Теоретико-категорная семантика модальной теории типов, основанной на интуиционистской эпистемической логике

Содержание

1 Предварительные замечания и определения

1.1 Глоссарий по основным конструкциям функционального языка программирования Haskell: функторы, монады, аппликативные функторы

Определение 1. *Класс типов*

Классом типов в языке Haskell – это реализация некоторого общего интерфейса для некоторой совокупности типов.

Представителем класса типов называется реализация данного класса для конкретного типа.

Определение 2. *Функтор*

Функтор – это однопараметрический класс типов, позволяющий пронести действие одноместной функции через значения, полученные в результате применения к их типу одноместного типового оператора.

Определение в стандартной библиотеке выглядит следующим образом:

```
class Functor f where
  fmap :: (a -> b) -> f a -> f b
```

Рассмотрим примеры:

- Список (неограниченная в длине последовательность) является функтором:

  ```
  instance Functor [] where
    fmap :: (a -> b) -> [a] -> [b]
    fmap f [] = []
    fmap f (x:xs) = (f x) : (fmap f xs)
  ```

  Данный пример достаточно прост: реализация функтора для списка – это функция высшего порядка, которая, принимая на входе одноместную функцию из типа a в тип b и список элементов типа a, возвращает список элементов типа b, который получен применением функции к каждому элементу списка, полученного на вход.

  Пример использования:

  ```
  > fmap succ ''bg`lo`fmd\USrtodqmnu`''
  ''champagne supernova''
  ```

  В данном примере мы отобразили функцию “следующий за” [1] на строчку, указанную выше. Строка в языка Haskell – это частный случай списка, списка символов. Тогда `fmap` применяет функцию `succ` к каждой букве строки, возвращая в результате строку со словосочетанием “champagne supernova”.

- Пара (тип декартова произведения типов) также функтор:

  ```
  instance Functor (b,) where
    fmap :: (a -> c) -> (b,a) -> (b,c)
    fmap f (x,y) = (x, f y)
  ```

[1] В языке Haskell функция “следующий за” определения для любого типа, между элементами которого есть линейный порядок, в частности, лексиграфический порядок на символах, вводимых с клавиатуры компьютера

Конструктор пары является двухпараметрическим типовым оператором, но мы сделали из него однопараметрический оператор фиксацией первого параметра.

Данная реализация также довольно проста: на вход принимается функция из типа a в тип c и кортеж, в котором первая координата имеет тип b, а вторая – тип a. На выходе мы получаем кортеж типа (b, c), применяя полученную на вход функцию ко второй координате пары.

- Тип $Maybe$ – это однопараметрический типовой оператор, для обработки неопределенных значений:

```
data Maybe a = Nothing | Just a
```

  Реализация функтора для типа $Maybe$:

```
instance Functor Maybe where
  fmap :: (a -> b) -> Maybe a -> Maybe b
  fmap f Nothing = Nothing
  fmap f (Just x) = Just (f x)
```

  Если второй аргумент является неопределенным значением (на вход передан $Nothing$), то и возвращается $Nothing$. Если же значение определено, то есть оно имеет вид $Just\, x$, тогда мы применяем функцию функцию к x, а результат вычисления оборачиваем в конструктор $Just$.

Определение 3. *Аппликативный функтор*

Аппликативным функтором называется класс типов, обобщающий функтор для функций произвольной арности.

Определение класса в языке Haskell:

```
class Functor f => Applicative f where
  pure :: a -> f a
  (<*>) :: f (a -> b) -> f a -> f b
```

Обобщение действия функтора с помощью методой класса `Applicative` для произвольной функции:

```
liftA2 :: (a -> b -> c) -> f a -> f b -> f c
liftA2 g x y = pure g <*> x <*> y
```

Данный комбинатор берет функцию, которая по объектам из типов a и b сопоставляет объект типа c и, по аргументам x и y типов соответственно $f\, a$ и $f\, b$, полученных в результате применения к данным типов функтора f, возвращает объект типа $f\, c$, тип которого также получен в результате применения функтора к типу $f\, c$.

Рассмотрим реализацию данной функции детальнее. Имея функцию g типа `a -> b -> c`, мы применяет к ней `pure`, получая в результате объект `pure g` типа `f (a -> b -> c)`, иными словами, мы подняли функцию g на уровень функтора f.

Далее мы применяем поднятую функцию f к аргументу x типа $f\, a$ с использованием `(<*>)` и получаем объект `pure g <*> x` типа `f (b -> c)`, получив одноместную функцию из `b` в `c`. Затем мы `pure g <*> x` применяем к аргументу y типа $f\, b$ опять же с использованием `(<*>)` и получаем объект типа $f\, c$.

Рассмотрим примеры представителей класса типов `Applicative`:

- Списки

```haskell
instance Applicative [] where
  pure x = [x]
  fs <*> xs = [ f x | f <- fs , x <- xs]
```

Метод `pure` для списков переводит объект x произвольного типа a в одноэлементный список $[x]$. Метод `(<*>)` в качестве левого операдна принимает список функций и список аргументов – в качестве правого и возващает список всевозможных применений элементов первого списка к элементам второго списка. Нотация здесь является калькой с теоретико-множественной нотации, которая могла бы иметь следующий вид $\{f(x) \mid f \in B^A \wedge x \in A\}$.

- Пары

```haskell
instance Monoid a => Applicative ((,) a) where
    pure x = (mempty, x)
    (u, f) <*> (v, x) = (u <> v, f x)
```

Чтобы пару сделать аппликативным функтором, необходимо соблюсти следующее ограничение: тип первой координаты должен быть моноидом [2].

Метод `pure` переводит объект x произвольного типа b в упорядоченную пару, первый элемент которой единица моноида a, а второй – x, полученный при входе. Метод `(<*>)` принимает две пары: первая пара – это пара элемента u моноида и некоторой функции f, вторая пара состоит из другого элемента v моноида и аргумента x. Возвращаемый результат: соединение элементво u и v с помощью моноидной операции в первой координате, а во второй координате – применение функции f к аргументу x.

Пример использования:

```haskell
> (''(what's the story) '' , succ) <*> (''morning glory?'' , 1994)
(''(what's the story) morning glory?'',1995)
```

где `succ` – функция “следующий за”.

- Тип Maybe

```haskell
instance Applicative Maybe where
  pure = Just
  Just f  <*> m = fmap f m
  Nothing <*> _m = Nothing
```

Пример использования:

```haskell
> liftA2 (++) (Just "Definitely") Nothing
Nothing

> liftA2 (++) (Just "Definitely ") (Just "Maybe")
Just ''Definitely Maybe''
```

[2]То есть тип должен содержать нейтральный и бинарную ассоциативную операцию. Моноид в языке Haskell также является классом типов, который, как видно из названия, является калькой с одноименной структуры в алгебре.

Данный пример как раз является примером того, как действие функтора обобщается на функций многих аргументов, в данном случае, двух аргументов. Здесь функцией двух аргументов является функция (++), конкатенация строк.

В первом случае определен только первый аргумент, строка `Definitely`, обернутая в `Just`, второй же аргумент является неопределенным (так как в качестве второго аргумента передан `Nothing`), поэтому данная попытка конкатенации строк также вернет `Nothing`.

Во втором примере, второй аргумент определен, это строка `Maybe`, обернутая в `Just`. Тогда конкатенация пройдет успешно и `liftA2` пронесет комбинатор (++) и совершит конкатенацию строк `Definitely` и `Maybe` внутри `Just`, и на выходе будет возвращена строка `Definitely Maybe`, к которой применен `Just`.

1.2 Приложение А. Глоссарий по теории категорий.

Определение 4. *Категория $\mathcal{C}$ состоит из:*

- *Класса объектов $Ob_{\mathcal{C}}$;*
- *Для любых объекта $A, B \in Ob_{\mathcal{C}}$ определено множество стрелок (или морфизмов) из A в B $Hom_{\mathcal{C}}(A, B)$;*
- *Если $f \in Hom_{\mathcal{C}}(A, B)$ и $g \in Hom_{\mathcal{C}}(B, C)$, то $g \circ f \in Hom_{\mathcal{C}}(A, C)$;*
- *Для любого объекта $A \in Ob_{\mathcal{C}}$, определен тождественный морфизм $id_A \in Hom_{\mathcal{C}}(A, A)$;*
- *Для любой стрелки $f \in Hom_{\mathcal{C}}(A, B)$, для любой стрелки $g \in Hom_{\mathcal{C}}(B, C)$ и для любой стрелки $h \in Hom_{\mathcal{C}}(C, D)$, $h \circ (g \circ f) = (h \circ g) \circ f$.*
- *Для любой стрелки $f \in Hom_{\mathcal{C}}(A, B)$, $f \circ id_A = f$ и $id_B \circ f = f$.*

Определение 5. *Функтор*

Пусть $\mathcal{C}, \mathcal{D}$ – категории. Функтором называется отображение $F : \mathcal{C} \to \mathcal{D}$, такое, что:

- *$F : A \mapsto FA$, где $A \in Ob_{\mathcal{C}}$;*
- *$F(g \circ f) = F(g) \circ F(f)$;*
- *$F(id_A) = id_{FA}$.*

Определение 6. *Естественное преобразование Пусть $\mathcal{F}, \mathcal{G} : \mathcal{C} \to \mathcal{D}$ – функторы. Естественным преобразованием $\alpha : \mathcal{F} \Rightarrow \mathcal{G}$ называется такое индексированное семейство стрелок $(\alpha_X)_{X \in Ob_{\mathcal{C}}}$, что для любых $A, B \in Ob_{\mathcal{C}}$, для любой стрелки $f \in Hom_{\mathcal{C}}(A, B)$, диаграмма коммутирует:*

$$\begin{tikzcd}
\mathcal{F}A \arrow[r, "\mathcal{F}(f)"] \arrow[d, "\alpha_A"'] & \mathcal{F}B \arrow[d, "\alpha_B"] \\
\mathcal{G}A \arrow[r, "\mathcal{G}(f)"'] & \mathcal{G}B
\end{tikzcd}$$

Определение 7. *Моноидальная категория*

Моноидальная категория – это категория $\mathcal{C}$ с дополнительной структурой:

- *Бифунктор $\otimes : \mathcal{C} \times \mathcal{C} \to C$, который мы будем называть тензором;*
- *Единица $\mathbb{1}$;*
- *Изоморфизм, который мы будем называть ассоциатором: для любых $A, B, C \in Ob_{\mathcal{C}}, \alpha_{A,B,C} : (A \otimes B) \otimes C \cong A \otimes (B \otimes C)$;*
- *Изоморфизм $L_A : \mathbb{1} \otimes A \cong A$;*
- *Изоморфизм $R_A : A \otimes \mathbb{1} \cong A$;*

- *Первое условие когерентности (пятиугольник Маклейна) (данная диаграмма коммутирует):*

$$\begin{tikzcd}
 & (A\otimes(B\otimes C))\otimes D \arrow[dr, "\alpha_{A,B\otimes C,D}"] & \\
((A\otimes B)\otimes C)\otimes D \arrow[ur, "\alpha_{A,B,C}\otimes id_D"] \arrow[d, "\alpha_{A\otimes B,C,D}"'] & & A\otimes((B\otimes C)\otimes D) \arrow[d, "id_A\otimes\alpha_{B,C,D}"] \\
(A\otimes B)\otimes(C\otimes D) \arrow[rr, "\alpha_{A,B,C\otimes D}"'] & & A\otimes(B\otimes(C\otimes D))
\end{tikzcd}$$

- *Второе условие когерентности (тождество треугольника):*

$$\begin{tikzcd}
(A\otimes\mathbb{1})\otimes B \arrow[rr, "\alpha_{A,\mathbb{1},B}"] \arrow[dr, "R_A\otimes id_B"'] & & A\otimes(\mathbb{1}\otimes B) \arrow[dl, "id_A\otimes L_B"] \\
 & A\otimes B &
\end{tikzcd}$$

Определение 8. *Декартово замкнутая категория*

Декартово замкнутая категория – это категория с терминальным объектом, произведениями и экспоненцированием:

1) Объект $\mathbb{1}$ в категории $\mathcal{C}$ называется терминальных, если для любого объекта $A \in Ob_{\mathcal{C}}$ и для любых морфизмов $f, g \in Hom_{\mathcal{C}}(A, 1)$, $f = g$.

2) Пусть $A, B \in Ob_{\mathcal{C}}$, тогда произведением объектов A и B называется объект $A \times B$, такой, для любого $C \in Ob_{\mathcal{C}}$ и для любых морфизмов $f \in Hom_{\mathcal{C}}(C, A)$ и $g \in Hom_{\mathcal{C}}(C, B)$, что диаграмма коммутирует:

$$\begin{tikzcd}
 & C \arrow[dl, "g"'] \arrow[d, "\langle f,g\rangle"] \arrow[dr, "f"] & \\
A & A\times B \arrow[l, "\pi_1"] \arrow[r, "\pi_2"'] & B
\end{tikzcd}$$

Морфизм $\langle f, g\rangle$ называется парой морфизмов f и g, а морфизмы вида π_1 и π_2 – каноническими проекциями.

3) Пусть $A, B \in Ob_{\mathcal{C}}$, тогда экспонентой объектов A и B называется объект B^A, такой, что диаграмма коммутирует для любого объекта $C \in Ob_{\mathcal{C}}$ и для любого морфизма $f \in Hom_{\mathcal{C}}(C \times A, B)$:

$$\begin{tikzcd}
B^A\times A \arrow[r, "\epsilon_{A,B}"] & B \\
C\times A \arrow[u, "\Lambda(f)\times id_A"] \arrow[ur, "f"'] &
\end{tikzcd}$$

где $\Lambda(f) \times id_A = \langle\Lambda(f) \circ \pi_1, id_A \circ \pi_2\rangle$

Морфизмы вида $\epsilon_{A,B}$ называются вычисляющими стрелками, а морфизмы вида $\Lambda(f)$ – каррированием стрелки f.

Определение 9. *Моноидальный функтор*

Пусть $\langle\mathcal{C}, \otimes_1, \mathbb{1}_{\mathcal{C}}\rangle$ и $\langle\mathcal{D}, \otimes_2, \mathbb{1}_{\mathcal{D}}\rangle$ моноидальные категории.

Моноидальный функтор $\mathcal{F} : \langle\mathcal{C}, \otimes_1, \mathbb{1}\rangle \to \langle\mathcal{D}, \otimes_2, \mathbb{1}'\rangle$ это функтор $\mathcal{F} : \mathcal{C} \to \mathcal{D}$ с дополнительными естественными преобразованиями:

- $u : \mathbb{1}_{\mathcal{D}} \to \mathcal{F}\mathbb{1}_{\mathcal{C}}$;
- $*_{A,B} : \mathcal{F}A \otimes_{\mathcal{D}} \mathcal{F}B \to \mathcal{F}(A \otimes_{\mathcal{C}} B)$.

и условиями когерентности:

- *Ассоциативность:*

$$\begin{tikzcd}
(\mathcal{F}A \otimes_{\mathcal{D}} \mathcal{F}B) \otimes_{\mathcal{D}} \mathcal{F}C \arrow[r, "\alpha^{\mathcal{D}}_{\mathcal{F}A,\mathcal{F}B,\mathcal{F}C}"] \arrow[d, "*_{A,B} \otimes_{\mathcal{D}} id_{\mathcal{F}B}"'] & \mathcal{F}A \otimes_{\mathcal{D}} (\mathcal{F}B \otimes_{\mathcal{D}} \mathcal{F}C) \arrow[d, "id_{\mathcal{F}A} \otimes_{\mathcal{D}} *_{B,C}"] \\
\mathcal{F}(A \otimes_{\mathcal{C}} B) \otimes_{\mathcal{D}} \mathcal{C} \arrow[d, "*_{A \otimes_{\mathcal{C}} B, C}"'] & \mathcal{F}A \otimes_{\mathcal{D}} \mathcal{F}(B \otimes_{\mathcal{C}} C) \arrow[d, "*_{A, B \otimes_{\mathcal{C}} C}"] \\
\mathcal{F}((A \otimes_{\mathcal{C}} B) \otimes_{\mathcal{C}} C) \arrow[r, "\mathcal{F}(\alpha^{\mathcal{C}}_{A,B,C})"'] & \mathcal{F}(A \otimes_{\mathcal{C}} (B \otimes_{\mathcal{C}} C))
\end{tikzcd}$$

- *Свойство левой единицы:*

$$\begin{tikzcd}
\mathbb{1}_{\mathcal{D}} \otimes_{\mathcal{D}} \mathcal{F}A \arrow[r, "u \otimes_{\mathcal{D}} id_{\mathcal{F}A}"] \arrow[d, "L^{\mathcal{D}}_{\mathcal{F}A}"'] & \mathcal{F}\mathbb{1}_{\mathcal{C}} \otimes_{\mathcal{D}} \mathcal{F}A \arrow[d, "*_{\mathbb{1}_{\mathcal{C}},A}"] \\
\mathcal{F}A & \mathcal{F}(\mathbb{1}_{\mathcal{C}} \otimes_{\mathcal{C}} A) \arrow[l, "\mathcal{F}(L^{\mathcal{C}}_A)"']
\end{tikzcd}$$

- *Свойство правой единицы:*

$$\begin{tikzcd}
\mathcal{F}A \otimes_{\mathcal{D}} \mathbb{1}_{\mathcal{D}} \arrow[r, "id_{\mathcal{F}A} \otimes_{\mathcal{D}} u"] \arrow[d, "R^{\mathcal{D}}_{\mathcal{F}A}"'] & \mathcal{F}A \otimes_{\mathcal{D}} \mathcal{F}\mathbb{1}_{\mathcal{C}} \arrow[d, "*_{A,\mathbb{1}_{\mathcal{C}}}"] \\
\mathcal{F}A & \mathcal{F}(A \otimes_{\mathcal{C}} \mathbb{1}_{\mathcal{C}}) \arrow[l, "\mathcal{F}(R^{\mathcal{C}}_A)"]
\end{tikzcd}$$

Определение 10. *Аппликативный функтор*

Аппликативный функтор – это тройка $\langle \mathcal{C}, \mathcal{K}, \eta \rangle$*, где* $\mathcal{C}$ *– это моноидальная категория,* $\mathcal{K}$ *- это моноидальный эндофунктор и* $\eta : Id_{\mathcal{C}} \Rightarrow \mathcal{K}$ *– это естественное преобразование, такое, что:*

- $u = \eta_{\mathbb{1}}$;

- $*_{A,B} \circ (\eta_A \otimes \eta_B) = \eta_{A \otimes B}$, *то есть диаграмма коммутирует:*

$$\begin{tikzcd}
A \otimes B \arrow[r, "\eta_A \otimes \eta_B"] \arrow[dr, "\eta_{A \otimes B}"'] & \mathcal{K}A \otimes \mathcal{K}B \arrow[d, "*_{A,B}"] \\
& \mathcal{K}(A \otimes B)
\end{tikzcd}$$

По умолчанию мы будем рассматривать ниже аппликативный функтор над декартово замкнутой категорией.

2 Введение

2.1 Обзор имеющихся результатов

Модальная теория типов – сравнительно молодая область современной математической логики, изучающая конструктивные модальные логики с точки зрения вычислений, в котором каждому объекту, участвующему в вычислительной процедуре приписан свой тип данных, представляющий тот или иной объект.

Если в обычной теории типов, рассматриваются системы типов соответствующие по Карри-Говарду конструктивным логикам (исчисление высказываний, исчисление высказываний второго порядка, исчисления предикатов первого или высших порядков, и т.д.) [11] [12] [14] [16], то в модальной теории типов изучатся системы типов, изоморфные по Карри-Говарду, интуиционистским модальным логикам.

В модальных теориях типов модальности рассматриваются не как операции над высказываниями, присоединяющие к ним вводные конструкции ("всегда было, что", "необходимо, что", "агент i знает, что", и т. д.), но как операторы над типами данных, позволяющие по одним типам данных получать другие.

То есть модальности при таком подходе не делятся на алетические, временные, эпистемические и прочие, а рассматриваются с вычислительной точки зрения. Текущее исследование также будет придерживаться данного подхода. Достаточно подробно имеющиейся результаты в данной области описаны в следующей обзорной статье [25].

2.2 Задача исследования

Объяснение задач исследования требует нескольких предварительных определений:

2.2.1 Логика IEL$^-$

Модальная интуиционистская логика IEL$^-$ была предложена С. Артемовым и Т. Протопопеску [1]. IEL$^-$ предлагает свою формальную теорию интуиционистских убеждений, согласанную с BHK-семантикой интуиционистской логики.

Неформально $\mathbf{K}A$ означает, что A верифицировано интуиционистки.

Логика IEL$^-$ следующими схемами аксиом и правилами вывода:

Определение 11. *Интуиционистская модальная логика IEL$^-$:*

1) Аксиомы интуиционистского исчисления высказываний;

2) $\mathbf{K}(A \to B) \to (\mathbf{K}A \to \mathbf{K}B)$ (нормальность);

3) $A \to \mathbf{K}A$ (ко-рефлексия);

Правило вывода: MP.

Далее мы будем обозначать модальность как $\Box$.

Легко видеть, что правило усиления в этой логике является производным.

В. Крупский и А. Ятманов построили секвенциальное исчисление для логики IEL (IEL$^-$ + $\mathbf{K}A \to \neg\neg A$) и показали, что задача поиска вывода в данной логике PSPACE-полна [2].

2.2.2 Мотивация из функционального программирования

Функциональные языки программирования, такие, как Haskell [3], Idris [4], Purescript [5] или Elm [6] содержат специальные классы типов [3] для вычислений с типами, вложенных в вычислительных контекст. Основные и наиболее интересные нам классы типов: `Functor` и `Applicative` [4]:

```
class Functor f where
  fmap :: (a -> b) -> f a -> f b


class Functor f => Applicative f where
  pure :: a -> f a
  (<*>) :: f (a -> b) -> f a -> f b
```

Вычислительным контекстом (или *контейнером*) мы называем оператор над типами f, где f – это "функция" из $*$ в $*$: типовой оператор берет простой тип (имеющий сорт $*$) и возвращает другой простой тип сорта `*`. Более подробное описание системы типов с сортами, используемая для этих целей в Haskell, описана здесь [12].

Из сигнатуры метода `fmap` видно, что имея функцию из типа a в тип b и имея значение типа a, к которому применен оператор f, мы можем получить значение типа b, к которому также применен типовой операторй f.

Иными словами, `fmap` позволяет пронести одноместную функцию между обычными типами через контейнер f.

Аппликативный функтор позволяет обобщить действие функтора для функций произвольной арности:

```
liftA2 :: Applicative f => (a -> b -> c) -> f a -> f b -> f c
liftA2 f x y = ((pure f) <*> x) <*> y
```

Легко видеть, что модальные аксиомы IEL^- и типы методов класса `Applicative` синтаксически идентичны, и мы собираемся придать этому совпадению вычислительный смысл.

Мы построим модальное λ-исчисление, которое будет изоморфно по Карри-Говарду логике IEL^-, предложим операционную семантику для данного исчисления и покажем необходимые метатеоретические свойства полученной системы: редукция субъекта (о сохранности типов в процессе вычисления), сильная нормализуемость (о конечности всех цепочек вычисления) и свойство Черча-Россера.

Мы покажем также, что полученная система корректна и полна относительно произвольного аппликативного функтора над декартово замкнутой категорией, используя обобщение категорного определения аппликативного функтора, предложенного Патерсоном [26].

---

[3] Класс типов – это общий интерфейс (наличие одних и тех же методов) для специальной группы типов данных.

[4] Читатель может более подробно узнать о данной разновидности вычислений в стандартной библиотеке языка [7] или в следующей книге [8]

3 Модальное λ-исчисление, основанное на исчислении IEL$^-$

3.1 Натуральный вывод для IEL$^-$

Определим натуральное исчисление для IEL$^-$:

Определение 12. *Натуральное исчисление NIEL$^-$ для интуиционистской эпистемической логики IEL$^-$ – это расширение натурального исчисления для интуиционистской логики высказываний с добавлением следующих правил вывода для модальности:*

$$\frac{\Gamma \vdash A}{\Gamma \vdash \Box A}\Box_I \qquad \frac{\Gamma \vdash \Box A_1, \ldots, \Gamma \vdash \Box A_n \qquad A_1, \ldots, A_n \vdash B}{\Gamma \vdash \Box B}$$

Первое правило позволяет выводить ко-рефлексию. Второе модальное правило – это аналог для правила $\Box_I$ в натуральном исчислении для конструктивной K (see [25]) без $\Diamond$.

Мы будем обозначать $\Gamma \vdash \Box A_1, \ldots, \Gamma \vdash \Box A_n$ и $A_1, \ldots, A_n \vdash B$ соответственно как $\Gamma \vdash \mathbf{K}\vec{A}$ и $\vec{A} \vdash B$ для краткости.

Лемма 1. $\Gamma \vdash_{NIEL^-} A \Rightarrow IEL^- \vdash \bigwedge \Gamma \to A$.

Доказательство. Индукция по построению вывода. Рассмотрим модальные случаи.

1) Если $\Gamma \vdash_{\mathrm{NIEL}^-} A$, тогда $\mathrm{IEL}^- \vdash \bigwedge \Gamma \to \Box A$.

(1)	$\bigwedge \Gamma \to A$	предположение индукции
(2)	$A \to \Box A$	ко-рефлексия
(3)	$(\bigwedge \Gamma \to A) \to ((A \to \Box A) \to (\bigwedge \Gamma \to \Box A))$	теорема IEL$^-$
(4)	$(A \to \Box A) \to (\bigwedge \Gamma \to \Box A)$	из (1), (3) и MP
(5)	$\bigwedge \Gamma \to \Box A$	из (2), (4) и MP

2) Если $\Gamma \vdash_{\mathrm{NIEL}^-} \Box\vec{A}$ и $\vec{A} \vdash B$, то $\mathrm{IEL}^- \vdash \bigwedge \Gamma \to \Box B$.

(1)	$\bigwedge \Gamma \to \bigwedge\limits_{i=1}^{n} \Box A_i$	предположение индукции
(2)	$\bigwedge\limits_{i=1}^{n} \Box A_i \to \Box \bigwedge\limits_{i=1}^{n} A_i$	теорема IEL$^-$
(3)	$\bigwedge \Gamma \to \Box \bigwedge\limits_{i=1}^{n} A_i$	по (1), (2) и правилу силлогизма
(4)	$\bigwedge\limits_{i=1}^{n} A_i \to B$	предположение индукции
(5)	$(\bigwedge\limits_{i=1}^{n} A_i \to B) \to \Box(\bigwedge\limits_{i=1}^{n} A_i \to B)$	ко-рефлексия
(6)	$\Box(\bigwedge\limits_{i=1}^{n} A_i \to B)$	из (4), (5) и MP
(7)	$\Box \bigwedge\limits_{i=1}^{n} A_i \to \Box B$	по (6) и по нормальности
(8)	$\bigwedge \Gamma \to \Box B$	по (3), (7) и правилу силлогизма

$\Box$

Лемма 2. *Если IEL$^-$ $\vdash A$, то NIEL$^-$ $\vdash A$.*

Доказательство. Построение выводов для модальных аксиом в NIEL$^-$. Мы рассмотрим эти выводы ниже с использованием термов. $\Box$

3.2 Модальное λ-исчисление $\lambda_{\mathbf{K}}$

Далее мы построим типизированное λ-исчисление по фрагменту NIEL^- с правилами для импликации, конъюнкции и модальности. Данный фрагмент эквивалентен IEL^- без аксиом для отрицания и дизъюнкции, что элементарно проверяется аналогично.

Определим термы и типы:

Определение 13. *Множество термов:*

Пусть $\mathbb{V}$ счетное множество переменных. Термы $\Lambda_{\mathbf{K}}$ порождается следующей грамматикой:

$$\Lambda_{\mathbf{K}} ::= \mathbb{V} \mid (\lambda\mathbb{V}.\Lambda_{\mathbf{K}}) \mid (\Lambda_{\mathbf{K}}\Lambda_{\mathbf{K}}) \mid (\Lambda_{\mathbf{K}}, \Lambda_{\mathbf{K}}) \mid (\pi_1\Lambda_{\mathbf{K}}) \mid (\pi_2\Lambda_{\mathbf{K}}) \mid$$
$$(\mathbf{pure}\ \Lambda_{\mathbf{K}}) \mid (\mathbf{let\ pure}\ \mathbb{V}^* = \Lambda_{\mathbf{K}}^* \ \mathbf{in}\ \Lambda_{\mathbf{K}})$$

Где $\mathbb{V}^*$ и $\Lambda_{\mathbf{K}}^*$ обозначают множество всех конечных последовательностей переменных $\bigcup\limits_{i=0}^{\infty} \mathbb{V}^i$ и множество всех конечных последовательностей термов $\bigcup\limits_{i=0}^{\infty} {\Lambda_{\mathbf{K}}}^i$. Последовательность переменных $\vec{x}$ и последовательность термов $\vec{M}$ в терме вида **let pure** должны иметь одинаковую длину. Иначе терм не будет правильно построенным.

Определение 14. *Множество типов:*

Пусть $\mathbb{T} = \{A_1, A_2, A_3, \dots\}$ – это счетное множество атормарных типов. Типы $\mathbb{T}_{\mathbf{K}}$ с типовым оператором $\Box$ порождаются следующей грамматикой:

$$\mathbb{T}_{\mathbf{K}} ::= \mathbb{T} \mid (\mathbb{T}_{\mathbf{K}} \to \mathbb{T}_{\mathbf{K}}) \mid (\mathbb{T}_{\mathbf{K}} \times \mathbb{T}_{\mathbf{K}}) \mid (\Box\mathbb{T}_{\mathbf{K}}) \tag{1}$$

Контекст, его домен и кодомен определены стандартно [11][12].

Наша система состоит из следующих правил типизации в стиле Карри:

Определение 15. *Модальное λ-исчисление, основанное на исчислении IEL^-:*

$$\frac{}{\Gamma, x : A \vdash x : A}\ ax$$

$$\frac{\Gamma, x : A \vdash M : B}{\Gamma \vdash \lambda x.M : A \to B} \to_i \qquad \frac{\Gamma \vdash M : A \to B \qquad \Gamma \vdash N : A}{\Gamma \vdash MN : B} \to_e$$

$$\frac{\Gamma \vdash M : A \qquad \Gamma \vdash N : B}{\Gamma \vdash \langle M, N\rangle : A \times B} \times_i \qquad \frac{\Gamma \vdash M : A_1 \times A_2}{\Gamma \vdash \pi_i M : A_i} \times_e,\ i \in \{1,2\}$$

$$\frac{\Gamma \vdash M : A}{\Gamma \vdash \mathbf{pure}\ M : \mathbf{K}A} \Box_I \qquad \frac{\Gamma \vdash \vec{M} : \Box\vec{A} \qquad \vec{x} : \vec{A} \vdash N : B}{\Gamma \vdash \mathbf{let\ pure}\ \vec{x} = \vec{M}\ \mathbf{in}\ N : \Box B}\ let_{\Box}$$

Правило типизации $\Box$ аналогично правилу $\bigcirc_I$ в монадическом метаязыке [17].

$\Box_I$ позволяет вкладывать объект типа A в текущий вычислительный контекст, изменяя его тип на $\Box A$.

Правило типизации $\mathrm{let}_{\Box}$ аналогично правилу $\Box$-правилу в модальном λ-исчислении для интуционистской минимальной нормальной модальной логики **IK** [19].

$\Gamma \vdash \vec{M} : \Box\vec{A}$ – это синтаксический сахар для $\Gamma \vdash M_1 : \Box A_1, \dots, \Gamma \vdash M_n : \Box A_n$ и $\vec{x} : \vec{A} \vdash N : B$ – это краткая форма для $x_1 : A_1, \dots, x_n : A_n \vdash N : B$. $\mathbf{let\ pure}\ \vec{x} = \vec{M}\ \mathbf{in}\ N$ – это мгновенное локальное связывание в терме N. Мы будем использовать такую краткую форму вместо $\mathbf{let\,pure}\, x_1, \dots, x_n = M_1, \dots, M_n \,\mathbf{in}\, N$.

Примеры выводов:

$$\dfrac{\dfrac{x : A \vdash x : A}{x : A \vdash \mathbf{pure}\, x : \Box A}}{\vdash (\lambda x.\mathbf{pure}\, x) : A \to \Box A}$$

$$\dfrac{\dfrac{\dfrac{f : \Box(A \to B) \vdash f : \Box(A \to B) \qquad x : \Box A \vdash x : \Box A \qquad \dfrac{g : A \to B \vdash g : A \to B \qquad y : A \vdash y : A}{g : A \to B, y : A \vdash gy : B}\to_e}{f : \Box(A \to B), x : \Box A \vdash \mathbf{let\,pure}\, g, y = f, x \,\mathbf{in}\, gy : \Box B}\mathrm{let}_{\mathbf{K}}}{f : \Box(A \to B) \vdash \lambda x.\mathbf{let\,pure}\, g, y = f, x \,\mathbf{in}\, gy : \Box A \to \Box B}}{\vdash \lambda f.\lambda x.\mathbf{let\,pure}\, g, y = f, x \,\mathbf{in}\, gy : \Box(A \to B) \to \Box A \to \Box B}$$

Нетрудно видеть, что данные примеры деревьев вывода являются в точности размеченными термами выводами модальных аксиом IEL^{-}.

Определим свободные переменные, подставновку, β-редукцию и η-редукцию. Многошаговая β-редукция и $\beta\eta$-эквивалентность определены стандартно:

Определение 16. *Множество свободных переменных $FV(M)$ для произвольного терма M:*

1) $FV(x) = \{x\}$;
2) $FV(\lambda x.M) = FV(M) \backslash \{x\}$;
3) $FV(MN) = FV(M) \cup FV(N)$;
4) $FV(\langle M, N\rangle) = FV(M) \cup FV(N)$;
5) $FV(\pi_i M) \subseteq FV(M)$, $i \in \{1, 2\}$;
6) $FV(pure\ M) = FV(M)$;
7) $FV(\mathbf{let\,pure}\, \vec{x} = \vec{M} \ \mathbf{in}\ N) = \bigcup\limits_{i=1}^{n} FV(M)$, *где* $n = |\vec{M}|$.

Во избежание лишних коллизий мы будем полагать, что если терм вида $\mathbf{let\,pure}\, x = (\mathbf{let\,pure}\, \vec{y} = \vec{N} \,\mathbf{in}\, P) \,\mathbf{in}\, M$ типизируется, то x не содержится в $\vec{y}$ (как следствие, свободные переменные термов M и $\vec{N}$ не должны пересекаться), то есть последовательное применение локальных связываний требует на каждом шаге различных переменных.

Определение 17. *Подстановка:*

1) $x[x := N] = N$, $x[y := N] = x$;
2) $(MN)[x := N] = M[x := N]N[x := N]$;
3) $(\lambda x.M)[x := N] = \lambda x.M[y := N]$, $y \in FV(M)$;
4) $(M, N)[x := P] = (M[x := P], N[x := P])$;
5) $(\pi_i M)[x := P] = \pi_i(M[x := P])$, $i \in \{1, 2\}$;
6) $(\mathbf{pure}\, M)[x := P] = \mathbf{pure}\,(M[x := P])$;
7) $(\mathbf{let\,pure}\, \vec{x} = \vec{M} \,\mathbf{in}\, N)[y := P] = \mathbf{let\,pure}\, \vec{x} = (\vec{M}[y := P]) \,\mathbf{in}\, N$.

Определение 18. *Правила β-редукции и η-редукции:*

1) $(\lambda x.M)N \to_\beta M[x := N]$;
2) $\pi_1\langle M, N\rangle \to_\beta M$;
3) $\pi_2\langle M, N\rangle \to_\beta N$;

4) $\begin{array}{l}\textbf{let pure}\ \vec{x}, y, \vec{z} = \vec{M}, \textbf{let pure}\ \vec{w} = \vec{N}\ \textbf{in}\ Q, \vec{P}\ in\ R \to_{\beta\Box} \\ \textbf{let pure}\ \vec{x}, \vec{w}, \vec{z} = \vec{M}, \vec{N}, \vec{P}\ \textbf{in}\ R[y := Q]\end{array}$

5) $\textbf{let pure}\ \vec{x} = \textbf{pure}\ \vec{M}\ \textbf{in}\ N \to_{\beta\Box\mathbf{pure}} \textbf{pure}\ N[\vec{x} := \vec{M}]$

6) $\textbf{let pure}\ \_\_ = \_\_\ \textbf{in}\ M \to_{\beta\mathbf{nec}} \textbf{pure}\ M$*, где* __ *– это пустая последовательность термов.*

7) $\lambda x.fx \to_\eta f$*;*

8) $\langle \pi_1 P, \pi_2 P\rangle \to_\eta P$*;*

9) $\textbf{let pure}\ x = M\ \textbf{in}\ x \to_{\Box id} M$*;*

Мы будет писать $M \to_r N$, если терм M редуцируется к терму N за один шаг по одному из перечисленных выше правил.

Определение 19. *Многошаговая редукция* $\twoheadrightarrow_r$*.*

Многошаговой редукцией $\twoheadrightarrow_r$ *является рефлексивно-транзитивное замыкание одношаговой редукции* $M \to_r N$*.*

По умолчанию мы используем стратегию вычисления с вызовом по имени.

3.3 Леммы о контекстах

Докажем стандартные леммы о контекстах [5]:

Лемма 3. *Инверсия отношения типизации* $\Box_I$*.*

Пусть $\Gamma \vdash \textbf{pure}\ M : \Box A$*, тогда* $\Gamma \vdash M : A$*;*

Доказательство. Очевидно. □

Лемма 4. *Базовые леммы.*

- *Если* $\Gamma \vdash M : A$ *и* $\Gamma \subseteq \Delta$*, тогда* $\Delta \vdash M : A$*;*
- *Если* $\Gamma \vdash M : A$*, тогда* $\Delta \vdash M : A$*, где* $\Delta = \{x_i : A_i \mid (x_i : A_i) \in \Gamma\ \&\ x_i \in FV(M)\}$
- *Если* $\Gamma, x : A \vdash M : B$ *и* $\Gamma \vdash N : A$*, где* $\Gamma \vdash M[x := N] : B$*.*

Рассмотрим случаи для правила $\text{let}_\Box$.

Доказательство.

1) Пусть вывод заканчивается следующим правилом:

$$\frac{\Gamma \vdash \vec{M} : \Box\vec{A} \qquad \vec{x} : \vec{A} \vdash N : B}{\Gamma \vdash \textbf{let pure}\ \vec{x} = \vec{M}\ \textbf{in}\ N : \Box B}\ \text{let}_\Box$$

По предположению индукции $\Delta \vdash \vec{M} : \mathbf{K}\vec{A}$, тогда $\Delta \vdash \textbf{let pure}\ \vec{x} = \vec{M}\ \textbf{in}\ N : \Box B$.

Случаи 2)–3) рассматриваются аналогично.

□

[5] Мы не будем рассматривать случаи для стандартных связок, так как они уже доказаны для просто типизированного λ-исчисления [11] [12]. Мы будем рассматривать только модальные случаи

3.4 Метатеоретические свойства системы

Теорема 1. *Редукция субъекта*

Если $\Gamma \vdash M : A$ *и* $M \twoheadrightarrow_{\beta\eta} N$, *тогда* $\Gamma \vdash N : A$

Доказательство. Индукция по выводу $\Gamma \vdash M : A$ и по порождению $\rightarrow_{\beta\eta}$.

Случаи с функцией и парами рассмотрены здесь [12] [13].

1) Если $\Gamma \vdash \mathbf{let\,pure}\,\vec{x}, y, \vec{z} = \vec{M}, \mathbf{let\,pure}\,\vec{w} = \vec{N}\,\mathbf{in}\,Q, \vec{P}\,\mathbf{in}\,R : \Box B$, тогда $\Gamma \vdash \mathbf{let\,pure}\,\vec{x}, \vec{w}, \vec{z} = \vec{M}, \vec{N}, \vec{P}\,\mathbf{in}\,R[y := Q] : \Box B$ по правилу 4).

2) Если $\Gamma \vdash \mathbf{let\,pure}\,x = M\,\mathbf{in}\,x : \Box A$, тогда $\Gamma \vdash M : \mathbf{K}A$ по правилу 9).

Рассмотрено здесь [19].

3) Пусть вывод заканчивается применением следующего правила

$$\frac{\Gamma \vdash \mathbf{pure}\,\vec{M} : \Box\vec{A} \qquad \vec{x} : \vec{A} \vdash N : B}{\Gamma \vdash \mathbf{let\,pure}\,\vec{x} = \mathbf{pure}\,\vec{M}\,\mathbf{in}\,N : \Box B}$$

Тогда $\Gamma \vdash \vec{M} : \vec{A}$ по инверсии отношения типизации для $\Box_I$ и $\Gamma \vdash N[\vec{x} := \vec{M}] : B$ по лемме 4, часть 3.

Тогда мы можем преобразовать данный вывод в следующий:

$$\frac{\Gamma \vdash N[\vec{x} := \vec{M}] : B}{\Gamma \vdash \mathbf{pure}\,N[\vec{x} := \vec{M}] : \Box B}\,\Box_I$$

4) Пусть вывод заканчивается применением правила $let_\Box$ для типового объявления, выводимого из пустого контекста:

$$\frac{\vdash M : A}{\vdash \mathbf{let\,pure}\,\_\_ = \_\_\,\mathbf{in}\,M : \Box A}$$

Тогда, если $\vdash M : A$, тогда $\vdash \mathbf{pure}\,M : \Box A$.

Данное рассуждение действует также и в обратную сторону. $\Box$

Теорема 2.

$\twoheadrightarrow_r$ *сильно нормализуемо;*

Доказательство.

Построим отображение из $\lambda_{\mathbf{K}}$ в просто типизированное λ-исчисление с типами $\rightarrow$, $\times$ и выделенным типом натуральных чисел $\mathbb{N}$, для которого есть следующие правила типизации и редукции:

$$\frac{}{\Gamma \vdash 0 : \mathbb{N}}$$

$$\frac{\Gamma \vdash n : \mathbb{N}}{\Gamma \vdash \mathbf{succ}\,n : \mathbb{N}}$$

$$\frac{\Gamma \vdash n : \mathbb{N} \qquad \Gamma \vdash m : \mathbb{N}}{\Gamma \vdash n + m : \mathbb{N}}$$

- $n + 0 \rightarrow_\beta n$;
- $(n + \mathbf{succ}\,m) \rightarrow_\beta \mathbf{succ}\,(n + m)$

Определим перевод $|.|$ между данными исчислениями отдельно на типах, и на термах

Определение 20. *Интерпретация типов*

- $A \in \mathbb{T} \Rightarrow |A| = A$;
- $|A \to B| = |A| \to |B|$;
- $|A \times B| = |A| \times |B|$;
- $|\Box A| = \mathbb{N} \times |A|$.

Определение 21. *Интерпретация термов*

- $x \in \mathbb{V} \Rightarrow |x| = x$;
- $|\lambda x.M| = \lambda x.|M|$;
- $|(MN)| = |M||N|$;
- $|\langle M, N\rangle| = \langle |M|, |N|\rangle$;
- $|\pi_i M| = \pi_i |M|,\ i \in \{1, 2\}$;
- $|\mathbf{pure}\, M| = \langle 0, |M|\rangle$;
- $|\mathbf{let\ pure}\, \_\_ = \_\_\ \mathbf{in}\ M| = \langle 0, M\rangle$
- $|\mathbf{let\ pure}\, \vec{x} = \vec{N}\ \mathbf{in}\ M| = \langle \sum\limits_{i=1}^{n} \pi_1|N|, |M|[\vec{x} := \pi_2\vec{N}]\rangle$

Рассмотрим интерпретацию последнего терма с помощью интерпретации правила типизации:

$$\frac{|\Gamma \vdash \vec{N} : \Box\vec{A}| = |\Gamma| \vdash |\vec{N}| : \mathbb{A} \times |\vec{A}| \qquad |\vec{x} : \vec{A} \vdash M : B| = \vec{x} : |\vec{A}| \vdash |M| : |B|}{|\Gamma \vdash \mathbf{let\ pure}\, \vec{x} = \vec{N}\ \mathbf{in}\ M : \Box B| = |\Gamma| \vdash \langle \sum\limits_{i=1}^{n} \pi_1|N|, |M|[\vec{x} := \pi_2\vec{N}]\rangle : \mathbb{N} \times |B|}\ \mathrm{let}_\Box$$

Лемма 5. *Интерпретация сохраняет подстановку:*

$|M[x := N]| = |M|[x := |N|]$ *для произвольного терма* M.

Доказательство. Несложная индукция по длине M. ☐

Лемма 6. $M \twoheadrightarrow_r N \Rightarrow |M| \twoheadrightarrow_{\beta\eta} |N|$

Доказательство. Рассмотрим случаи с $\beta\Box$, $\beta\Box$**pure** и $\Box id$.

1)

$|\mathbf{let\ pure}\, x = (\mathbf{let\ pure}\, y = N\ \mathbf{in}\ P)\ \mathbf{in}\ M| =$

По интерпретации

$\langle \pi_1|N|, |M|[x := |P|[y := \pi_2|N|]]\rangle$

$|\mathbf{let\ pure}\, y = N\ \mathbf{in}\ M[x := P]| =$

$\langle \pi_1|N|, |M|[x := |P|][y := \pi_2|N|]\rangle \qquad \equiv$

По лемме Барендрегта по подстановке

$\langle \pi_1|N|, |M|[y := \pi_2|N|][x := |P|[y := \pi_2|N|]]\rangle \qquad \equiv$

Поскольку $y \notin FV(M)$

$\langle \pi_1|N|, |M|[x := |P|[y := \pi_2|N|]]$

2)

$|\mathbf{let\ pure}\,\vec{x} = \mathbf{pure}\,\vec{N}\ \mathbf{in}\ M| =$

По интерпретации

$\langle 0 + \cdots + 0, |M|[\vec{x} := |\vec{N}|]\rangle \twoheadrightarrow_\beta$

Многошаговая редукция для натуральных чисел

$\langle 0, |M|[\vec{x} := |\vec{N}|]\rangle =$

По интерпретации

$|\mathbf{pure}\,M[\vec{x} := \vec{N}]|$

3)

$|\mathbf{let\ pure}\,x = M\ \mathbf{in}\ x| =$

По интерпретации

$\langle \pi_1|M|, x[x := \pi_2|M|]\rangle =$

Подстановка $\square$

$\langle \pi_1|M|, \pi_2|M|\rangle \to_\eta$

Правило η-редукции для пары

$|M|$

Таким образом, мы показали, что $\lambda_{\mathbf{K}}$ корректно относительно $\lambda_{\to,\times,\mathbb{N}}$, тогда $\lambda_{\mathbf{K}}$ сильно нормализуемо, поскольку $\lambda_{\to,\times,\mathbb{N}}$ сильно нормализуемо.

$\square$

Теорема 3. *Свойство Черча-Россера*

$\twoheadrightarrow_r$ *конфлюентно.*

Доказательство. По лемме Ньюмана, если отношение сильно нормализуемо и локально конфлюентно, то отношение конфлюентно.

Достаточно показать локальную конфлюентность.

Лемма 7. *Локальная конфлюентность.*

Если $M \to_r N$ *и* $M \to_r Q$*, тогда найдется такой терм* P*, что* $N \twoheadrightarrow_r P$ *и* $Q \twoheadrightarrow_r P$*.*

Доказательство. Рассмотрим данную критическую пару и покажем, что оба терма из данной пары редуцируются к одному и тому же терму:

$$\begin{tikzcd}
\mathbf{let\ pure}\,x = (\mathbf{let\ pure}\,\vec{y} = \mathbf{pure}\,\vec{N}\ \mathbf{in}\ P)\ \mathbf{in}\ M \arrow[d, "\beta\square"'] \arrow[dr, "\beta\square\mathbf{pure}"] & \\
\mathbf{let\ pure}\,\vec{y} = \mathbf{pure}\,\vec{N}\ \mathbf{in}\ M[x := P] & \mathbf{let\ pure}\,x = \mathbf{pure}\,P[\vec{y} := \vec{N}]\ \mathbf{in}\ M
\end{tikzcd}$$

$\mathbf{let\ pure}\,\vec{y} = \mathbf{pure}\,\vec{N}\ \mathbf{in}\ M[x := P] \to_{\beta\square\mathbf{pure}}$

$\mathbf{pure}\,M[x := P][\vec{y} = \vec{N}]$

$\mathbf{let\ pure}\,x = \mathbf{pure}\,P[\vec{y} := \vec{N}]\ \mathbf{in}\ M \to_{\beta\square\mathbf{pure}}$

$\mathbf{pure}\,M[x := P[\vec{y} := \vec{N}]]$

$\square$

По лемме о подстановке

$\mathbf{pure}\,M[x := P][\vec{y} = \vec{N}] \equiv \mathbf{pure}\,M[\vec{y} = \vec{N}][x := P[\vec{y} := \vec{N}]]$

По нашему соглашению, $x \notin \vec{y}$, тогда

$M[\vec{y} = \vec{N}][x := P[\vec{y} := \vec{N}]] \equiv M[x := P[\vec{y} := \vec{N}]]$

□

Теорема 4.

Нормальная форма $\lambda_{\mathbf{K}}$ *со стратегией вычисления с вызовом по имени обладает свойством подформульности: если* M *в нормальной форме, то всего его подтермы также в нормальной форме.*

Доказательство. Индукция по структуре M.

Случай $\mathbf{let\ pure}\,\vec{x} = \vec{M}\ \mathbf{in}\ N$ рассмотрен Какутани [19] [20].

Пусть $\mathbf{pure}\,M$ в нормальной форме, тогда M в нормальной форме и все его подтермы также в нормальной форме по предположению индукции.

Тогда, если $\mathbf{pure}\,M$ в нормальной форме, то и все его подтермы также в нормальной форме. □

4 Теоретико-категорная семантика системы типов $\lambda_{\mathbf{K}}$

4.1 Корректность

Теорема 5. *Корректность*

Пусть $\Gamma \vdash M : A$ *и* $M =_{\beta\eta} N$, *тогда* $[\![\Gamma \vdash M : A]\!] = [\![\Gamma \vdash N : A]\!]$

Доказательство.

Определение 22. *Семантическая трансляция из* $\lambda_{\mathbf{K}}$ *в аппликативный функтор* $\langle \mathcal{C}, \boxdot, \eta \rangle$ *над декартово замкнутой категорией* $\mathcal{C}$, *где* $\boxdot$ *– это моноидальный эндофунктор и* η *– это естественное преобразование* $Id_{\mathcal{C}} \Rightarrow \boxdot$*:*

- *Интерпретация типов:*
  - $[\![A]\!] := \hat{A}, A \in \mathbb{T}$, *где* $\hat{A}$ *– это объект категории* $\mathcal{C}$, *полученный в результате некоторого присваивания;*
  - $[\![A \to B]\!] := [\![B]\!]^{[\![A]\!]}$*;*
  - $[\![A \times B]\!] := [\![A]\!] \times [\![B]\!]$.
- *Интерпретация для модальных типов:*
  - $[\![\Box A]\!] = \boxdot[\![A]\!]$*;*
- *Интерпретация для контекстов:*
  - $[\![\quad]\!] = \mathbb{1}$, *где* $\mathbb{1}$ *– это терминальный объект в заданной декартово замкнутой категории;*
  - $[\![\Gamma, x : A]\!] = [\![\Gamma]\!] \times [\![A]\!]$
- *Интерпретация для типовых объявлений:*
  - $[\![\Gamma \vdash M : A]\!] := [\![M]\!] : [\![\Gamma]\!] \to [\![A]\!]$.
- *Интерпретация для правил типизации:*

$$\frac{}{[\![\Gamma, x : A \vdash x : A]\!] = \pi_2 : [\![\Gamma]\!] \times [\![A]\!] \to [\![A]\!]}$$

$$\frac{[\![\Gamma, x : A \vdash M : B]\!] = [\![M]\!] : [\![\Gamma]\!] \times [\![A]\!] \to [\![B]\!]}{[\![\Gamma \vdash (\lambda x.M) : A \to B]\!] = \Lambda([\![M]\!]) : [\![\Gamma]\!] \to [\![B]\!]^{[\![A]\!]}}$$

$$\frac{[\![\Gamma \vdash M : A \to B]\!] = [\![M]\!] : [\![\Gamma]\!] \to [\![B]\!]^{[\![A]\!]} \qquad [\![\Gamma \vdash N : A]\!] = [\![N]\!] : [\![\Gamma]\!] \to [\![A]\!]}{[\![\Gamma \vdash (MN) : B]\!] = [\![\Gamma]\!] \xrightarrow{\langle [\![M]\!], [\![N]\!] \rangle} [\![B]\!]^{[\![A]\!]} \times [\![A]\!] \xrightarrow{\epsilon} [\![B]\!]}$$

$$\frac{[\![\Gamma \vdash M : A]\!] = [\![M]\!] : [\![\Gamma]\!] \to [\![A]\!] \qquad [\![\Gamma \vdash N : B]\!] = [\![N]\!] : [\![\Gamma]\!] \to [\![B]\!]}{[\![\Gamma \vdash \langle M, N \rangle : A \times B]\!] = \langle [\![M]\!], [\![N]\!] \rangle : [\![\Gamma]\!] \to [\![A]\!] \times [\![B]\!]}$$

$$\frac{[\![\Gamma \vdash M : A_1 \times A_2]\!] = [\![M]\!] : [\![\Gamma]\!] \to [\![A_1]\!] \times [\![A_2]\!]}{[\![\Gamma \vdash \pi_i M : A_i]\!] = [\![\Gamma]\!] \xrightarrow{[\![M]\!]} [\![A_1]\!] \times [\![A_2]\!] \xrightarrow{\pi_i} [\![A_i]\!]} \; i \in \{1, 2\}$$

$$\frac{[\![\Gamma \vdash M : A]\!] = [\![M]\!] : [\![\Gamma]\!] \to [\![A]\!]}{[\![\Gamma \vdash \mathbf{pure}\, M : \Box A]\!] := [\![\Gamma]\!] \xrightarrow{[\![M]\!]} [\![A]\!] \xrightarrow{\eta_{[\![A]\!]}} \boxdot[\![A]\!]}$$

$$\frac{[\![\Gamma \vdash \vec{M} : \Box\vec{A}]\!] = \langle [\![M_1]\!], \dots, [\![M_n]\!]\rangle : [\![\Gamma]\!] \to \prod_{i=1}^{n} \boxdot[\![A_i]\!] \qquad [\![\vec{x} : \vec{A} \vdash N : B]\!] = [\![N]\!] : \prod_{i=1}^{n} [\![A_i]\!] \to [\![B]\!]}{[\![\Gamma \vdash \mathbf{let\ pure}\, \vec{x} = \vec{M}\ \mathbf{in}\ M : \Box B]\!] = \boxdot([\![N]\!]) \circ ([\![A_1]\!] * \dots * [\![A_n]\!]) \circ \langle [\![M_1]\!], \dots, [\![M_n]\!]\rangle : [\![\Gamma]\!] \to \boxdot[\![B]\!]}$$

Определение 23. *Одновременная подстановка*

Пусть $\Gamma = \{x_1 : A_1, ..., x_n : A_n\}$, $\Gamma \vdash M : A$ *и для любых* $i \in \{1, ..., n\}$, $\Gamma \vdash M_i : A_i$.

Одновременная подстановка $M[\vec{x} := \vec{M}]$ *определяется рекурсивно:*

- $x_i[\vec{x} := \vec{M}] = M_i$;
- $(\lambda x.M)[\vec{x} := \vec{M}] = \lambda x.(M[\vec{x} := \vec{M}])$;
- $(MN)[\vec{x} := \vec{M}] = (M[\vec{x} = \vec{M}])(N[\vec{x} := \vec{M}])$;
- $\langle M, N\rangle = \langle (M[\vec{x} = \vec{M}]), (N[\vec{x} := \vec{M}])\rangle$;
- $(\pi_i P)[\vec{x} := \vec{M}] = \pi_i(P[\vec{x} = \vec{M}])$;
- $(\mathbf{pure}\, M)[\vec{x} := \vec{M}] = \mathbf{pure}\,(M[\vec{x} = \vec{M}])$;
- $(\mathbf{let\ pure}\, \vec{x} = \vec{M}\ \mathbf{in}\ N)[\vec{y} := \vec{P}] = \mathbf{let\ pure}\, \vec{x} = (\vec{M}[\vec{y} := \vec{P}])\ \mathbf{in}\ N$

Лемма 8.

$[\![M[x_1 := M_1, \dots, x_n := M_n]]\!] = [\![M]\!] \circ \langle [\![M_1]\!], \dots, [\![M_n]\!]\rangle$.

Доказательство.

1)

$[\![\Gamma \vdash (\mathbf{pure}\, M)[\vec{x} := \vec{M}] : \Box A]\!] = [\![\Gamma \vdash \mathbf{pure}\,(M[\vec{x} := \vec{M}]) : \Box A]\!]$

Определение подстановки

$\eta_{[\![A]\!]} \circ [\![(M[\vec{x} := \vec{M}])]\!]$

интерпретация для **pure**

$\eta_{[\![A]\!]} \circ ([\![M]\!] \circ \langle [\![M_1]\!], \dots, [\![M_n]\!]\rangle) =$

предположение индукции∗

$(\eta_{[\![A]\!]} \circ [\![M]\!]) \circ \langle [\![M_1]\!], \dots, [\![M_n]\!]\rangle =$

ассоциативность композиции

$[\![\Gamma \vdash \mathbf{pure}\, M : \Box A]\!] \circ \langle [\![M_1]\!], \dots, [\![M_n]\!]\rangle =$

интерпретация для **pure**

2)

$$[\![\Gamma \vdash (\textbf{let pure}\, \vec{x} = \vec{M}\, \textbf{in}\, N)[\vec{y} := \vec{P}] : \Box B]\!] =$$

определение одновременной подстановки

$$[\![\Gamma \vdash \textbf{let pure}\, \vec{x} = (\vec{M}[\vec{y} := \vec{P}])\, \textbf{in}\, N : \Box B]\!] =$$

интерпретация $let_{\Box}$

$$\boxdot([\![N]\!]) \circ ([\![A_1]\!] * \cdots * [\![A_n]\!]) \circ [\![\Gamma \vdash (\vec{M}[\vec{y} := \vec{P}]) : \Box \vec{A}]\!] =$$

предположение индукции

$$\boxdot([\![N]\!]) \circ ([\![A_1]\!] * \cdots * [\![A_n]\!]) \circ ([\![\vec{M}]\!] \circ \langle [\![P_1]\!], \ldots, [\![P_n]\!] \rangle) =$$

ассоциативность композиции

$$(\boxdot([\![N]\!]) \circ ([\![A_1]\!] * \cdots * [\![A_n]\!]) \circ [\![\vec{M}]\!]) \circ \langle [\![P_1]\!], \ldots, [\![P_n]\!] \rangle =$$

по интерпретации

$$[\![\Gamma \vdash \textbf{let pure}\, \vec{x} = \vec{M}\, \textbf{in}\, N : \Box B]\!] \circ \langle [\![P_1]\!], \ldots, [\![P_n]\!] \rangle$$

□

Лемма 9.

Пусть $\Gamma \vdash M : A$ *и* $M \twoheadrightarrow_{\beta\eta} N$, *тогда* $[\![\Gamma \vdash M : A]\!] = [\![\Gamma \vdash N : A]\!]$*;*

Доказательство.

Случаи с правилом β-редукции для $let_{\Box}$ рассмотрены здесь [20]. Рассмотрим случаи с **pure**.

1) $[\![\Gamma \vdash \textbf{let pure}\, \vec{x} = \textbf{pure}\, \vec{M}\, \textbf{in}\, N : \Box B]\!] = [\![\Gamma \vdash \textbf{pure}\, N[\vec{x} := \vec{M}] : \Box B]\!]$

$$[\![\Gamma \vdash \textbf{let pure}\, \vec{x} = \textbf{pure}\, \vec{M}\, \textbf{in}\, N : \Box B]\!] =$$

интепретация

$$\boxdot([\![N]\!]) \circ ([\![A_1]\!] * \cdots * [\![A_n]\!]) \circ \langle \eta_{[\![A_1]\!]} \circ [\![M_1]\!], \ldots, \eta_{[\![A_n]\!]} \circ [\![M_n]\!] \rangle =$$

свойство пары морфизмов

$$\boxdot([\![N]\!]) \circ ([\![A_1]\!] * \cdots * [\![A_n]\!]) \circ (\eta_{[\![A_1]\!]} \times \cdots \times \eta_{[\![A_n]\!]}) \circ \langle [\![M_1]\!], \ldots, [\![M_n]\!] \rangle =$$

ассоциативность композиции

$$\boxdot([\![N]\!]) \circ (([\![A_1]\!] * \cdots * [\![A_n]\!]) \circ (\eta_{[\![A_1]\!]} \times \ldots \eta_{[\![A_n]\!]})) \circ \langle [\![M_1]\!], \ldots, [\![M_n]\!] \rangle =$$

по определению аппликативного функтора

$$\boxdot([\![N]\!]) \circ \eta_{[\![A_1]\!] \times \cdots \times [\![A_n]\!]} \circ \langle [\![M_1]\!], \ldots, [\![M_n]\!] \rangle =$$

естественность η

$$\eta_{[\![B]\!]} \circ [\![N]\!] \circ \langle [\![M_1]\!], \ldots, [\![M_n]\!] \rangle =$$

ассоциативность композиции

$$\eta_{[\![B]\!]} \circ ([\![N]\!] \circ \langle [\![M_1]\!], \ldots, [\![M_n]\!]) \rangle =$$

по лемме об одновременной подстановке

$$\eta_{[\![B]\!]} \circ [\![N[\vec{x} := \vec{M}]]\!]$$

интерпретация

$$[\![\Gamma \vdash \textbf{pure}\, (N[\vec{x} := \vec{M}]) : \Box B]\!]$$

2) $[\![\vdash \textbf{let pure}\, \_\_ = \_\_\, \textbf{in}\, M : \Box A]\!] = [\![\vdash \textbf{pure}\, M : \Box A]\!]$

$[\![\vdash \textbf{let pure} \_\_ = \_\_ \textbf{ in } M : \Box A]\!] =$

интерпретация

$\boxdot([\![M]\!]) \circ u_{\mathbb{1}} =$

определение аппликативного функтора

$\boxdot([\![M]\!]) \circ \eta_{\mathbb{1}} =$ $\Box$

естественность η

$\eta_{[\![A]\!]} \circ [\![M]\!] =$

интерпретация

$[\![\vdash \textbf{pure}\, M : \Box A]\!]$

$\Box$

4.2 Полнота

Теорема 6. *Полнота*

Пусть $[\![\Gamma \vdash M : A]\!] = [\![\Gamma \vdash N : A]\!]$, *тогда* $M =_{\beta\eta} N$.

Доказательство.

Мы будем работать с термовой моделью для простого типизированного λ-исчисления с $\times$ и $\to$, стандартно описанной здесь [22]:

Определение 24. *Эквивалетность на парах вида переменная-терм:*

Определим такое бинарное отношение $\sim_{A,B} \subseteq \mathbb{V} \times \Lambda_{\mathbf{K}}$, *что:*

$(x, M) \sim_{A,B} (y, N) \Leftrightarrow x : A \vdash M : B \ \& \ y : A \vdash N : A \ \& \ M =_{\beta\eta} N[y := x]$.

Нетрудно заметить, что данное отношение является отношением эквивалентности.

Обозначим класс эквивалентности как $[x, M]_{A,B} = \{(y, N) \mid (x, M) \sim_{A,B} (y, N)\}$ (ниже мы будем опускать индексы).

Определение 25. *Категория* $\mathcal{C}(\lambda)$*:*

- $Ob_{\mathcal{C}} = \{\hat{A} \mid A \in \mathbb{T}\} \cup \{\mathbb{1}\}$;
- $Hom_{\mathcal{C}(\lambda)}(\hat{A}, \hat{B}) = (\mathbb{V} \times \Lambda_{\mathbf{K}})/_{\sim_{A,B}}$;
- *Пусть* $[x, M] \in Hom_{\mathcal{C}(\lambda)}(\hat{A}, \hat{B})$ *и* $[y, N] \in Hom_{\mathcal{C}(\lambda)}(\hat{B}, \hat{C})$, *тогда* $[y, M] \circ [x, M] = [x, N[y := M]]$;
- *Тождественный морфизм* $id_{\hat{A}} = [x, x] \in Hom_{\mathcal{C}(\lambda)(\hat{A})}$;
- *Терминальный объект* $\mathbb{1}$;
- $\widehat{A \times B} = \hat{A} \times \hat{B}$;
- *Каноническая проекция:* $[x, \pi_i x] \in Hom_{\mathcal{C}(\lambda)}(\hat{A}_1 \times \hat{A}_2, \hat{A}_i)$ *for* $i \in \{1, 2\}$;
- $\widehat{A \to B} = \hat{B}^{\hat{A}}$;
- *Вычисляющая стрелка* $\epsilon = [x, (\pi_2 x)(\pi_1 x)] \in Hom_{\mathcal{C}(\lambda)(\hat{B}^{\hat{A}} \times \hat{A}, \hat{B})}$.

Определение 26. *Определим эндофунктор* $\boxdot : \mathcal{C}(\lambda) \to \mathcal{C}(\lambda)$ *таким образом, что для любых* $[x, M] \in Hom_{\mathcal{C}(\lambda)}(\hat{A}, \hat{B}), \boxdot([x, M]) = [y, \textbf{let pure}\, x = y \textbf{ in } M] \in Hom_{\mathcal{C}(\lambda)}(\boxdot\hat{A}, \boxdot\hat{B})$ *(обозначения: fmap* f *для произвольной стрелки* f*).*

Достаточно показать, что $\boxdot$ – это аппликативный функтор над $\mathcal{C}(\lambda)$.

Лемма 10. *Функториальность*

- *fmap* $(g \circ f) =$ *fmap* $(g) \circ$ *fmap* (f);
- *fmap* $(id_{\hat{A}}) = id_{\boxdot \hat{A}}$.

Доказательство.

1)

$\text{fmap }(g \circ f) = \text{fmap}([y, N] \circ [x, M]) =$

По определению композиции

$\text{fmap }([x, N[y := M]]) =$

По определению fmap

$[z, \textbf{let pure}\, x = z \,\textbf{in}\, N[y := M]]$

$\text{fmap }(g) \circ \text{fmap }(f) = \text{fmap }([y, N]) \circ \text{fmap }([x, M]) =$

По определению fmap

$[y_1, \textbf{let pure}\, y = y_1 \,\textbf{in}\, N] \circ [z, \textbf{let pure}\, x = z \,\textbf{in}\, M] =$

По определению композиции

$[z, \textbf{let pure}\, y = y_1 \,\textbf{in}\, N[y_1 := \textbf{let pure}\, x = z \,\textbf{in}\, M]] =$

Подстановка

$[z, \textbf{let pure}\, y = (\textbf{let pure}\, x = z \,\textbf{in}\, M) \,\textbf{in}\, N] =$

Правило $\beta\Box$

$[z, \textbf{let pure}\, x = z \,\textbf{in}\, N[y := M]]$

2)

$\text{fmap }(id_{\hat{A}}) =$

Определение тождественного морфизма

$\text{fmap }[x, x] =$

По определению fmap

$[z, \textbf{let pure}\, x = z \,\textbf{in}\, x]$

Правило $\Box id$

$[z, z] = id_{\boxdot \hat{A}}$

$\Box$

Определение 27. *Определим естественные преобразования:*

- $\eta : Id \Rightarrow \boxdot$, *такое, что* $\forall \hat{A} \in Ob_{\mathcal{C}(\lambda)}$, $\eta_{\hat{A}} = [x, \textbf{pure}\, x] \in Hom_{\mathcal{C}(\lambda)}(\hat{A}, \boxdot \hat{A})$;
- $*_{A,B} : \boxdot \hat{A} \times \boxdot \hat{B} \to \boxdot(\hat{A} \times \hat{B})$, *такое, что* $\forall \hat{A}, \hat{B} \in Ob_{\mathcal{C}(\lambda)}, *_{\hat{A},\hat{B}} = [p, \textbf{let pure}\, x, y = \pi_1 p, \pi_2 p \,\textbf{in}\, \langle x, y \rangle] \in Hom_{\mathcal{C}(\lambda)}(\boxdot A \times \boxdot B, \boxdot(A \times B))$.

Реализация $*$ в нашей термовой модели – это частный случай правила $\text{let}_{\Box}$:

$$\dfrac{\dfrac{p : \Box A \times \Box B \vdash p : \Box A \times \Box B}{p : \Box A \times \Box B \vdash \pi_1 p : \Box A} \quad \dfrac{p : \Box A \times \Box B \vdash p : \Box A \times \Box B}{p : \Box A \times \Box B \vdash \pi_2 p : \Box B} \quad \dfrac{x : A \vdash x : A \qquad y : B \vdash y : B}{x : A, y : B \vdash \langle x, y \rangle : A \times B}}{p : \Box A \times \Box B \vdash \textbf{let pure}\, x, y = \pi_1 p, \pi_2 p \,\textbf{in}\, \langle x, y \rangle : \Box(A \times B)}$$

Лемма 11.

$\boxed{\cdot}$ – *моноидальный эндофунктор.*

Доказательство.

Показывается аналогично [19]. $\square$

Лемма 12. *Естественность и когерентность η:*

- $fmap\ f \circ \eta_A = \eta_B \circ f$;
- $*_{\hat{A},\hat{B}} \circ (\eta_A \times \eta_B) = \eta_{\hat{A}\times\hat{B}}$;

Доказательство.

i) $\text{fmap } f \circ \eta_{\hat{A}} = \eta_{\hat{B}} \circ f$

$\eta_{\hat{B}} \circ f =$

по определению

$[y, \mathbf{pure}\, y] \circ [x, M] =$

композиция

$[x, \mathbf{pure}\, y[y := M]] =$

подстановка

$[x, \mathbf{pure}\, M]$

С другой стороны:

$\text{fmap } f \circ \eta_{\hat{A}} =$

по определению

$[z, \mathbf{let\ pure}\, x = z\ \mathbf{in}\ M] \circ [x, \mathbf{pure\ x}] =$

композиция

$[x, \mathbf{let\ pure}\, x = z\ \mathbf{in}\ M[z := \mathbf{pure}\, x]] =$

подстановка

$[x, \mathbf{let\ pure}\, x = \mathbf{pure}\, x\ \mathbf{in}\ M] =$

правило β-редукции

$[x, \mathbf{pure}\, M[x := x]] =$

постановка

$[x, \mathbf{pure}\, M]$

ii) $*_{\hat{A},\hat{B}} \circ (\eta_{\hat{A}} \times \eta_{\hat{B}}) = \eta_{\hat{A}\times\hat{B}}$

$*_{\hat{A},\hat{B}} \circ (\eta_{\hat{A}} \times \eta_{\hat{B}}) =$

раскрытие

$[q, \mathbf{let\ pure}\, x, y = \pi_1 q, \pi_2 q\ \mathbf{in}\ \langle x, y\rangle] \circ [p, \langle \mathbf{pure}\,(\pi_1 p), \mathbf{pure}\,(\pi_2 p)\rangle] =$

композиция

$[p, \mathbf{let\ pure}\, x, y = \pi_1 q, \pi_2 q\ \mathbf{in}\ \langle x, y\rangle[q := \langle \mathbf{pure}\,(\pi_1 p), \mathbf{pure}\,(\pi_2 p)\rangle]] =$

подстановка

$[p, \mathbf{let\ pure}\, x, y = \pi_1(\langle \mathbf{pure}\,(\pi_1 p), \mathbf{pure}\,(\pi_2 p)\rangle), \pi_2(\langle \mathbf{pure}\,(\pi_1 p), \mathbf{pure}\,(\pi_2 p)\rangle)\ \mathbf{in}\ \langle x, y\rangle] =$

правило β-редукции

$[p, \mathbf{let\ pure}\, x, y = \mathbf{pure}\,(\pi_1 p), \mathbf{pure}\,(\pi_2 p)\ \mathbf{in}\ \langle x, y\rangle] =$

правило β-редукции

$[p, \mathbf{pure}\,(\langle x, y\rangle[x := \pi_1 p, y := \pi_2 p])] =$

подстановка

$[p, \mathbf{pure}\,\langle \pi_1 p, \pi_2 p\rangle] =$

правило η-редукции

$[p, \mathbf{pure}\, p] =$

определение

$\eta_{\hat{A}\times\hat{B}}$

□

Определение 28.

$u_{\mathbb{1}} = [\blacksquare, \mathbf{let\ pure}\, \_\_ = \_\_\ \mathbf{in}\ \blacksquare] \in Hom_{\mathcal{C}(\lambda)}(\mathbb{1}, \boxdot\mathbb{1})$.

Лемма 13.

$u_{\mathbb{1}} = \eta_{\mathbb{1}}$

Доказательство. Следует напрямую из правила $\beta\mathbf{nec}$. □

Лемма 14. $\boxdot$ *– это аппликативный функтор.*

Доказательство. Непосредственно следует из предыдущих лемм. □

Аналогично [24], мы применяем трансляцию из $\lambda_{\mathbf{K}}$ к произвольной декартово замкнутой категории с аппликативным функтором $\boxdot$, тогда мы имеем $[\![\Gamma \vdash M : A]\!] = [x, M[x_i := \pi_i x]]$, so $M =_{\beta\eta} N \Leftrightarrow [\![\Gamma \vdash M : A]\!] = [\![\Gamma \vdash N : A]\!]$.

□

Список литературы

[1] Artemov S. and Protopopescu T., "Intuitionistic Epistemic Logic", *The Review of Symbolic Logic*, 2016, vol. 9, no 2. pp. 266-298.

[2] Krupski V. N. and Yatmanov A., "Sequent Calculus for Intuitionistic Epistemic Logic IEL", *Logical Foundations of Computer Science: International Symposium, LFCS 2016, Deerfield Beach, FL, USA, January 4-7, 2016. Proceedings*, 2016, pp. 187-201.

[3] Haskell Language. // URL: https://www.haskell.org. (Date: 1.08.2017)

[4] Idris. A Language with Dependent Types.// URL:https://www.idris-lang.org. (Date: 1.08.2017)

[5] Purescript. A strongly-typed functional programming language that compiles to JavaScript. URL: http://www.purescript.org. (Date: 1.08.2017)

[6] Elm. A delightful language for reliable webapps. // URL: http://elm-lang.org. (Date: 1.08.2017)

[7] Hackage, "The base package" // URL: https://hackage.haskell.org/package/base-4.10.0.0 (Date: 1.08.2017)

[8] Lipovaca M, "Learn you a Haskell for Great Good!". //URL: http://learnyouahaskell.com/chapters (Date: 1.08.2017)

[9] McBride C. and Paterson R., "Applicative programming with effects *Journal of Functional Programming*, 2008, vol. 18, no 01. pp 1-13.

[10] McBride C. and Paterson R, "Functional Pearl. Idioms: applicative programming with effects", *Journal of Functional Programming*, 2005. vol. 18, no 01. pp 1-20.

[11] R. Nederpelt and H. Geuvers, "Type Theory and Formal Proof: An Introduction". *Cambridge University Press*, New York, NY, USA, 2014. pp. 436.

[12] Sorensen M. H. and Urzyczyn P, "Lectures on the Curry-Howard isomorphism", *Studies in Logic and the Foundations of Mathematics*, vol. 149, *Elsevier Science*, 1998. pp 261.

[13] Pierce B. C., "Types and Programming Languages". *Cambridge, Mass: The MIT Press*, 2002. pp. 605.

[14] Girard J.-Y., Taylor P. and Lafont Y, "Proofs and Types", *Cambridge University Press*, New York, NY, USA, 1989. pp. 175.

[15] Barendregt. H. P., "Lambda calculi with types"// Abramsky S., Gabbay Dov M., and S. E. Maibaum, "Handbook of logic in computer science (vol. 2), Osborne Handbooks Of Logic In Computer Science", Vol. 2. *Oxford University Press, Inc.*, New York, NY, USA, 1993. pp 117-309.

[16] Hindley J. Roger, "Basic Simple Type Theory". *Cambridge University Press*, New York, NY, USA, 1997. pp. 185.

[17] Pfenning F. and Davies R., "A judgmental reconstruction of modal logic", *Mathematical Structures in Computer Science*, vol. 11, no 4, 2001, pp. 511-540.

[18] H.P. Barendregt. The Lambda Calculus — Its Syntax and Semantics. Studies in Logic and the Foundations of Mathematics, vol. 103. Amsterdam: North-Holland, 1985.

[19] Yoshihiko KAKUTANI, A Curry-Howard Correspondence for Intuitionistic Normal Modal Logic, Computer Software, Released February 29, 2008, Online ISSN , Print ISSN 0289-6540.

[20] Kakutani Y. (2007) Call-by-Name and Call-by-Value in Normal Modal Logic. In: Shao Z. (eds) Programming Languages and Systems. APLAS 2007. Lecture Notes in Computer Science, vol 4807. Springer, Berlin, Heidelberg

[21] T. Abe. Completeness of modal proofs in

first-order predicate logic. Computer Software, JSSST Journal, 24:165 – 177, 2007.

[22] Lambek, J. and Scott P.J. (1986) Introduction to Higher Order Categorical Logic, Cambridge Studies in Advanced Mathematics 7, Cambridge: Cambridge University Press.

[23] Samuel Eilenberg and Max Kelly, Closed categories. Proc. Conf. Categorical Algebra (La Jolla, Calif., 1965).

[24] Samson Abramsky and Nikos Tzevelekos, Introduction to Categories and Categorical Logic

[25] G. A. Kavvos. The Many Worlds of Modal Λ–calculi: I. Curry-Howard for Necessity, Possibility and Time

[26] Ross Paterson. in Mathematics of Program Construction, Madrid, 2012, Lecture Notes in Computer Science, vol. 7342, pp. 300–323, Springer, 2012.