта. Колмогоров показал, что две логики тесно связаны: он дал перевод с двойным отрицанием, такой, что формула доказуема в классической логике тогда и только тогда, когда ее перевод доказуем в интуиционистской логике.

Предложения как типы были впервые сформулированы для интуиционистской логики. Это идеально подходит, потому что в интуиционистской интерпретации формула A ∨ B доказуема именно тогда, когда демонстрируется либо доказательство A, либо доказательство B, поэтому тип, соответствующий дизъюнкции, является непересекающейся суммой.

6. Другая логика, другие вычисления

Принцип высказываний как типов был бы замечательным, даже если бы он применялся только к одному варианту логики и одному варианту вычисления . Насколько же более примечательно, что это применимо к широкому разнообразию логики и вычислений.

Квантификация по пропозициональным переменным в логике второго порядка соответствует абстракции типов в лямбда-вычислениях второго порядка . По этой причине лямбда-исчисление второго порядка было открыто дважды, один раз логиком Жан-Ивом Жираром [21] и один раз специалистом по информатике Джоном Рейнольдсом [53]. И по той же причине подобная система, поддерживающая вывод по принципу типа, также была открыта дважды, один раз логиком Роджером Хиндли [30] и один раз компьютерщиком Робином Милнером [45]. Основываясь на соответствии, Джон Митчелл и Гордон Плоткин [46] заметили, что экзистенциальная квантификация в логике второго порядка точно соответствует абстракции данных, идее, которая сейчас лежит в основе многих повторных поисков в семантике языков программирования. Разработка универсальных типов в Java и C # опирается непосредственно на Джирарда-Рейнольдса, в то время как системы типов функциональных языков, включая ML и Haskell, основаны на Хиндли-Милнере. Философы могут спорить о том, "открыты" или "изобретены" математические системы, но одна и та же система, возникающая в двух разных контекстах, утверждает, что здесь правильным словом является "обнаружены".

Двумя основными вариантами логики являются интуиционистская и классическая. В оригинальной статье Говарда наблюдалось соответствие с интуиционистской логикой. Только два десятилетия спустя соответствие было распространено и на классическую логику, когда Тим Гриффин [26] заметил, что закон Пирса в классической логике предоставляет тип для оператора вызова / cc Схемы. Далее Чет Мерти [49] отметил, что перевод Колмогорова и Геделя с двойным отрицанием, широко используемый для связи интуиционистской и классической логики, соответствует преобразованию стиля продолжения, широко используемому как семантиками, так и разработчиками лямбда-исчисления. Париго [50], Кюриен и Херблин [12] и Вадлер [64] представили различные предположительные исчисления, мотивированные соответствиями классической логике.

Модальная логика позволяет маркировать пропозиции как “обязательно истинные’ или "возможно истинные". Кларенс Льюис представил модальную логику в 1910 году, и его учебник 1938 года [42] описывает пять вариантов, S1–S5. Некоторые утверждают, что каждый из этих вариантов имеет интерпретацию как форма вычисления с помощью предложений как типов, и авансовый платеж по этому утверждению дает интерпретация S4 как поэтапного вычисления благодаря Дэвису и Пфеннингу [16], а S5 как пространственно распределенного вычисления благодаря Мерфи и др. [48].

Эуджению Моджи [47] представил монады как метод для объяснения семантики важных функций языков программирования, таких как состояние, исключения и ввод-вывод. Монады стали широко распространены в функциональном языке Haskell, а позже проникли в другие языки, включая Clojure, Scala, F # и C #. Бентон, Бирман и де Пайва [4] заметили, что монады соответствуют еще одной модальной логике, отличной от всех S1-S5.

Временная логика допускает различие между такими модальностями, как "выполняется сейчас", "будет выполняться в конечном итоге" и "будет выполняться на следующем временном шаге’. Темпоральная логика была впервые формализована Артуром Прайором в его тексте 1957 года [52] и стала играть важную роль в спецификации и верификации вычислительных систем, начиная с работы Амира Пнуэли [51]. Интерпретации временной логики через пропозиции как типов включают применение к частичной оценке благодаря Дэвису [15] и применение к функциональному реактивному программированию благодаря Джеффри [33].

В классической, интуиционистской и модальной логике любая гипотеза может использоваться произвольное количество раз — ноль, один раз или много. Линейная логика, представленная в 1987 году Жираром [22], требует, чтобы каждая гипотеза использовалась ровно один раз. Линейная логика "осознает ресурсы" в том смысле, что факты могут быть использованы и заменены другими фактами, подходящими для рассуждений о мире, где ситуации меняются. С самого начала предполагалось, что линейная логика применима к задачам, важным для специалистов по информатике, и ее первая публикация была опубликована не в "Анналах математики" , а в журнале "Теоретическая информатика". Ком-

предполагаемые аспекты линейной логики обсуждаются Абрамским [1] и Вадлером [63], среди многих других, а приложения к квантовым вычислениям рассматриваются Геем [19]. Совсем недавно типы сеансов, способ описания протоколов связи, представленный Honda [31], были связаны с интуиционистской линейной логикой Caires и Pfenning [5], а также с классической линейной логикой Wadler [65].

Одним из ключей к пониманию соответствия между логикой и вычислениями является изучение теории категорий. Как простотипизированное лямбда-исчисление, так и интуиционистский естественный вывод соответствуют понятию декартовой замкнутой категории [54]. Возникает множество расширений этой идеи, включая захватывающую работу, связывающую категории, вычисления , линейную логику и квантовую физику [2].

Владимир Воеводский, лауреат медали Филдса, вызвал большой интерес своей недавней работой по теории гомотопических типов (HoTT) и одновалентным основаниям, которая связывает топологию с предложениями как типами. Особый год, посвященный этому предмету и организованный Институтом перспективных исследований в Принстоне, родине Черча, привел к публикации в прошлом году книги Хотта, которую действительно горячо ждали, и авторами которой были более 50 математиков и компьютерщиков, от Акцеля до Цайленберга.

Предложения как типы остаются темой активных исследований.

7. Естественная дедукция

Теперь мы переходим к более формальному развитию, представляя фрагмент естественного вывода и фрагмент типизированного лямбда-исчисления в стиле, который проясняет связь между ними.

Мы начнем с деталей естественного вывода, определенных Генценом [20]. Правила доказательства показаны на рисунке 1. Чтобы упростить наше обсуждение, мы рассмотрим только два связных термина естественного вывода. Мы пишем A и B как заполнители, обозначающие произвольные формулы. Конъюнкция пишется A & B , а импликация - A ⊃ B.

Мы представляем доказательства в виде деревьев, где каждый узел дерева является экземпляром правила доказательства. Каждое правило доказательства состоит из нуля или более формул, написанных над строкой, называемой предпосылками, и единственной формулы, написанной под строкой, называемой заключением. Интерпретация правила заключается в том, что когда верны все предпосылки, то следует вывод.

Правила доказательства представлены парами, с правилами введения и устранения каждой связки, обозначенными буквами -I и -E соответственно. Когда мы читаем правила сверху донизу, вводные и исключающие правила делают то, что написано на tin: первое вводит формулу для связки, которая появляется в заключении, но не в посылках; второе устраняет формулу для связки, которая появляется в посылке, но не в заключении. Вводное правило описывает, при каких условиях, как мы говорим, сохраняется связка — как определить связку. Правило исключения описывает, к какому выводу мы можем прийти, когда связь сохраняется — как использовать связь.

Вводное правило для соединения &-I гласит, что если формула A выполняется и формула B выполняется, то формула A & B также должна выполняться . Для конъюнкции существует два правила исключения. Первое, &-E $_1$, утверждает, что если формула A & B выполняется, то формула A также должна выполняться. Второй, &-E $_2$, заключает B, а не A.

Правило введения для импликации, ⊃-I, гласит, что если из предположения о том, что формула A справедлива, мы можем вывести формулу B, тогда мы можем заключить, что формула A ⊃ B справедлива, и выполнить предположение. Чтобы указать, что A используется как нулевое допущение, один или много раз в доказательстве B, мы заключаем A в квадратные скобки и привязываем его к B с помощью многоточий. Доказательство является полным только тогда, когда каждое допущение в нем было подтверждено соответствующим использованием ⊃-I, что обозначается написанием одного и того же имени (здесь x) в качестве надстрочного знака в каждом экземпляре высказанного предположения и в правиле высказывания. Правило исключения для импликации, ⊃-E, гласит, что если выполняется формула A ⊃ B и если выполняется формула A, то

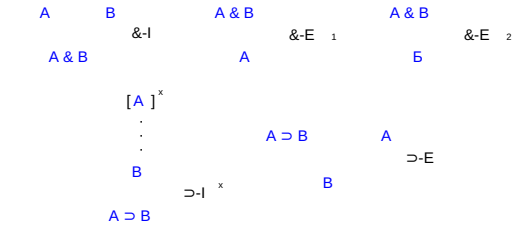

Рисунок 1. Герхард Генцен (1935) — естественная дедукция

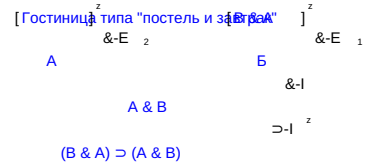

Рисунок 2. Доказательство

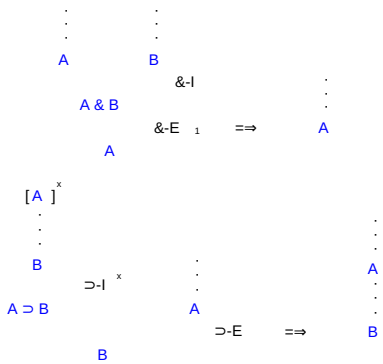

Рисунок 3. Упрощающие доказательства

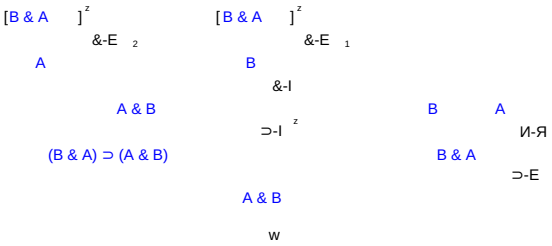

Рисунок 4. Упрощение доказательства

мы можем заключить, что формула B также справедлива; как упоминалось ранее, это правило также называется modus ponens.

Критически настроенные читатели заметят, что мы используем аналогичный язык для описания правил написания ('когда-то") и формул ("подразумевает"). Та же идея распространяется на двух уровнях-мета-уровень (нормы) и уровень объекта (для-Мулас), и в двух нотациях, с использованием линии с территории выше и ниже вывод результатов на мета-уровне, и символ ⊃ с помещения налево и вывод справа в Обь-екту уровне. Это почти как если бы для того, чтобы понять подтекст, нужно сначала понять подтекст! Этот логический парадокс Зенона был иронично замечен Льюисом Кэрроллом [6], а сам феномен был глубоко исследован Мартином Лофом [44]. Пусть это нас не беспокоит; каждый обладает хорошим неформальным пониманием импликации, которое может послужить основой для ее формального описания.

Доказательство формулы

(B & A) ⊃ (A & B) .

показано на рисунке 2. Другими словами, если B и A выполняются, то A и B выполняются. Это может показаться настолько очевидным, что вряд ли заслуживает доказательства! Однако формулы B ⊃A и A⊃B имеют разные значения, и нам нужен какой-то формальный способ заключить, что формулы B & A и A & B имеют одинаковые значения. Это то, что показывает наше доказательство, и обнадеживает тот факт, что оно может быть построено на основе правил, которые мы устанавливаем.

Доказательство гласит следующее. Из B & A мы заключаем A через &-E $_2$, а из B & A мы также заключаем B через &-E $_1$. Из A и B мы заключаем A & B через &-I. То есть, исходя из предположения B & A (используется дважды), мы заключаем A & B. Мы выполняем предположение и завершаем (B & A) ⊃ (A & B) с помощью ⊃-I, связывая выполненные предположения с правилом выполнения, записывая z как верхний индекс на каждом.

Некоторые доказательства излишне окольны. Правила для упрощения приведения доказательств приведены на рисунке 3, а пример такого доказательства приведен на рисунке 4. Давайте сначала сосредоточимся на примере.

В верхней части рисунка 4 показано более крупное доказательство, построенное на основе доказательства на рисунке 2. Более крупное доказательство предполагает в качестве предпосылок две формулы, B и A, и завершается формулой A & B. Однако вместо того, чтобы заключать это напрямую, мы выводим результат окольным путем, чтобы проиллюстрировать пример ⊃-E, modus ponens. Доказательство выглядит следующим образом. Слева приведено доказательство, приведенное ранее, заключающееся в (B & A) ⊃ (A & B). Справа, из B и A мы заключаем B & A через &-I. Объединение этих результатов дает A & B через ⊃-E.

Мы можем упростить доказательство, применив правила перезаписи, приведенные на рисунке 3. Эти правила определяют, как упростить доказательство, когда за правилом введения немедленно следует соответствующее правило исключения. Каждое правило показывает два доказательства, соединенные стрелкой, указывающей на то, что повторный анализ (доказательство слева) может быть переписан или упрощен, чтобы получить reduct (доказательство справа). Перезапись всегда приводит к замене действительного доказательства другим действительным доказательством.

Для & повторный анализ состоит из доказательства A и доказательства B, которые в совокупности дают A & B через &-I, что, в свою очередь, дает A через &-E $_1$. Редукция состоит просто из доказательства A, отбрасывающего ненужное доказательство B. Существует аналогичное правило, не показанное, для упрощения появления &-I, за которым следует &-E $_2$.

Для ⊃ повторный анализ состоит из доказательства B из предположения A, которое дает A ⊃ B по ⊃-I, и доказательства A, которые в совокупности дают B по ⊃-E. Сокращение состоит из того же доказательства B, но теперь при каждом появлении предположения A заменяется приведенным доказательством A. Предположение A может использоваться ноль, один или много раз в доказательстве B в redex, поэтому доказательство A может быть скопировано ноль, один или много раз в доказательстве B в reduct. По этой причине reduct может быть больше, чем redex , но это будет быть проще в том смысле, что он устранил ненужный обход через подпроверку A ⊃ B.

Мы можем рассматривать предположение о A в ⊃-I как долг, который погашается доказательством A, представленным в ⊃-E. Доказательство в redex накапливает долг и выплачивает его позже; в то время как доказательство в reduct оплачивает напрямую каждый раз, когда используется предположение. Подтверждающий долг отличается от денежного долга тем, что проценты отсутствуют, и одно и то же доказательство может свободно дублироваться столько раз, сколько необходимо для погашения предположения, того самого свойства, которого деньги, поскольку их трудно подделать, призваны избегать!

На рисунке 4 показано использование этих правил для упрощения доказательства. Первое доказательство содержит экземпляр ⊃-I, за которым следует ⊃-E, и упрощается путем замены каждого из двух предположений B & A на слева копией доказательства B & A справа. Результатом является второе доказательство, которое в результате замены теперь содержит экземпляр &-I, за которым следует &-E $_2$, и еще один пример &-I за которым следует &-E. Упрощение каждого из них дает третье доказательство, которое выводит A & B непосредственно из предположений A и B и не может быть упрощено дальше.

Нетрудно видеть, что доказательства в нормальной форме удовлетворяют свойству под-формулы: каждая формула такого доказательства должна быть под-формулой одного из его невыраженных предположений или его заключения . Доказательство на рис. 2 и окончательное доказательство на рис. 4 удовлетворяют этому свойству, в то время как первое доказательство на рис. 4 - нет, поскольку (B & A) ⊃ (A & B) не является подформулой A & B.

8. Лямбда-исчисление

Теперь мы обратим наше внимание на простотипизированное лямбда-исчисление Черча [10]. Правила ввода показаны на рисунке 5. Чтобы упростить наше обсуждение, мы берем и продукты, и функции как примитивные типы; исходное исчисление Черча содержало только типы функций, с продуктами в качестве производной конструкции. Теперь мы запишем A и B как заполнители для произвольных типов, а L, M, N как заполнители для произвольных терминов. Типы продуктов записываются A × B, а типы функций записываются A → B. Теперь вместо формул наши посылки и выводы являются суждениями о форме

M : A

указывает, что член M относится к типу A.

Подобно доказательствам, мы представляем производные типов деревьями, где каждый узел дерева является экземпляром правила типа. Правило каждого типа состоит из нуля или более суждений, написанных над строкой, называемой предпосылками, и одного суждения, написанного под строкой, называемой заключением. Интерпретация правила заключается в том, что когда верны все предпосылки, то следует вывод.

Как и правила доказательства, правила ввода бывают парами. Вводное правило описывает, как определить или сконструировать термин данного типа, в то время как исключающее правило описывает, как использовать или деконструировать термин данного типа.

Правило введения для продуктов ×-I гласит, что если член M имеет тип A, а член N имеет тип B, то мы можем сформировать пару член M, N типа продукта A × B. Для продуктов существует два правила исключения . Первый, ×-E $_1$, гласит, что если член L имеет тип A × B, тогда мы можем сформировать член π $_1$ L типа A, который выбирает первый компонент пары. Второй, ×-E $_2$ аналогично, за исключением того, что оно образует член π $_2$ L типа B.

Правило введения для функций →-I гласит, что если задана переменная x типа A, мы сформировали член N типа B, то мы можем сформировать лямбда-член λx. N функции типа A → B. Переменная x выглядит свободной в N и связанной в λx. N . Нерасчлененные как предположения соответствуют свободным переменным, в то время как разряженные предположения соответствуют связанным переменным. Чтобы указать, что переменная x может быть равна нулю, один или много раз в члене N , мы пишем x : A в скобках и привязываем его к N : B через многоточие. Член является замкнутым только тогда, когда каждая переменная в нем связана соответствующим членом λ. Правило исключения для функций, →-E, гласит, что задано

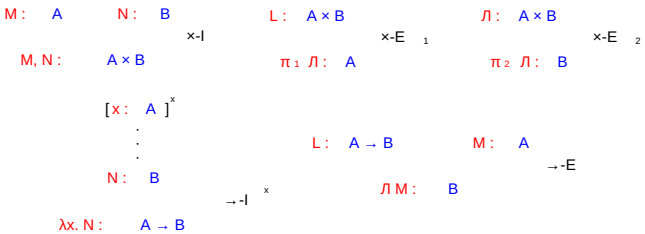

Рисунок 5. Алонзо Черч (1935) — Лямбда-исчисление

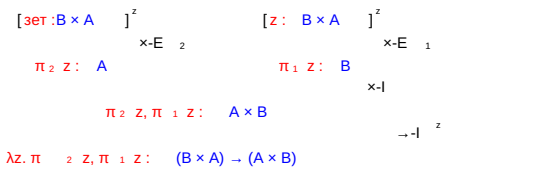

Рисунок 6. Программа

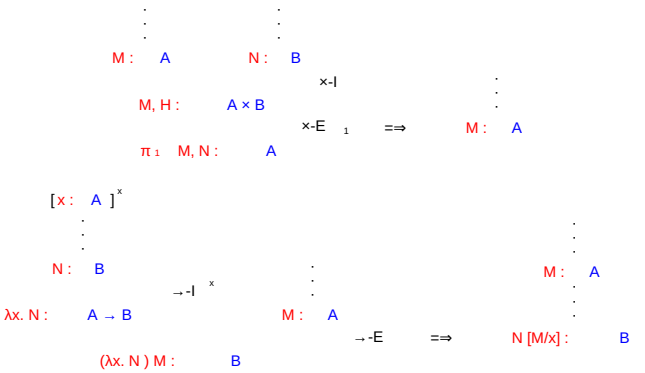

Рисунок 7. Оценка программ

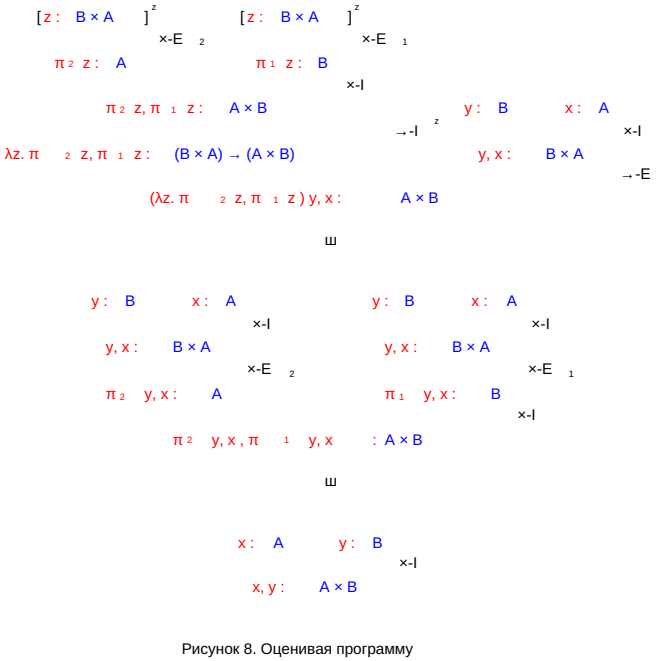

Рисунок 8. Оценивая программу

термин L типа A → B и термин M типа A мы можем сформировать прикладной термин L M типа B.

Для естественного вывода мы отметили, что может возникнуть путаница между импликацией на метауровне и объектном уровне. Для лямбда-исчисления различие более четкое, поскольку у нас есть импликация на метауровне (если термины над строкой хорошо типизированы, то и термины ниже тоже), но функции на объектном уровне (функция имеет тип A → B потому что если ей передается значение типа A, то она возвращает значение типа B). То, что раньше было выводом из предположений (возможно, слегка размытое понятие), становится привязкой переменных (концепция, понятная большинству специалистов по информатике).

Читатель, должно быть, уже заметил поразительное сходство между правилами Генцена из предыдущего раздела и правилами Черча из этого раздела: игнорируя термины в правилах Черча, тогда они идентичны, если заменить & на × и ⊃ на →. Цвет правил выбран таким образом, чтобы подчеркнуть сходство.

Программа типа

(B×A) → (A×B)

показано на рисунке 6. В то время как разница между B & A и A & B кажется простой формальностью, разницу между B × A и A × B легче оценить: преобразование последнего в первое требует замены элементов пары, что как раз и является задачей, выполняемой программой, соответствующей нашему предыдущему доказательству.

Программа выглядит следующим образом. Из переменной z типа B × A формируем член π 2 обозначение типа A через ×-E, а также член π 1 z типа B через ×-E 1 . Из этих двух мы формируем пару π 2 z, π 1 z типа A × B через ×-I. Наконец, мы связываем свободную переменную z с образованием лямбда-члена λz. π 2 z, π 1 z типа (B × A) → (A × B) через →-I, соединяя связанные типы с правилом привязки, записывая z надстрочным знаком на каждом. Функция принимает пару и меняет местами ее элементы в точности так, как описано в ее типе.

Программа может быть оценена путем переписывания. Правила оценки программ показаны на рисунке 7, а пример - на рисунке 8. Давайте сначала сосредоточимся на примере.

В верхней части рисунка 8 показана увеличенная программа, созданная на основе программы, изображенной на рисунке 6. Более крупная программа имеет две свободные переменные, у типа B и x типа A, и создает значение типа A × B. Однако, вместо того, чтобы создавать его напрямую, мы достигаем результата окольным путем, чтобы проиллюстрировать пример применения функции →-E. Программа выглядит следующим образом. На слева приведена приведенная ранее программа, формирующая функцию типа (B × A) → (A × B). Справа, из B и A мы формируем пару y, x типа B × A через ×-I. Применение функции к паре образует член типа A × B через →-E.

Мы можем оценить эту программу, применив правила перезаписи , приведенные на рисунке 7. Эти правила определяют, как переписать термин, когда за вводным правилом немедленно следует соответствующее исключающее правило. Каждое правило показывает два вывода, соединенных стрелкой, указывающей на то, что redex (термин слева) может быть переписан или оценен для получения reduct (термин справа). Перезаписи всегда преобразуют допустимую деривацию типа в другую допустимую деривацию типа , гарантируя, что при перезаписи сохраняются типы - свойство, известное как сокращение объема или правильность типа.

Для × повторный анализ состоит из члена M типа A и члена N типа B, которые в совокупности дают член M, N типа A × B через ×-I, что, в свою очередь, дает член π 1 M, N типа A через ×-E 1 . Сокращение состоит просто из члена M типа A, отбрасывая ненужный член N типа B. Существует аналогичное правило, не показанное, для перезаписи вхождения ×-I, за которым следует ×-E 2 .

Для → повторное выражение состоит из вывода члена N типа B из переменной x типа A, что дает лямбда-член λx. N из типа A → B через →-I и производное от члена M типа A, которые в совокупности дают приложение (λx. N) M типа B через →-E. Сокращение состоит из члена N [M/ x], который заменяет каждое свободное число

вхождение переменной x в член N через член M . Далее, если в
выводе, что N имеет тип B , мы заменим каждое предположение о том, что x
имеет тип A, выводом, что M имеет тип A, мы получим вывод,
показывающий, что N [M / x] имеет тип B. Поскольку переменная x может появляться
ноль, один или много раз в члене N , член M может быть скопирован
ноль, один или много раз в приведенном N [M / x]. По этой причине,
reduct может быть больше, чем redex , но это будет проще в
том смысле, что is удалил подтерм типа A → B. Таким образом,
выполнение предположений соответствует применению функции к ее
аргументу.

На рисунке 8 показано использование этих правил для оценки программы.
Первая программа содержит экземпляр →-I, за которым следует →-E,
и переписывается путем замены каждого из двух вхождений z
типа B × A слева копией термина y, x типа B × A
справа. Результатом является вторая программа, которая в результате
замены теперь содержит экземпляр ×-I, за которым следует ×-E [2],
и другой пример ×-I, за которым следует ×-E [1]. Перезапись каждого из
этих параметров приводит к созданию третьей программы, которая выводит термы x, y типа
A × B и больше не может быть оценена.

Следовательно, упрощение доказательств в точности соответствует оценке
программ, в данном случае демонстрируя, что применение
функции к паре действительно меняет местами ее элементы.

9. Заключение

Предложения как типы отражают наш взгляд на универсальность определенных
языков программирования.

На космическом корабле "Пионер" установлена табличка, предназначенная для связи с инопланетянами, если кто-нибудь когда-нибудь ее перехватит (см рис 9). Им
может показаться, что некоторые части этого легче интерпретировать, чем другие. Радиальная диаграмма
показывает расстояние от Солнца до четырнадцати пульсаров и центра
галактики. Инопланетяне, вероятно, определят, что длина
каждой линии пропорциональна расстояниям до каждого тела. На другой диаграмме
люди изображены перед силуэтом "Пионера". Если “Звездный путь”
даст точное представление об инопланетных видах, они могут ответить
"Они выглядят точно так же, как мы, за исключением того, что у них отсутствуют волосы на лобке". Однако, если
система восприятия инопланетян сильно отличается от нашей собственной, они
могут быть не в состоянии расшифровать эти закорючки.

Что произошло бы, если бы мы попытались связаться с инопланетянами,
передав компьютерную программу? В фильме "День независимости"
герои уничтожают вторгшийся материнский корабль пришельцев, заражая его
компьютерным вирусом. Внимательный просмотр переданной программы
показывает, что она содержит фигурные скобки — она написана на диалекте C!
Маловероятно, что инопланетные виды будут программировать на C, и неясно, что
инопланетяне могли бы расшифровать программу, написанную на C, если бы она была представлена.

Как насчет лямбда-исчисления? Предложения как типы говорят нам, что
лямбда-исчисление изоморфно естественному выводу. Кажется трудным
представить себе инопланетных существ, не знающих основ
логики, и мы могли бы ожидать, что проблема расшифровки программы,
написанной в лямбда-исчислении, будет ближе к проблеме
понимания радиальной диаграммы пульсаров, чем проблема понимания
изображения мужчины и женщины на мемориальной доске пионеров.

У нас может возникнуть соблазн заключить, что лямбда-исчисление универсально,
но сначала давайте поразмыслим над уместностью слова "универсальный".
В наши дни широко распространена многомировая интерпретация квантовой физики
. Ученые предполагают, что в разных вселенных
можно столкнуться с разными фундаментальными константами, такими как
сила тяжести или постоянная Планка. Но как бы легко ни было
представить себе вселенную, где гравитация отличается, трудно представить себе
вселенную, где фундаментальные правила логики неприменимы. Естественная
дедукция, а следовательно, и лямбда-исчисление, должны быть известны
инопланетянам не только в нашей вселенной, но и в других. Таким образом,
мы можем заключить, что было бы ошибкой характеризовать лямбда-математику
как универсальный язык, потому что называть его универсальным было бы
слишком ограничивающим

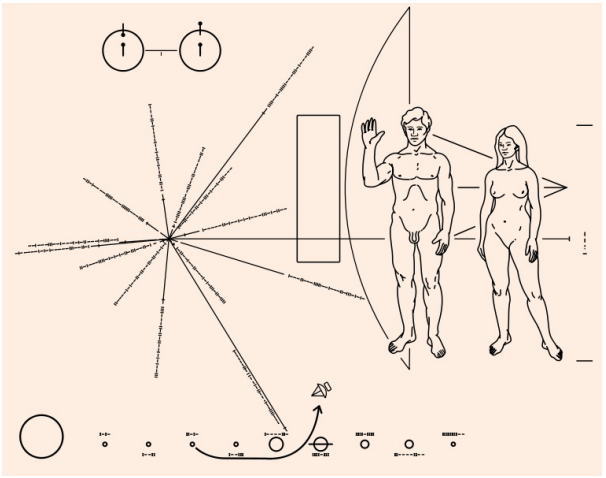

Рисунок 9. Мемориальная доска на космическом корабле "Пионер"

Благодарности. Спасибо Гершому Базерману, Питу
Бевину, Гаю Блеллоху, Ринциусу Блоку, Эзре Куперу, Бену Дарвину,
Бенджамину Денкле, Питеру Дыбьеру, Йоханнесу Эмериху, Мартину Эр-
Вигу, Ицу Гейлу, Михаилу Глушенкову, Габору Грейфу, Виноду Гроверу,
Сильвену Генри, Филипу Хольценшпису, Уильяму Ховарду, Джону Хьюзу,
Колину Лаптоун, Дэниэлу Марсдену, Крейгу Макфлинну, Тому Мертелу,
Саймону Пейтон-Джонс, Бенджамин Пирс, Ли Пайк, Андрес Сикард-
Рамирес, Скотт Рострап, Данн Толивер, Мошe Варди, Джереми Ял-
лоп, Ричард Зак, Лео Зовик и судьи. Эта работа была
профинансирована EPSRC EP/K034413/1.

Филип Уодлер (wadler@inf.ed.ac.uk, @PhilipWadler) является профес-
сор теоретической информатики в лаборатории Шая
основах информатики на факультете информатики в
Университете Эдинбурга, Шотландия.

A. Говард с карри-Howard

Во время написания этой статьи я понял, что мне непонятны некоторые моменты истории.
Ниже приводится письмо, которое я написал Уильяму Ховарду, и его ответ
(с исправлениями, которые он внес после того, как я попросил опубликовать его). Я считаю, что это
полезный исторический документ, и я благодарен Говарду за его
разрешение на публикацию. Соответствие относится к Shell-Gellasch
[55], а ссылки на рисунки 5 и 6 ниже относятся к
рисункам в этой статье.

Вот мой первоначальный запрос.

Тема: Понятие построения формул как типов

Уважаемый профессор Ховард,

на мои исследования большое влияние оказали ваши собственные, особенно
особенно статья, цитируемая в моей теме. Сейчас я пишу статью о
области работы, выросшей из этой статьи, которая была запрошена
для публикации отделом коммуникаций ACM (флагман
профессиональной организации компьютерщиков). Черновик
документа прилагается.

Я хотел бы точно изобразить историю предмета. Я
прочитал ваше интервью с Shell-Gallasch, но
остается несколько вопросов, на которые, я надеюсь, вы будете достаточно любезны ответить.

Ваша бумага разламывается на две половинки. Первая описывает соответ-
соответствие между пропозициональной логики и простых типов, ТРЦ-
ОНД соответствие между предикатов логика и де-
кулон типов. Вы считали первую половину новым материалом или
просто повторением того, что было известно? В какой степени вы кон-
Сидер своей работе опирается на или предвидели работы Heyt-
ING и Колмогоров, и Клини-х реализуемость? В какой степени

ваша работа повлияла на последующее творчество де Брюйна и Мартина Лофа? Какова была история вашего мимеографа на эту тему и почему он не был опубликован до фестиваля карри в 1980 году?

Большое спасибо за ваше внимание, не говоря уже о том, что вы нашли-в моей области! С уважением, —P

И вот его ответ:

Уважаемый проф. Вадлер,

Как упоминалось в интервью Shell-Gellasch, моя работа над предложения как типы (p-a-t) возникли из моей переписки с Крейзелем, который был очень заинтересован в получении математического понятия (т. Е. в обычной математике) для идеи Брауэра о конструкции (как объяснил Хейтинг). Я не был знаком с работой Брауэра или Хейтинга, не говоря уже о Колмогорове, но, из того, что сказал Крайзель, идея была достаточно ясна: конструкция из $\alpha \to \beta$ должна была быть конструкцией F, которая, действуя на конструкцию A из α, дает конструкцию B из β. Итак, у нас есть конструкции, действующие на конструкции, скорее как функционалы, действующие на функционалы. Итак, в качестве приближения,

(1) давайте считать, что “конструкция” означает “функциональный”.

Но что это за функционалы? В конструктивной математике функционал не задается как набор упорядоченных пар. Скорее,

(2) указать функционал - значит указать не только действие или процесс, который он выполняет, но и указать его тип (домен и контрдомен).

Очевидно, что структура типа будет сложной. Я поставил перед собой задачу найти подходящее обозначение для символов типа. Итак, нужен подходящий символ типа для функционала F, приведенного выше. Ну, просто примите это за саму alpha (на данный момент я думал о логике высказываний). Внезапно я вспомнил кое-что, о чем Карри говорил на семинаре по логике во время моего пребывания в Пенсильванском университете. Если мы рассмотрим типизированные комбинаторы и посмотрим на структуру символов типа основных комбинаторов (например, S, K, I), мы увидим, что каждый из символов типа соответствует (изоморфен) одной из аксиом чистой импликативной логики. Что ж! Это было как раз то, что мне было нужно!

Как мы можем сформулировать следующее понятие?

(3) F - это конструкция phi.

Рассмотрим случай, в котором φ имеет вид $\alpha \supset \beta$ Соблазн состоит в том, чтобы определить, что “F - конструкция из $\alpha \supset \beta$ означает “для всех A: если A - конструкция из α, то Fa - конструкция из β”. Ну, это замкнуто, потому что мы использовали “если · · ·, то · · ·” для определения импликации. Это то, что вы называете “логическим парадоксом Зеноса”. Я избежал этой округлости, приняв (3) за значение:

(4) F присваивается тип φ в соответствии со способом построения F; т.е. Способом построения F.

Таким образом, F является построением φ по построению. Ваш рисунок 6 точно иллюстрирует, что я имел в виду под этим. (В то время у меня не было этого красивого обозначения, но оно передает то, что я имел в виду.)

Подводя итог: мое базовое понимание состояло одновременно из мыслей (2) и (4) плюс мысли о том, что наблюдение Карри предоставило средства для реализации (2), (4). Позвольте мне сказать это по-другому. Мысль (2) была не нова. У меня была эта мысль (2) в течение многих лет, с тех пор как я начал изучать примитивные рекурсивные функционалы конечного типа. Новой была сама мысль (4) плюс признание того, что идея Карри обеспечила способ реализации (4). Я получил это базовое понимание летом 1966 года. Как только я увидел, как это делается с помощью комбинаторов, мне стало интересно, как это будет выглядеть с точки зрения лямбда-исчисления, и увидел, к моему восторгу, что это соответствует интуиционистской версии последовательного исчисления Генцена.

Кстати, наблюдение Карри относительно типов базовых комбинаторов представлено в его книге с Фейсами (Curry-Feys), но я не знал об этом, хотя у меня был экземпляр в течение нескольких лет (с 1959 года, когда я был нанят в Пенсильванском государственном университете). Проработав детали p-a-t в течение нескольких месяцев, я начал подумывать о том, чтобы написать об этом, поэтому я подумал, что мне лучше посмотреть, есть ли это в книге. Что ж, его достаточно легко найти, если вы знаете, что вы ищете. Взглянув на это, я испытал шок: они не только распространили идеи на последовательное исчисление Генцена, но и выявили связь между устранением сокращений при выводе и нормализацией соответствующего лямбда-члена. Но, присмотревшись повнимательнее, я пришел к выводу, что между ними была связь, но не сама связь. Оказывается, в этом я тоже был не совсем прав. Смотрите мое замечание об их теореме 5 ниже. Не то чтобы это имело большое значение для всего, что я мог бы опубликовать: даже если в них была связь между последовательным исчислением Генцена и лямбда-исчислением, у меня было далеко идущее обобщение (т. Е. К арифметике Хейтинга).

Вышесказанное более подробное, чем требовалось бы для ответа на ваши вопросы, но мне нужно было написать это, чтобы прояснить свои мысли по этому поводу; поэтому я могу также включить вышесказанное, поскольку думаю, что это заинтересует вас. Это ответ на один из ваших вопросов: “В какой степени вы считаете, что ваша работа опирается на работу Хейтинга и Колмогорова или была предвосхищена ею, а также возможностью реализации Клини?” А именно, моя работа опирается на работу Хейтинга и Брауэра, через объяснение Крайзелем этой работы мне. Ничто из этого не было предвосхищено работой Хейтинга, Колмогорова или Клина: они не думали о функционалах конечного типа. Хотя я был знаком с рекурсивной реализуемостью Клини, в то время я не думал об этом. По общему признанию, это затрагивает идеи о конструкциях Брауэра, но далеко не отражает понятие конструкции (на самом деле, Клини однажды высказал замечания на этот счет, я забыл где). Из-за связи между конструкциями и рекурсивной реализуемостью Клини, могло быть какое-то бессознательное влияние; но, в любом случае, незначительное влияние.

Повлияла ли ваша работа на последующую работу де Брюйна и Мартина Лофа? Насколько я знаю, моя работа не оказала никакого влияния на работу де Брюйна. Его работа, похоже, полностью независима от моей. Я помню, что однажды он прислал мне пакет материалов Automath . Проект компьютерной программы для проверки существующих доказательств показался мне не очень интересным, и я не ответил. Что меня заинтересовало бы, так это программа для поиска доказательств результатов, которые еще не были доказаны! Даже помощник по проверке подошел бы. Почему он прислал мне материалы Automath? Я не помню, в каком году это было. Где-то в 1970-х. Каким бы ни было сопроводительное письмо, оно не было информативным; просто что-то вроде: "Уважаемый профессор Ховард, вас может заинтересовать следующий материал ...". С тех пор я прочитал две или три статьи него, и у меня сложилось более благоприятное впечатление. Это хорошая, основательная работа. Очевидно, оригинальная. Он самостоятельно открыл идею производных как терминов, и сопутствующую идею формул как типов. Он использует лямбда-термины, но, я думаю, только для целей описания. Другими словами, я не думаю, что у него есть связь между нормализацией и устранением сокращений, но я не проводил тщательного изучения его работы. На самом деле, использует ли он вообще систему Генцена? Я просто не знаю. На последние два вопроса ответил бы любой, кто знаком с его работой. В любом случае, отдайте ему должное. Вкусностей хватит на всех.

Мое мнение о Мартина-Лофа? Никаких проблем. Я встретил его на конференции в Буффало в 1968 году и поделился с ним своими идеями. Его мгновенной реакцией было: “Так вот, почему я об этом не подумал?” У него была выездная встреча в UIC на 1968-1969 учебный год, так что у нас было много возможностей поговорить, и он начал разрабатывать свой собственный подход к этим идеям. В январе 1969 года, главным образом для того, чтобы убедиться, что нам обоим было ясно, кто что обнаружил, я изложил свои собственные идеи в виде рукописных заметок. К тому времени ксероксы были широко распространены, поэтому я отправил копию Крейселю, и он раздал копии разным людям, включая Жирара. По крайней мере, я думаю, что именно так Жирар получил копию, или, возможно, Мартин-Лоф дал ему ее. Мне нравятся работы Мартина-Лофа. Я мог бы сказать об этом больше, но краткий ответ на

ваш вопрос таков: работа Мартина-Лофа основана на моей. Он всегда отдавал мне должное, и мы хорошие друзья.

Продолжая размышлять, я должен упомянуть, что в том первом разговоре Мартин-Лоф предположил, что идея производных как терминов будет особенно хорошо работать в связи с теорией естественной дедукции Правица. Я подумал: "Ладно, но ничего страшного. На самом деле, в то время я не был знаком с результатами Правица (или, если вообще был, то лишь смутно). Но это было намного опасней, чем я думал, потому что Prawitz по сокращению шагов для вычета соответствует напря повторно производственных шагов для связанного лямбда срок! На самом деле, для большинства целей мне нравится последовательная формулировка естественного вывода, как приведено на страницах 33 и 88 книги Соренсена и Уржичина (2006). Фактически, если мы добавим к этому введение с левой импликацией (давайте ограничимся-чистой импликативной логикой), результирующая система P # будет довольно интересной. Все проявления modus ponens могут быть устранены, а не только те, которым предшествует левое подразумевание-введение. Это то, чем я занимаюсь в своей статье JSL 1980 года “Порядковый анализ терминов конечного типа”. Кроме того, правило сокращения легко вывести в P # (просто учтите, для типизированных лямбда-терминов: правильно сформированный термин, замененный на правильно сформированный термин, приводит к правильно сформированному термину); следовательно, P # - это консервативное расширение системы P * в части I моей небольшой статьи в Curry Festschrift.

Фраза "формулы как типы" была придумана Крайзелем для того, чтобы у нас было название предмета в нашей переписке туда и обратно. Я бы предположил, что фраза ”предложения как типы” была придумана Мартином-Лофом; по крайней мере, во время нашей первой дискуссии на встрече в Буффало в 1968 году он предположил, что можно думать о типе как о предложении, согласно идее, что в интуитивной ционистской математике значение предложения φ задается видом “всех” доказательств φ. Я использую здесь кавычки, потому что мы не говорим о теоретико-множественной завершенной бесконечности.

“Вторая [часть] вводит соответствие между логикой предикатов и зависимыми типами”. Я вообще не думал об этом в таком ключе. Я хотел дать интерпретацию понятия построения для какой-нибудь нетривиальной части интуиционистской математики (арифметики Хейтинга). Часть I статьи была лишь предварительной для этого. На самом деле, то, что вы говорите в формате PDF, согласуется с этим. Здесь нет необходимости в переменах.

“Вы считали первую половину новым материалом или просто повторением того, что было известно?” Новый. Но в январе прошлого года у меня была возможность по-настоящему внимательно изучить материал из книги Карри-Фейс, стр. 313-314; и теперь я вижу, что существует гораздо более тесная связь между моей теоремой 2 в части I и их теоремой 5, стр. 326, чем я думал. Проблемы здесь довольно интересные. Я могу провести обсуждение, если вы хотите.

Во введении к моей небольшой статье я упоминаю, что Тейт оказал на меня влияние. Позвольте мне сказать несколько слов об этом. Летом 1963 года у нас были беседы, в которых он объяснил мне, что он разработал теорию бесконечных членов по аналогии с теорией бесконечных доказательств Штте, где нормализация (посредством лямбда-редукций) бесконечных членов соответствует исключению соответствующего доказательства. Он не знал, что с этим делать. Он думал о своей теории бесконечных членов как о своего рода игре слов на тему теории бесконечных доказательств Штте. Но мы оба согласились, что должна быть глубокая связь между нормализацией лямбда-терминов и устранением сокращений Генцена. Мы ломали голову над этим в течение двух или трех наших бесед, но не смогли прийти к ответу.

Как объяснялось в первом абзаце этого электронного письма, моя работа возникла из-за проблемы, поставленной Крайзелем; итак, в начале этой работы я, конечно, не думал об этих разговорах с Тейтом.

Но, как упоминалось выше, как только я получил базовое представление о значимости комбинаторов Карри, я подумал, как это будет работать для лямбда-терминов. В этот момент я вспомнил свои разговоры с Тейтом. Другими словами, когда я подтвердил, что

(5) исключение сокращений для производной соответствует нормализации

что касается термина,

разговоры с Тейтом были очень заняты моими мыслями. Скорее всего, я бы заметил (5), не побеседовав с Тейтом. Но кто знает? В любом случае, он заслуживает похвалы за то, что заметил соответствие между производными и терминами.

Чего у него не было, так это соответствующего соответствия между предложениями и типами. На самом деле, он не использовал для этого достаточно общее понятие типа. Оглядываясь назад, мы можем видеть, что в его системе существует гомоморфизм, а не изоморфизм, от предложений к типам.

Мне нужно сказать немного больше о стиле и типах. Поскольку Шутте расширил свою систему доказательств до трансфинитных порядков, Тейт расширил свою систему терминов до уровней трансфинитного типа. У меня уже была своя система примитивно-рекурсивных функционалов трансфинитного типа. В нашем самом первом разговоре мы сравнили идеи по этой теме. Эта тема требует, чтобы человек очень серьезно задумался о понятии типа. Конечно, я уже много думал о понятии типа (из-за (2) выше) до того, как я познакомился с Тейтом, но мои беседы с ним усилили эту тенденцию. Мыслей о типах было очень много у меня в голове, когда я начал рассматривать пункты (1), (2) выше.

Как уже упоминалось, заметки были написаны от руки и ксерокопированы; никаких мимеографов. “Почему [они] не были опубликованы до "Фестиваля карри" в 1980 году?" Сначала позвольте мне упомянуть, почему они были опубликованы в "Фестивале карри". Селден готовил праздничный стол к 80-летию Карри. Он попросил меня внести заметки. Я сказал: “Конечно. Я напишу улучшенную версию. Теперь я могу сделать намного лучше”. Он ответил: “Нет, мне нужны оригинальные заметки. Это исторический документ”. Другими словами, к тому времени были распространены различные копии, и в литературе имелся ряд ссылок на них. Поэтому я напечатал их на машинке и отправил.

Почему я не опубликовал их раньше? Просто потому, что они не решили исходную проблему. Таков был вердикт Крайзеля и Геделя (Крайзель показал или описал работу Геделю). На самом деле, еще до того, как я сообщил о проделанной работе Крайселю, я знал, что получил лишь приблизительное представление о конструкции и что предстоит проделать еще больше работы. По сути, критика заключается в следующем. В моей небольшой статье я не привожу аксиомы и правила вывода для доказательства утверждений вида

(3) F - это конструкция из φ.

Помните, мы должны избегать "логического парадокса Зеноса"! Ответ заключается в том, что доказательства будут выглядеть так, как показано на рисунке 6. Другими словами, рисунок 6 - это не только программа; это еще и доказательство (или: оно может быть переосмыслено как доказательство). Но рисунок 6 также можно интерпретировать как объяснение того, как должна быть построена конструкция (синяя), чтобы иметь заданный тип (красная). Другими словами, рисунки, такие как рисунок 6, реализуют идею (4), упомянутую в начале этого электронного письма; т. е. F присваивается тип φ в соответствии со способом построения F.

Надеюсь, вам это нравится; мне это определенно нравится. Конечно, правила вывода такие же, как на рисунке 5. Таким образом, эти простые идеи обеспечивают недостающую теорию конструкций; или, по крайней мере, обеспечивают значительный шаг в этом направлении.

В январе 2013 года я обменялся несколькими электронными письмами с Тейтом и Кон-стейблом по истории p-a-t. Это заставило меня по-настоящему внимательно взглянуть на книгу "Карри-Фейс". Вот кое-что, что я обнаружил, что действительно рассмешило меня: требуемая теория, выводы из которой имеют форму, приведенную на рисунке 5, уже есть в Карри-Фейсе. По общему признанию, чтобы увидеть это, сначала нужно стереть все стили поворота ( ); Карри, кажется, одержим ими. В частности, удалите турникеты из дерева доказательств на странице 281. Результатом будет в точности дерево доказательств общей формы, приведенное на вашем рисунке 6. (Подсказка: $(\cdots)X$ следует читать как “X имеет тип $(\cdots)$”. Другими словами, перепишите $(\cdots)X$ как $X : (\cdots)$.) Что означает F значение bc, где F выделено жирным шрифтом? Просто перепишите F bc как $b \to c$. Вы видите? Я эксперт. Я мог бы, вероятно,

заработать деньги написанием ручного перевода. Таким образом, требуемая теория - это, по сути, просто теория функциональности Карри (точнее, подходящий вариант теории Карри). Итак, пропустил ли я здесь лодку? Мог бы я увидеть все это в 1969 году, если бы только у меня хватило решимости внимательно присмотреться к "Карри-Фейс"? Я не знаю. Для этого может потребоваться ясность ума, представленная обозначением на рис. 5. Есть ли у вас какие-нибудь идеи, когда и где это обозначение вошло в обиход?

Еще одно замечание относительно причины моего отказа от публикации. Разве я не чувствовал, что совершил важный прорыв, несмотря на критику Крайзеля и Геделя? С одной стороны, да. С другой стороны, у меня были сомнения. За исключением Мартина-Лофа, Правица, Тейта и Жирара, очень немногие люди проявили интерес к этим идеям. Но, возможно, Мартина-Лофа, Правица, Тейта и Жирара должно было быть достаточно. Вы говорите: “Конечно, Говард гордился выявленной им связью, называя это одним из двух величайших достижений своей карьеры” [43]. Должны ли мы оставить этот отрывок в силе? Конечно. Интервью состоялось весной 2000 года. К тому времени я получал много похвал от сообщества компьютерных наук. Итак, гордость - это своеобразная вещь. Позвольте мне закончить это на позитивной ноте. В 1969 году Правиц был в США и приехал в МСЖД с докладом. Войдя в комнату, он направился прямиком ко мне, посмотрел мне в глаза и пожал руку. Сообщение гласило: "Молодец!" ЭТО заставляло меня гордиться.

Есть еще что сказать; но, я думаю, это отвечает на ваши вопросы; поэтому я отправлю его, чтобы избежать дальнейших задержек.

Ваш PDF-файл "Предложения как типы" очень удобочитаем.

Билл

В более позднем сообщении была предоставлена дополнительная информация об отношении Карри и Фейс [14].

Карри заметил поразительный факт, что

(1) если базовые комбинаторы типизированы, то типы, которые они изменяют они имеют ту же структуру, что и различные аксиомы чистой импликативной логики, P.

Как простое следствие этого, получается соответствие между теоремами P и типами всех комбинаторов, которые состоят из основных комбинаторов. Чтобы избежать околичностей, давайте сформулируем это в терминах системы простых типизированных комбинаторов: существует соответствие между теоремами P и типами типизированных комбинаторов. Только что упомянутое соответствие лучше выразить, заметив, что существует

(2) соответствие между производными в P и типами типизированных комбинаторов.

В подходе Карри комбинатор не задается с типом; скорее, комбинатор получает тип с помощью “базовой теории функциональности”, Func. Следовательно, он дает эквивалент

(3) соответствие между теоремами P и теоремами функции (плюс аксиомы, приведенные в (1)).

Это приведено в "Карри-Фейс", стр. 313-314. Затем разрабатывается вариант этого, который дает соответствие между выводами в стиле Гентцена и “базовой теорией функциональности”, адаптированной к лямбда-терминам (стр. 315-332).

Рассмотрим интуиционистское последовательное исчисление Гентцена, ограниченное импликацией. Таким образом, правилами, характеризующими LJ, являются: modus ponens, левая импликация-введение и сокращение. Теорема об исключении сокращения для этой системы гласит:

(4) Из вывода последовательности в LJ мы можем получить вывод той же последовательности в системе LJ*, где LJ * - это LJ без правила отсечения.

В подходе Карри-Фейса к терминам и их типам нетрудно представить утверждение, эквивалентное (4), поэтому немного удивительно, что они этого не делают — по крайней мере, не в той форме, которую можно было бы ожидать. Ближе всего они подходят к этому в формулировке теоремы 5, страница 326. Более того, теорема 5 имеет пятистраничное доказательство, которому нелегко следовать, тогда как с точки зрения типизированных лямбда-выражений условия (4) довольно очевидны. А именно, если данный вывод в (4) соответствует члену A, то нормальная форма A обеспечивает требуемый вывод без вырезов. Другими словами, результат (4) легко следует из нормализации A.

Итак, перед нами небольшая загадка. Мне кажется, что доказательство теоремы 5 в основном посвящено доказательству того, что

(5) типизированный лямбда-термин может быть нормализован.

Если я прав насчет этого, то объяснение тайны состояло бы в том, что (5) не было широко известно в то время, когда был написан "Карри-Фейс" (дата публикации: 1958).

Позже Говард развил свой последний пункт выше.

Что касается вопроса о том, было ли (5) широко известно в то время, когда был написан "Карри-Фейс", ответ, к моему удивлению, таков: по-видимому, нет. Я только что вспомнил, что Робин Ганди, которого я довольно хорошо знал, опубликовал в журнале Curry Festschrift статью о доказательстве Тьюринга (5). (На самом деле, он объяснил мне доказательство в 1978 году.) Ганди говорит на стр. 454:

“Самое раннее опубликованное доказательство [из (5)], известное мне, находится в " Карри" и книга Фея "Комбинаторная логика " . . . "

Ганди говорит нам, что (5) сформулировано как следствие теоремы 9, страница 340. Теорема 9 - чудовище. Может быть, кто-нибудь когда-нибудь объяснит мне, о чем это говорит. К счастью, соответствующее следствие, которое находится на стр. 341, ясно говорит (5). В моей борьбе с Карри-Фейсами я так и не добрался до стр. 341.

Спасибо, Робин. Хорошее шоу!

Доказательство Тьюринга - это то, о чем подумал бы почти каждый (сначала выполните переопределения в стратегическом порядке: “самый правый”–“самый внутренний”–лямбда-операторы самого высокого типа). Напротив, доказательство Карри-Фея в доказательстве теоремы 5 следует стилю метода наглядности Тейта (“Интенсиональные интерпретации ... ”) или его варианта . По крайней мере, таково мое впечатление. Кто-то должен это проверить.

Список литературы

[1] С. Абрамский. Вычислительные интерпретации линейной логики. Теоретическая информатика , 111(1&2):3-57, 1993.

[2] Дж. Баэз и М. Стэйн. Физика, топология, логика и вычисления: розеттский камень. В Б. Кеке, редакторе, Новые структуры для физики, Конспекты лекций по физике, страницы 91-166. Шпрингер-Ферлаг, 2009.

[3] Дж. Л. Бейтс и Р. Л. Констебль. Доказательства как программы. Транзакции на Языки и системы программирования , 7(1):113-136, Январь 1985 г.

[4] П. Н. Бентон, Г. М. Бирман и В. де Пайва. Типы вычислений с логической точки зрения. Журнал функционального программирования, 8(2):177-193, 1998.

[5] Л. Кэйрс и Ф. Пфеннинг. Типы сеансов как интуиционистские линейные предложения. В “СОГЛАСИИ”, страницы 222-236, 2010.

[6] Л. Кэрролл. Что Черепаха сказала Ахиллесу. Разум, 4 (14): 278-280, Апрель 1895 года.

[7] А. Черч. Набор постулатов для основания логики. Анналы Математики , 33(2):346-366, 1932.

[8] А. Черч. Замечание о проблеме entscheidungsproblem. Журнал символизма Логика , 1:40-41, 1936. Получено 15 апреля 1936 года. Исправление, там же, 1:101-102 (1936), получено 13 августа 1936 г.

[9] А. Черч. Неразрешимая проблема теории элементарных чисел. Американский математический журнал , 58(2):345-363, Апрель 1936 г. Представлено Американскому математическому обществу, 19 апреля 1935 г.; аннотация в Бюллетене Американского математического общества, 41, май 1935 г.

[10] А. Черч. Формулировка простой теории типов. Журнал Символической логики, 5(2):56-68, Июнь 1940 г.

[11] Т. Коканд и Г. П. Хьюэт. Математические построения. Информация и вычисления , 76(2/3):95-120, 1988.

[12] П.-Л. Кюриен и Х. Гербелин. Двойственность вычислений. В Международная конференция по функциональному программированию (ICFP) , страницы 233–243, 2000.

[13] Х. Б. Карри. Функциональность в комбинаторной логике. Труды Национальной академии наук , 20:584-590, 1934.

[14] Х. Б. Карри и Р. Фейс. Комбинаторная логика. Северная Голландия, 1958.

[15] Р. Дэвис. Темпорально-логический подход к анализу времени привязки. В Логика в информатике (LICS) , страницы 184-195, 1996.

[16] Р. Дэвис и Ф. Пфеннинг. Модальный анализ поэтапных вычислений. В "Принципах языков программирования" (POPL), страницы 258-270, 1996.

[17] N. G. de Bruijn. Математический язык Automath, его использование и некоторые из его расширений. В "Симпозиуме по автоматической демонстрации", том 125 "Конспектов лекций по информатике", страницы 29-61. Springer-Verlag, 1968.

[18] Р. Ганди. Слияние идей в 1936 году. У Р. Херкена, редактора, Универсальная машина Тьюринга: обзор за полвека , страницы 51-102. Спрингер, 1995.

[19] С. Гей. Квантовые языки программирования: обзор и библиография. Математические структуры в информатике , 16(4):581-600, 2006.

[20] Г. Генцен. Untersuchungen ¨uber das logische Schließen. Mathematische Zeitschrift , 39(2–3):176–210, 405–431, 1935. Перепечатано в [58].

[21] Ж.-Ю. Жирар. Interpr´etation functionelle et ´elimination des coupures dans l'arithm´etique d'ordre sup´erieure, 1972. Парижский университет VII, Эти документы.

[22] Ж.-Ю. Жирар. Линейная логика. Теоретическая информатика, 50: 1-102, 1987.

[23] Ж.-Я. Жирар, П. Тейлор и Ю. Лафон. Доказательство и типы. Кембридж Издательство университета, 1989.

[24] К. Гедель. Über formal unterscheidbare S¨atze der Principia Mathematica und verwandter Systeme I. Monatshefte f¨ur Mathematik und Physik , 38:173–198, 1931. Перепечатано в [61].

[25] Г. Гонтье. Формальное доказательство – теорема о четырех цветах. Уведомления о AMS , 55(11):1382-1393, 2008.

[26] Т. Гриффин. Понятие контроля в виде формул. В "Принципах" Языки программирования (POPL) , страницы 47-58. ACM, январь 1990 года.

[27] М. Хейден и Р. ван Ренессе. Оптимизация многоуровневой связи Протоколы. В материалах 6-го Международного симпозиума по высокопроизводительным распределенным вычислениям, HPDC , страницы 169-177. Компьютерное общество IEEE, 1997.

[28] Д. Э. Хесселинг. Гномы в тумане: восприятие Брауэра интуиционизм 1920-х . Биркхаузер, 2003.

[29] А. Хейтинг. Mathematische Grundlagenforschung Intuitionismus Bewiestheorie . Ergebnisse der Mathematik und ihren Grenagebiete. Springer Verlag, Berlin, 1934.

[30] Р. Хиндли. Схема основного типа объекта в комбинаторной логике. Труды Американского математического общества, 146:29–60, декабрь 1969г.

[31] К. Хонда. Типы для диадического взаимодействия. В CONCUR, страницы 509-523, 1993.

[32] У. А. Ховард. Понятие построения формул как типов. В Х. Б. Карри: Очерки комбинаторной логики, лямбда-исчисления и формализма , страницы 479-491. Академическая пресса, 1980. Оригинальная Версия была распространена частным образом в 1969 году.

[33] А. Джеффри. Причинность бесплатно!: параметричность подразумевает причинность для функциональных реактивных программ. В языках программирования соответствует верификации программ (PLPV) , страницы 57-68, 2013.

[34] С. Клини. Истоки теории рекурсивных функций. Анналы истории вычислительной техники, 3(1):52-67, 1981.

[35] С. К. Клини. Общие рекурсивные функции натуральных чисел. Математический Аннален , 112(1), декабрь 1936. Аннотация опубликована в Бюллетене AMS, июль 1935 г.

[36] С. К. Клини. λ-определимость и рекурсивность. Математический анализ Дюка Журнал , 2:340-353, 1936.

[37] С. К. Клини. Об интерпретации интуиционистской теории чисел. Журнал символической логики , 10:109-124, 1945.

[38] С. К. Клини и Дж. Б. Россер. Противоречивость некоторых формальных логик. Анналы математики, 36: 630-636, 1936.

[39] А. Н. Колмогоров. Zur deutung der intuitionistischen logik. Mathematische Zeitschrift , 35:58–65, 1932.

[40] С. Kreitz. Построение надежных высокопроизводительных сетей с помощью системы разработки nuprl proof. Функциональный журнал Программирование , 14(1):21-68, 2004.

[41] Х. Лерой. Формальная проверка реалистичного компилятора. Commun. ACM, 52(7):107-115, 2009.

[42] К. Льюис и К. Лэнгфорд. Символическая логика. 1938. перепечатано Довером, 1959.

[43] П. Мартин-Лоф. Интуиционистская теория типов. Библиополис Неаполь, Италия, 1984.

[44] П. Мартин-Лоф. О значении логических констант и обосновании логических законов (Сиенские лекции, 1983). Северный журнал философской логики , 1(1):11-60, 1996.

[45] Р. Мильнер. Теория полиморфизма типов в программировании. J. Вычисления. Сист. Наука. , 17(3):348–375, 1978.

[46] Дж. К. Митчелл и Г. Д. Плоткин. Абстрактные типы имеют экзистенциальный тип. Труды по языкам и системам программирования , 10(3):470–502, июль 1988 года.

[47] Е. Moggi. Понятия вычислений и монад. Информация и Вычисления , 93(1):55-92, 1991.

[48] Т. Мерфи VII, К. Крэри, Р. Харпер и Ф. Пфеннинг. Симметричное Модальное лямбда-исчисление для распределенных вычислений. В Логике в информатике (LICS) , страницы 286-295, 2004.

[49] К. Мерти. Оценочная семантика для классических доказательств. В логике в Компьютерные науки (LICS) , страницы 96-107, 1991.

[50] М. Париго. λμ-математический анализ: алгоритмическая интерпретация классического естественного вывода. В "Логическом программировании и автоматизированных рассуждениях", том 624 Конспектов лекций по информатике, страницы 190-201. Springer-Verlag, 1992.

[51] А. Пнуэли. Временная логика программ. В FOCS, страницы 46-57, 1977.

[52] А. Прайор. Время и модальность. 1957.

[53] Дж. К. Рейнольдс. К теории структуры типов. На симпозиуме по программированию , том 19 "Конспектов лекций по информатике", страницы 408-423, 1974.

[54] Д. Скотт. Относящиеся теории λ-исчисления. В К Х. Б. Карри: Очерки комбинаторной логики, лямбда-исчисления и формализма , страницы 375-402. Академическая пресса, 1980.

[55] А. Э. Шелл-Геллаш. Размышления моего советника: Истории о математике и математичках. Математический интеллект, 25(1):35-41, 2003.

[56] М. Х. Серенсен и П. Уржичин. Лекции о Карри-Ховарде Изоморфизм . Эльзевир, 2006.

[57] Н. Свами, Дж. Чен, К. Фурне, П. Струб, К. Бхаргаван и Дж. Янг. Безопасное распределенное программирование с типами, зависящими от значений. В М. М. Т. Чакраварти, З. Ху и О. Данви, редакторы, Международная конференция по функциональному программированию (ICFP) , страницы 266-278. ACM, 2011.

[58] М. Э. Сабо, редактор. Собрание сочинений Герхарда Генцена. Север Голландия, 1969 год.

[59] С. Томпсон. Теория типов и функциональное программирование. Аддисон-Уэсли, 1991.

[60] А. М. Тьюринг. О вычислимых числах с приложением к

Entscheidungsproblem . Материалы Лондонского математического Общество, s2-42(1), 1937. Получено 28 мая 1936 г., прочитано 12 ноября 1936.

[61] Дж. ван Хейеноорт. От Фреге до Геделя: справочник по математике математическая логика, 1879-1931. Издательство Гарвардского университета, 1967.

[62] М. Я. Варди. Из церкви и до ПСЛ. В книге О. Грумберга и Х. Вейта, редакторов, 25 лет проверки моделей—история, достижения, перспективы , том 5000 конспектов лекций в Компьютерные науки , страницы 150-171. Спрингер, 2008.

[63] П. Уодлер. Вкус линейной логики. В математических основах Информатика (MFCS) , том 711 LNCS, страницы 185-210. Шпрингер-Ферлаг, 1993.

[64] П. Вадлер. Обращение по значению является двойным по отношению к обращению по имени. На Международной Конференции по функциональному программированию (ICFP) , страницы 189-201. ACM, 2003.

[65] П. Вадлер. Предложения как сессии. На Международной конференции по Функциональное программирование (ICFP) , страницы 273-286. ACM, 2012.

[66] А. Н. Уайтхед и Б. Рассел. Математические принципы. Кембридж Издательство университета, 1912.