Nominal logic for reasoning about terms with variable bindings

(Logika dziedzinowa do wnioskowania
o termach z wiązaniem zmiennych)

Dominik Gulczyński

Praca magisterska

Promotor: dr Piotr Polesiuk

Uniwersytet Wrocławski
Wydział Matematyki i Informatyki
Instytut Informatyki

18 lipca 2023

Abstract

We describe logic for reasoning about terms with variable bindings.

Streszczenie

Przedstawiamy logikę dziedzinową do wnioskowania o termach z wiązaniem zmiennych.

Contents

Chapter 1

Introduction

1.1 Problem statement

...

1.2 Motivation

...

1.3 Related work

1.3.1 Nominal logics & permutations

1.4 Contributions

...

Chapter 2

Terms and constraints

In classical first-order logic, terms are built from variables and applications of functional symbols to other terms. In this work we expand terms with expressions closely resembling the syntax of lambda calculus, aiming to provide a flexible framework for reasoning about the lambda calculus and its derivations.

To this end we introduce an infinite set of *atoms* (denoted by lower-case letters), representing the bound variables in terms — i.e. the variables in the sense of lambda calculus. That set is disjount with the set of variables commonly found in first-order logic, which from now on we will call *variables* (and denote by uppper-case letters) as apposed to *atoms*.

Terms are given by the following grammar:

$$\begin{array}{lll} \pi & ::= & \mathsf{id} \mid (\alpha\ \alpha)\pi \qquad \text{(permutations)} \\ \alpha & ::= & \pi\ a \qquad \text{(atom expressions)} \\ t & ::= & \alpha \mid \pi\ X \mid \alpha.t \mid t\ t \mid s \qquad \text{(terms)} \end{array}$$

It is important to note that terms do not incorporate any inherent notions of computation, reduction, or binding. These expressions simply *look* like the lambda calculus but they lack the operational semantics of it. However, the intuitions associated with such expressions are not unfounded. We will observe their practical application in the sublogic of constraints that we define on top of terms to reason about notions of *freshness*, *variable binding* and *structural* order and its logical model.

Constraints are given by the following grammar:

$$c \quad ::= \quad \alpha \# t \mid t = t \mid t \sim t \mid t \prec t \qquad \text{(constraints)}$$

with following semantics:

$\alpha \# t$	—	atom α is Fresh in term t, i.e. does not occur in t as a free variable
$t_1 = t_2$	—	terms t_1 and t_2 are alpha-equivalent
$t_1 \sim t_2$	—	terms t_1 and t_2 possess an identical shape, i.e. after erasing all atoms, terms t_1 and t_2 would be equal
$t_1 \prec t_2$	—	shape of term t_1 is structurally smaller than the shape of term t_2, i.e. after erasing all atoms t_1 would be equal to some subterm of t_2

We use metavariable Γ for finite sets of constraints.

$$\begin{array}{lcll} T & ::= & A \mid n \mid \$T \mid T@T \mid s & \text{(semantic terms)} \\ S & ::= & \_ \mid \_.S \mid S@S \mid s & \text{(semantic shapes)} \end{array}$$

$$\begin{array}{rcl} [\![\pi\ a]\!]_\rho & = & [\![\pi]\!]_\rho(\rho(a)) \\ [\![\pi\ X]\!]_\rho & = & [\![\pi]\!]_\rho(\rho(X)) \\ [\![\alpha.t]\!]_\rho & = & \$([\![t]\!]_\rho\uparrow)\{[\![\alpha]\!]_\rho \mapsto 0\} \\ [\![t_1\ t_2]\!]_\rho & = & [\![t_1]\!]_\rho @ [\![t_2]\!]_\rho \\ [\![s]\!]_\rho & = & s \end{array}$$

$$\begin{array}{rcl} |A| & = & \_ \\ |n| & = & \_ \\ |\$T| & = & \_.|T| \\ 1|T_1@T_2| & = & |T_1|@|T_2| \end{array}$$

$$\begin{array}{rll} \rho \vDash t_1 = t_2 & \text{iff} & [\![t_1]\!]_\rho = [\![t_2]\!]_\rho \\ \rho \vDash \alpha \# t & \text{iff} & [\![\alpha]\!]_\rho \notin \mathsf{FreeAtoms}([\![t]\!]_\rho) \\ \rho \vDash t_1 \sim t_2 & \text{iff} & |[\![t_1]\!]_\rho| = |[\![t_2]\!]_\rho| \\ \rho \vDash t_1 \prec t_2 & \text{iff} & |[\![t_1]\!]_\rho| \text{ is a strict subshape of } |[\![t_2]\!]_\rho| \end{array}$$

We write $\rho \vDash \Gamma$ iff for all $c \in \Gamma$ we have $\rho \vDash c$. We write $\Gamma \vDash c$ iff for every ρ such that $\rho \vDash \Gamma$ we have $\rho \vDash c$.

With this model in mind we will se that there exists a decidibile algorithm for determining whether C1,...,Cn —= C0, i.e. a deterministic way of checking if constraints c1, ..., cn imply c0. We present such algorithm in the next chapter.

Chapter 3

Constraint solver

Bird's eye view: Solver breaks down constraints (on both sides of the turnstile) to irreducible components that are solved easily.

At the core of our work lies the Solver — the algorithm of resolving the constraints. Given a list of assumptions $c_1, \ldots, c_n$ it checks whether given goal c_0 holds. In other words it is an algorithm that verifies whether, for every possible substitution of closed terms (in terms of variables, not atoms) for variables in $c_0, c_1, \ldots, c_n$ such that the constraints $c_1, \ldots, c_n$ are satisfied, c_0 is also satisfied.

For convenience and effectiveness of implementation, the Solver works with constraints a little different constraints (although not more expressive) than those occuring in formulas and kinds, main difference being use of *shapes* instead of terms for shape constraints. Solver contraints and shapes are given by the following grammar:

$$\begin{array}{rcll} \mathcal{C} & ::= & \alpha \mathrel{\#} t \mid t = t \mid S \sim S \mid S \prec S & \text{(solver constraints)} \\ S & ::= & \_ \mid X \mid \_.S \mid S\ S \mid s & \text{(shapes)} \end{array}$$

Solver erases atoms from terms in shape contstraints, effectively transforming them from *constraints* to *solver constraints*.

We add another environment Δ to distinguish between the potentially-reducible assumptions in Γ. For convenience we will write $a \neq \alpha$ instead of $a \mathrel{\#} \alpha$ as it gives good intuition of atom freshness implying inequality and for $\alpha = \pi a$ we will write $\alpha \mathrel{\#} t$ meaning $a \mathrel{\#} \pi^{-1} t$. Irreducible constraints are:

$a_1 \neq a_2$	—	atoms a_1 and a_2 are different
$a \mathrel{\#} X$	—	atom a is Fresh in variable X
$X_1 \sim X_2$	—	variables X_1 and X_2 posses the same shape
$X \sim t$	—	variable X has a shape of term t
$t \prec X$	—	term t strictly subshapes variable X

After all the constraints are reduced to such simple constraints we reduce the goal-constraint and repeat the reduction procedure on new assumptions and goal. We either arrive on a contradictory environment or all the assumptions and goal itself are reduced to irreducible constraints which is as simple as checking if the goal occurs on the left side of the turnstile.

$$\frac{\mathcal{C} \in \Delta}{\Gamma; \Delta \vDash \mathcal{C}}$$

Decidability of atom equality plays an important role in the reduce procedure:

$$\frac{\Gamma; \Delta \vDash a = \pi^{-1}\alpha}{\Gamma; \Delta \vDash \pi a = \alpha} \qquad \frac{\begin{array}{c} a \neq \alpha_1, a \neq \alpha_2, \Gamma; \Delta \vDash a = \alpha \\ a = \alpha_1, a \neq \alpha_2, \Gamma; \Delta \vDash \alpha_2 = \alpha \qquad a = \alpha_2, \Gamma; \Delta \vDash \alpha_1 = \alpha \end{array}}{\Gamma; \Delta \vDash a = (\alpha_1\ \alpha_2)\alpha}$$

$$\frac{\Gamma; \Delta \vDash \pi \text{ idempotent on } X}{\Gamma; \Delta \vDash X = \pi X} \qquad \frac{\Gamma; \Delta \vDash X_1 = \pi_1^{-1}\pi_2 X_2}{\Gamma; \Delta \vDash \pi_1 X_1 = \pi_2 X_2}$$

$$\frac{\Gamma; \Delta \vDash \alpha_1 \# t_2 \qquad \Gamma; \Delta \vDash t_1 = (\alpha_1\ \alpha_2)t_2}{\Gamma; \Delta \vDash \alpha_1.t_1 = \alpha_2.t_2} \qquad \frac{\Gamma; \Delta \vDash t_1 = t_2 \qquad \Gamma; \Delta \vDash t_1' = t_2'}{\Gamma; \Delta \vDash t_1 t_1' = t_2 t_2'}$$

$$\frac{}{\Gamma; \Delta \vDash a = a} \qquad \frac{}{\Gamma; \Delta \vDash X = X} \qquad \frac{}{\Gamma; \Delta \vDash s = s}$$

$$\frac{\forall a \in \pi.\ \Gamma; \Delta \vDash a = \pi a\ \vee\ \Gamma; \Delta \vDash a \# X}{\Gamma; \Delta \vDash \pi \text{ idempotent on } X}$$

$$\frac{a_1 \neq a_2 \in \Delta}{\Gamma; \Delta \vDash a_1 \# a_2} \qquad \frac{\begin{array}{c} a \neq \alpha_1, a \neq \alpha_2, \Gamma; \Delta \vDash a \# \alpha \\ a = \alpha_1, a \neq \alpha_2, \Gamma; \Delta \vDash \alpha_1 \# \alpha \qquad a = \alpha_2, \Gamma; \Delta \vDash \alpha_2 \# \alpha \end{array}}{\Gamma; \Delta \vDash a \# (\alpha_1\ \alpha_2)\alpha}$$

$$\frac{a \# X \in \Delta}{\Gamma; \Delta \vDash a \# X} \qquad \frac{\begin{array}{c} a \neq \alpha_1, a \neq \alpha_2, \Gamma; \Delta \vDash a \# \pi X \\ a = \alpha_1, a \neq \alpha_2, \Gamma; \Delta \vDash \alpha_1 \# \pi X \qquad a = \alpha_2, \Gamma; \Delta \vDash \alpha_2 \# \pi X \end{array}}{\Gamma; \Delta \vDash a \# (\alpha_1\ \alpha_2)\pi X}$$

$$\frac{a \neq \alpha, \Gamma; \Delta \vDash a \# t}{\Gamma; \Delta \vDash a \# \alpha.t} \qquad \frac{\Gamma; \Delta \vDash a \# t_1 \qquad \Gamma; \Delta \vDash a \# t_2}{\Gamma; \Delta \vDash a \# t_1 t_2} \qquad \frac{}{\Gamma; \Delta \vDash a \# s}$$

$$\frac{X_1 \sim X_2 \in \Delta}{\Gamma; \Delta \vDash X_1 \sim X_2} \qquad \frac{X \sim S' \in \Delta \qquad \Gamma; \Delta \vDash S' \sim S}{\Gamma; \Delta \vDash X \sim S}$$

$$\frac{\Gamma; \Delta \vDash S_1 \sim S_2}{\Gamma; \Delta \vDash \_.S_1 \sim \_.S_2} \qquad \frac{\Gamma; \Delta \vDash S_1 \sim S_2 \qquad \Gamma; \Delta \vDash S_1' \sim S_2'}{\Gamma; \Delta \vDash S_1 S_1' \sim S_2 S_2'} \qquad \frac{}{\Gamma; \Delta \vDash s \sim s}$$

$$\frac{S_2 \prec X \in \Delta \qquad \Gamma; \Delta \vDash S_2 \sim X}{\Gamma; \Delta \vDash S_1 \prec X} \qquad \frac{S_2 \prec X \in \Delta \qquad \Gamma; \Delta \vDash S_2 \prec X}{\Gamma; \Delta \vDash S_1 \prec X}$$

$$\frac{\Gamma; \Delta \vDash S_1 \sim S_2}{\Gamma; \Delta \vDash S_1 \prec \_.S_2} \qquad \frac{\Gamma; \Delta \vDash S_1 \prec S_2}{\Gamma; \Delta \vDash S_1 \prec \_.S_2}$$

$$\frac{\Gamma;\Delta \vDash S_1 \sim S_2}{\Gamma;\Delta \vDash S_1 \prec S_2S_2'} \qquad \frac{\Gamma;\Delta \vDash S_1 \sim S_2'}{\Gamma;\Delta \vDash S_1 \prec S_2S_2'} \qquad \frac{\Gamma;\Delta \vDash S_1 \prec S_2}{\Gamma;\Delta \vDash S_1 \prec S_2S_2'} \qquad \frac{\Gamma;\Delta \vDash S_1 \prec S_2'}{\Gamma;\Delta \vDash S_1 \prec S_2S_2'}$$

$$\frac{a_1 \neq a_2 \in \Delta}{a_1 = a_2, \Gamma;\Delta \vDash \mathcal{C}} \qquad \frac{\Gamma\{a_1 \mapsto a_2\};\Delta\{a_1 \mapsto a_2\} \vDash \mathcal{C}\{a_1 \mapsto a_2\}}{a_1 = a_2, \Gamma;\Delta \vDash \mathcal{C}}$$

$$\frac{\begin{array}{c} a \neq \alpha_1, a \neq \alpha_2, a = \alpha, \Gamma;\Delta \vDash \mathcal{C} \\ a = \alpha_1, a \neq \alpha_2, \alpha_2 = \alpha, \Gamma;\Delta \vDash \mathcal{C} \qquad a = \alpha_2, \alpha_1 = \alpha, \Gamma;\Delta \vDash \mathcal{C} \end{array}}{a = (\alpha_1\ \alpha_1)\alpha, \Gamma;\Delta \vDash \mathcal{C}}$$

$$\frac{a = \pi^{-1}\alpha, \Gamma;\Delta \vDash \mathcal{C}}{\pi a = \alpha, \Gamma;\Delta \vDash \mathcal{C}} \qquad \frac{}{a = t_1t_2, \Gamma;\Delta \vDash \mathcal{C}} \qquad \frac{}{a = \alpha.t, \Gamma;\Delta \vDash \mathcal{C}} \qquad \frac{}{a = s, \Gamma;\Delta \vDash \mathcal{C}}$$

$$\frac{\vDash \text{ idempotent on } X}{X = \pi X, \Gamma;\Delta \vDash \mathcal{C}} \qquad \frac{\pi \text{ idempotent on } X, \Gamma;\Delta \vDash \mathcal{C}}{X = \pi X, \Gamma;\Delta \vDash \mathcal{C}}$$

$$\frac{\Gamma\{X \mapsto t\};\Delta\{X \mapsto t\} \vDash \mathcal{C}\{X \mapsto t\}}{X = t, \Gamma;\Delta \vDash \mathcal{C}} \qquad \frac{X = \pi^{-1}t, \Gamma;\Delta \vDash \mathcal{C}}{\pi X = t, \Gamma;\Delta \vDash \mathcal{C}}$$

$$\frac{\alpha_1 \# \alpha_2.t_2,\ t_1 = (\alpha_1\ \alpha_2)t_2,\ \Gamma;\Delta \vDash \mathcal{C}}{\alpha_1.t_1 = \alpha_2.t_2, \Gamma;\Delta \vDash \mathcal{C}} \qquad \text{Other term constructors trivial}$$

$$\frac{t_1 = t_2,\ t_1' = t_2',\ \Gamma;\Delta \vDash \mathcal{C}}{t_1t_1' = t_2t_2', \Gamma;\Delta \vDash \mathcal{C}} \qquad \text{Other term constructors trivial}$$

$$\frac{s_1 \neq s_2}{s_1 = s_2, \Gamma;\Delta \vDash \mathcal{C}} \qquad \frac{}{s = s, \Gamma;\Delta \vDash \mathcal{C}} \qquad \text{Other term constructors trivial}$$

$$\frac{(\forall a \in \pi.\ \Gamma;\Delta \vDash a = \pi a\ \vee\ \Gamma;\Delta \vDash a \# X), \Gamma;\Delta \vDash \mathcal{C}}{\pi \text{ idempotent on } X, \Gamma;\Delta \vDash \mathcal{C}}$$

$$\frac{}{a \neq a, \Gamma;\Delta \vDash \mathcal{C}} \qquad \frac{\Gamma;\{a_1 \neq a_2\} \cup \Delta \vDash \mathcal{C}}{a_1 \neq a_2,\ \Gamma;\Delta \vDash \mathcal{C}} \qquad \frac{\Gamma;\{a \# X\} \cup \Delta \vDash \mathcal{C}}{a \# X,\ \Gamma;\Delta \vDash \mathcal{C}}$$

$$\frac{\begin{array}{c} a \neq \alpha_1, a \neq \alpha_2, a \# \alpha, \Gamma;\Delta \vDash \mathcal{C} \\ a = \alpha_1, a \neq \alpha_2, \alpha_2 \# \alpha, \Gamma;\Delta \vDash \mathcal{C} \qquad a = \alpha_2, \alpha_1 \# \alpha, \Gamma;\Delta \vDash \mathcal{C} \end{array}}{a \# (\alpha_1\ \alpha_1)\alpha, \Gamma;\Delta \vDash \mathcal{C}}$$

$$\frac{\begin{array}{c} a \neq \alpha_1, a \neq \alpha_2, a \# X, \Gamma;\Delta \vDash \mathcal{C} \\ a = \alpha_1, a \neq \alpha_2, \alpha_2 \# X, \Gamma;\Delta \vDash \mathcal{C} \qquad a = \alpha_2, \alpha_1 \# X, \Gamma;\Delta \vDash \mathcal{C} \end{array}}{a \# (\alpha_1\ \alpha_1)X, \Gamma;\Delta \vDash \mathcal{C}}$$

$$\frac{a \# \alpha,\ \Gamma;\Delta \vDash \mathcal{C} \qquad a \# \alpha,\ a \# t,\ \Gamma;\Delta \vDash \mathcal{C}}{a \# \alpha.t, \Gamma;\Delta \vDash \mathcal{C}}$$

$$\frac{a \# t_1, \Gamma;\Delta \vDash \mathcal{C} \qquad a \# t_2, \Gamma;\Delta \vDash \mathcal{C}}{a \# t_1t_2, \Gamma;\Delta \vDash \mathcal{C}}$$

$$\frac{\Gamma;\Delta\vDash\mathcal{C}}{a\,\#\,s,\Gamma;\Delta\vDash\mathcal{C}}$$

$$\frac{\Gamma;\{X_1\sim X_2\}\cup\Delta\vDash\mathcal{C}}{X_1\sim X_2,\Gamma;\Delta\vDash\mathcal{C}}\qquad\frac{\Gamma;\{X\sim S\}\cup\Delta\vDash\mathcal{C}}{X\sim S,\ \Gamma;\Delta\vDash\mathcal{C}}$$

$$\frac{\Gamma;\Delta\vDash\mathcal{C}}{a_1\sim a_2,\Gamma;\Delta\vDash\mathcal{C}}\quad\text{Other term constructors trivial}$$

$$\frac{t_1\sim t_2,\Gamma;\Delta\vDash\mathcal{C}}{\_.t_1\sim\_.t_2,\Gamma;\Delta\vDash\mathcal{C}}\quad\text{Other term constructors trivial}$$

$$\frac{t_1\sim t_2,\Gamma;\Delta\vDash\mathcal{C}\qquad t_1'\sim t_2',\Gamma;\Delta\vDash\mathcal{C}}{t_1t_1'\sim t_2t_2',\Gamma;\Delta\vDash\mathcal{C}}\quad\text{Other term constructors trivial}$$

$$\frac{s_1\neq s_2}{s_1\sim s_2,\Gamma;\Delta\vDash\mathcal{C}}\qquad\frac{}{s\sim s,\Gamma;\Delta\vDash\mathcal{C}}\quad\text{Other term constructors trivial}$$

$$\frac{\Gamma;\{t\prec X\}\cup\Delta\vDash\mathcal{C}}{t\prec X,\Gamma;\Delta\vDash\mathcal{C}}$$

$$\frac{t_1\sim t_2,\Gamma;\Delta\vDash\mathcal{C}\qquad t_1\prec t_2,\Gamma;\Delta\vDash\mathcal{C}}{t_1\prec\_.t_2,\Gamma;\Delta\vDash\mathcal{C}}$$

$$\frac{t_1\sim t_2,\Gamma;\Delta\vDash\mathcal{C}\qquad t_1\sim t_2',\Gamma;\Delta\vDash\mathcal{C}\qquad t_1\prec t_2,\Gamma;\Delta\vDash\mathcal{C}\qquad t_1\prec t_2',\Gamma;\Delta\vDash\mathcal{C}}{t_1\prec t_2t_2',\Gamma;\Delta\vDash\mathcal{C}}$$

$$\frac{}{t\prec\alpha,\Gamma;\Delta\vDash\mathcal{C}}\qquad\frac{}{t\prec s,\Gamma;\Delta\vDash\mathcal{C}}$$

TODO: explain what is $\{\mathcal{C}\}\cup\Delta$ Additional rule for ariving in contradictory Δ:

$$\frac{}{\Gamma;\lightning\vDash\mathcal{C}}$$

Define state of the solver by triple $(\Gamma,\Delta,\mathcal{C}_0)$ and such ordering of the states:

1. Number of distinct variables in Γ, Δ,$\mathcal{C}_0$.
2. Depth of $\mathcal{C}_0$.
3. Number of assumptions of given depth in Γ and Δ.
4. Number of assumptions of given depth in Γ.

Then by analysing each rule we can see the reductions always arrive in a smaller state.

3.1 Implementation

Environment Δ is a quintuple $(NeqAtoms_\Delta, Fresh_\Delta, VarShape_\Delta, Shape_\Delta, Subshape_\Delta)$ where:
NeqAtoms is a set of pairs of atoms that we know are different,
$Fresh$ is a mapping from atoms to variables that we know the atom is Fresh in,
$VarShape$ is a mapping from variables to shape-representative variables (i.e. all variables that are mapped in $VarShape$ to the same variable are of the same shape),
Shape is a mapping from shape-representative variables to shape that we know this variable must have,
SubShape is a mapping from shape-representative variables to sets of shapes that we know this variable must supershape.

We can now define a way to compute the shape-representative variable:

$$X_\Delta := \begin{cases} X & \text{if } VarShape_\Delta(X) = \emptyset \\ X'_\Delta & \text{if } VarShape_\Delta(X) = X' \end{cases}$$

And shape-reconstruction:

$$\begin{aligned} |X|_\Delta &:= \begin{cases} |X'|_\Delta & \text{if } VarShape_\Delta(X) = X' \\ S & \text{if } Shape_\Delta(X) = S \\ X & \text{otherwise} \end{cases} \\ |\_|_\Delta &:= \_ \\ |\_.S|_\Delta &:= \_.|S|_\Delta \\ |S_1 S_2|_\Delta &:= |S_1|_\Delta |S_2|_\Delta \\ |s|_\Delta &:= s \\ |t|_\Delta &:= ||t||_\Delta \end{aligned}$$

Now we can easily check for irreducible constraints in Δ:

$$\begin{aligned} (a_1 \neq a_2) \in \Delta &:= (a_1 \neq a_2) \in NeqAtoms_\Delta \\ (a \# X) \in \Delta &:= X \in Fresh_\Delta(a) \\ (X_1 \sim X_2) \in \Delta &:= |X_1|_\Delta = |X_2|_\Delta \\ (X \sim S) \in \Delta &:= S = Shape_\Delta(X_\Delta) \\ (S \prec X) \in \Delta &:= S \in SubShape_\Delta(X_\Delta) \end{aligned}$$

Now we can define rules for the special occurs check:

$$\frac{X_\Delta \text{ occurs syntactically in } |S|_\Delta}{\Delta \vDash X \text{ occurs in } S}$$

$$\frac{X'_\Delta \text{ occurs syntactically in } |S|_\Delta \qquad (S' \prec X') \in \Delta \qquad \Delta \vDash X \text{ occurs in } S'}{\Delta \vDash X \text{ occurs in } S}$$

And finally the rules for $\mathcal{C} \cup \Delta$. Note that we are using meta-field of *Assumptions* to indicate that some of the assumptions in Δ are no longer "simple" and escape from Δ back to Γ to be broken up by the *Solver*.

$$\{a \# X\} \cup \Delta := \Delta[Fresh(a) \mathrel{+}= X]$$

$$\{a \neq a'\} \cup \Delta := \begin{cases} \lightning & \text{if } a = a' \\ \Delta[NeqAtoms \mathrel{+}= (a \neq a')] & \text{otherwise.} \end{cases}$$

$$\begin{aligned} \{X \sim S\} \cup \Delta \quad &:= \quad \begin{cases} \lightning & \text{if } \Delta \vDash X \text{ occurs in } S \\ \Delta' & \text{otherwise.} \end{cases} \\ \text{where } \Delta' \quad &= \quad \Delta.Symbols\{X_\Delta \rightsquigarrow |S|_\Delta\} \\ &\qquad .Subshapes\{X_\Delta \rightsquigarrow |S|_\Delta\} \\ &\qquad .Shape\{X_\Delta \rightsquigarrow |S|_\Delta\} \end{aligned}$$

$$\begin{aligned} \{X \sim X'\} \cup \Delta \quad &:= \quad \begin{cases} \Delta & \text{if } X_\Delta = X'_\Delta \\ \Delta & \text{if } |X|_\Delta = |X'|_\Delta \\ \lightning & \text{if } X_\Delta \text{ occurs in } |X'|_\Delta \\ \lightning & \text{if } X'_\Delta \text{ occurs in } |X|_\Delta \\ \Delta' & \text{otherwise.} \end{cases} \\ \text{where } \Delta' \quad &= \quad \Delta.Symbols\{X_\Delta \rightsquigarrow X'_\Delta\} \\ &\qquad .Subshapes\{X_\Delta \rightsquigarrow X'_\Delta\} \\ &\qquad .TransferShape\{X_\Delta \rightsquigarrow X'_\Delta\} \\ &\qquad [\, Shape \mathrel{-}= (X_\Delta) \\ &\qquad ,SubShape \mathrel{-}= (X_\Delta) \\ &\qquad ,VarShape \mathrel{+}= (X_\Delta \mapsto X'_\Delta) \\ &\qquad] \end{aligned}$$

$$\Delta.Symbols\{X \rightsquigarrow S\} := \begin{cases} \Delta[Symbols \mathrel{-}= X, Assumptions \mathrel{+}= \text{symbol } S] & \text{if } X_\Delta \in \Delta.Symbols \\ \Delta & \text{otherwise.} \end{cases}$$

$$\Delta.Shape\{X \rightsquigarrow S\} := \begin{cases} \Delta[Assumptions \mathrel{+}= (S \sim S')] & \text{if } Shape_\Delta(X_\Delta) = S' \\ \Delta[Shapes \mathrel{+}= (X \mapsto S)] & \text{otherwise.} \end{cases}$$

$$\Delta.SubShapes\{X \rightsquigarrow S\} := \Delta[Assumptions \mathrel{+}= Subshapes_\Delta(X) \prec S]$$

$$\Delta.TransferShape\{X \rightsquigarrow X'\} := \begin{cases} \Delta.Shape\{termv' \rightsquigarrow S'\} & \text{if } Shape_\Delta(X_\Delta) = S \\ \Delta & \text{otherwise.} \end{cases}$$

$$\Delta\{X \mapsto t\} := \{X \sim |t|_\Delta\} \cup \Delta.Fresh\{X \mapsto t\}$$

$$\Delta.Fresh\{X \mapsto t\} := \Delta[Fresh.map(\text{fun } (a \mathrel{\#} \mathbb{X}) \mapsto a \mathrel{\#} (\mathbb{X} \backslash \{X\})] \cup \bigcup_{\substack{(a \,\#\, \mathbb{X}) \in Fresh_\Delta \\ X \in \mathbb{X}}} \{a \mathrel{\#} t\}$$

$$\Delta\{a \mapsto a'\} := \Delta.Fresh\{a \mapsto a'\}.NeqAtoms\{a \mapsto a'\}]$$

$$\Delta.Fresh\{a \mapsto a'\} := \Delta[Fresh \mathrel{-=} a][Fresh \mathrel{+=} \{a' \mathrel{\#} \Delta.Fresh(a)\}]$$

$$\Delta.NeqAtoms\{a \mapsto a'\} := \Delta[NeqAtoms = \emptyset] \cup \bigcup_{(a_1 \neq a_2) \in NeqAtoms_\Delta} \{a_1\{a \mapsto a'\} \neq a_2\{a \mapsto a'\}\}$$

Chapter 4

Higher Order Logic

On top of sublogic of constraints we build an higher order logic.

4.1 Kinds

$$\kappa \quad ::= \quad \star \mid \kappa \to \kappa \mid \forall_A a.\kappa \mid \forall_T X.\kappa \mid [c]\kappa \quad \text{(kinds)}$$

4.2 Subkinding

$$\frac{}{\Gamma \vdash \kappa <: \kappa} \qquad \frac{\Gamma \vdash \kappa_1 <: \kappa_2 \quad \Gamma \vdash \kappa_2 <: \kappa_3}{\Gamma \vdash \kappa_1 <: \kappa_3} \qquad \frac{\Gamma \vdash \kappa_1 <: \kappa_2}{\Gamma \vdash \forall_A a.\kappa_1 <: \forall_A a.\kappa_2} \qquad \frac{\Gamma \vdash \kappa_1 <: \kappa_2}{\Gamma \vdash \forall_T X.\kappa_1 <: \forall_T X.\kappa_2}$$

$$\frac{\Gamma \vdash \kappa_1' <: \kappa_1 \quad \Gamma \vdash \kappa_2 <: \kappa_2'}{\Gamma \vdash \kappa_1 \to \kappa_2 <: \kappa_1' \to \kappa_2'} \qquad \frac{\Gamma \vDash c}{\Gamma \vdash [c]\kappa <: \kappa} \qquad \frac{\Gamma, c \vdash \kappa_1 <: \kappa_2}{\Gamma \vdash \kappa_1 <: [c]\kappa_2}$$

Note that there is no structural subkinding rule for guarded kinds like

$$\frac{\Gamma \vdash \kappa_1 <: \kappa_2}{\Gamma \vdash [c]\kappa_1 <: [c]\kappa_2}.$$

Such a rule can be derived from both subkinding rules for guarded kind, transitivity, and weakening.

4.3 Formulas

$$\begin{array}{rcll} \varphi & ::= & \bot \mid \varphi \vee \varphi \mid \varphi \wedge \varphi \mid \varphi \to \varphi \mid c \mid [c] \wedge \varphi \mid [c] \to \varphi & \\ & \mid & \forall_A a.\varphi \mid \forall_T X.\varphi \mid \exists_A a.\varphi \mid \exists_T X.\varphi \mid \ \dots & \text{(formulas)} \end{array}$$

$$\frac{}{\Gamma; \Sigma \vdash c :: \star} \qquad \frac{\Gamma, c; \Sigma \vdash \varphi :: \star}{\Gamma; \Sigma \vdash [c] \wedge \varphi :: \star} \qquad \frac{\Gamma, c; \Sigma \vdash \varphi :: \star}{\Gamma; \Sigma \vdash [c] \to \varphi :: \star} \qquad \frac{\Gamma; \Sigma \vdash \varphi :: \kappa_1 \quad \Gamma \vdash \kappa_1 <: \kappa_2}{\Gamma; \Sigma \vdash \varphi :: \kappa_2}$$

$$\varphi \quad ::= \quad \dots \mid P \mid \lambda_A a.\varphi \mid \lambda_T X.\varphi \mid \lambda P :: \kappa.\varphi \mid \varphi\ \alpha \mid \varphi\ t \mid \varphi\ \varphi \mid \ \dots \quad \text{(formulas)}$$

4.4 Fixpoint

4.5 Proof theory

$$\frac{}{\Gamma;\varphi\vdash\varphi}\ (\mathit{Assumption})\qquad \frac{\Gamma;\varphi\vdash\bot}{\Gamma;\varphi\vdash\varphi}\ (\bot^e)\qquad \frac{\Gamma\models c}{\Gamma;\Theta\vdash c}\ (\mathit{constr}^i)\qquad \frac{\Gamma\models\bot}{\Gamma;\Theta\vdash\varphi}\ (\mathit{constr}^e)$$

$$\frac{\Gamma;\Theta,\varphi_1\vdash\varphi_2}{\Gamma;\Theta\vdash\varphi_1\to\varphi_2}\ (\to^i)\qquad \frac{\Gamma_1;\Theta_1\vdash\varphi_1\quad \Gamma_2;\Theta_2\vdash\varphi_1\to\varphi_2}{\Gamma_1\cup\Gamma_2;\Theta_2\cup\Theta_2\vdash\varphi_2}\ (\to^e)$$

$$\frac{\Gamma,c;\Theta\vdash\varphi}{\Gamma;\Theta\vdash[c]\to\varphi}\ ([\cdot]\to^i)\qquad \frac{\Gamma_1;\Theta_1\vdash c\quad \Gamma_2;\Theta_2\vdash[c]\to\varphi}{\Gamma_1\cup\Gamma_2;\Theta_2\cup\Theta_2\vdash\varphi}\ ([\cdot]\to^e)$$

$$\frac{\Gamma;\Theta\vdash\varphi_1}{\Gamma;\Theta\vdash\varphi_1\vee\varphi_2}\ (\vee_1^i)\qquad \frac{\Gamma;\Theta\vdash\varphi_2}{\Gamma;\Theta\vdash\varphi_1\vee\varphi_2}\ (\vee_2^i)\qquad \frac{\begin{array}{c}\Gamma;\Theta\vdash\varphi_1\vee\varphi_2\\ \Gamma;\Theta,\varphi_1\vdash\psi\quad \Gamma;\Theta,\varphi_2\vdash\psi\end{array}}{\Gamma;\Theta\vdash\psi}\ (\vee^e)$$

$$\frac{\Gamma_1;\Theta_1\vdash\varphi_1\quad \Gamma_2;\Theta_2\vdash\varphi_2}{\Gamma_1\cup\Gamma_2;\Theta_2\cup\Theta_2\vdash\varphi_1\wedge\varphi_2}\ (\wedge^i)\qquad \frac{\Gamma;\Theta\vdash\varphi_1\wedge\varphi_2}{\Gamma;\Theta\vdash\varphi_1}\ (\wedge_1^e)\qquad \frac{\Gamma;\Theta\vdash\varphi_1\wedge\varphi_2}{\Gamma;\Theta\vdash\varphi_2}\ (\wedge_2^e)$$

$$\frac{\Gamma\models c\quad \Gamma,c;\Theta\vdash\varphi}{\Gamma;\Theta\vdash[c]\wedge\varphi}\ ([\cdot]\wedge^i)\qquad \frac{\Gamma;\Theta\vdash[c]\wedge\varphi}{\Gamma;\Theta\vdash c}\ ([\cdot]\wedge_1^e)\qquad \frac{\Gamma\vdash[c]\wedge\varphi\quad \Gamma;\Theta\vdash\varphi:\star}{\Gamma;\Theta\vdash\varphi}\ ([\cdot]\wedge_2^e)$$

$$\frac{a\notin \mathrm{FV}(\Gamma;\Theta)\quad \Gamma;\Theta\vdash\varphi}{\Gamma;\Theta\vdash\forall_A a.\varphi}\ (\forall_A.^i)\qquad \frac{\Gamma;\Theta\vdash\forall_A a.\varphi}{\Gamma;\Theta\vdash\varphi\{a\mapsto a'\}}\ (\forall_A.^e)$$

$$\frac{X\notin \mathrm{FV}(\Gamma;\Theta)\quad \Gamma;\Theta\vdash\varphi}{\Gamma;\Theta\vdash\forall_T X.\varphi}\ (\forall_T.^i)\qquad \frac{\Gamma;\Theta\vdash\forall_T X.\varphi}{\Gamma;\Theta\vdash\varphi\{X\mapsto X'\}}\ (\forall_T.^e)$$

$$\frac{\Gamma;\Theta\vdash\varphi\{a\mapsto a'\}}{\Gamma;\Theta\vdash\exists_A a.\varphi}\ (\exists_A.^i)\qquad \frac{\begin{array}{c}\Gamma_1;\Theta_1\vdash\exists_A a.\varphi\\ \Gamma_2;\Theta_2,\varphi\{a\mapsto a'\}\vdash\psi\\ a'\notin \mathrm{FV}(\Gamma_1\cup\Gamma_2;\Theta_2\cup\Theta_2)\end{array}}{\Gamma_1\cup\Gamma_2;\Theta_2\cup\Theta_2\vdash\psi}\ (\exists_A.^e)$$

$$\frac{\Gamma;\Theta\vdash\varphi\{X\mapsto X'\}}{\Gamma;\Theta\vdash\exists_T X.\varphi}\ (\exists_T.^i)\qquad \frac{\begin{array}{c}\Gamma_1;\Theta_1\vdash\exists_T X.\varphi\\ \Gamma_2;\Theta_2,\varphi\{X\mapsto X'\}\vdash\psi\\ X'\notin \mathrm{FV}(\Gamma_1\cup\Gamma_2;\Theta_2\cup\Theta_2)\end{array}}{\Gamma_1\cup\Gamma_2;\Theta_2\cup\Theta_2\vdash\psi}\ (\exists_T.^e)$$

$$\frac{\Gamma\models a=\alpha\quad \Gamma;\Theta\vdash\varphi}{\Gamma\{a\mapsto\alpha\};\Theta\{a\mapsto\alpha\}\vdash\varphi\{a\mapsto\alpha\}}\ (\mapsto_A)\qquad \frac{\Gamma\models X=t\quad \Gamma;\Theta\vdash\varphi}{\Gamma\{X\mapsto t\};\Theta\{X\mapsto t\}\vdash\varphi\{X\mapsto t\}}\ (\mapsto_T)$$

$$\frac{\Gamma;\Theta\vdash\psi\quad \Gamma;\Theta\vdash\psi\equiv\varphi}{\Gamma;\Theta\vdash\varphi}\ (\mathit{Equiv})$$

$$\frac{\Gamma; \Theta, (\forall_T X'.[X' \prec X] \to \varphi(X')) \vdash \varphi(X)}{\Gamma; \Theta \vdash \forall_T X.\varphi(X)} \quad (\mathit{Induction})$$

$$\frac{}{\vdash \forall_A\, a,\ a'.(a = a') \vee (a \neq a')} \quad (\mathit{Axiom}_{\mathit{Compare}})$$

$$\frac{}{\vdash \forall_T X.\ \exists_A a.(a \,\#\, X)} \quad (\mathit{Axiom}_{\mathit{Fresh}})$$

$$\frac{}{\begin{array}{c}\vdash \forall_T X.(\exists_A a.\ X = a) \vee (\exists_A a.\ \exists_T X'.\ X = a.X') \\ \vee(\exists_T X_1,\ X_2.\ X = a.X') \vee (\mathit{symbol}\ X)\end{array}} \quad (\mathit{Axiom}_{\mathit{Inversion}})$$

The equivalence relation $(\varphi_1 \equiv \varphi_2)$ is rather straightforward, the only interesting parts being fixpoint, functions and applications.

$$\frac{\Gamma; \Sigma \vdash \varphi_1[X_1 \mapsto t_1] \equiv \varphi_2[X_2 \mapsto t_2]}{\Gamma; \Sigma \vdash (\lambda_T X_1.\varphi_1)\ t_1 \equiv (\lambda_T X_2.\varphi_2)\ t_2}$$

(TODO: Also we do kind-checking). Otherwise we compute weak head normal form and recurse on subformulas:

$$\frac{\Gamma; \Sigma; Sn \vdash (\text{fix } P(X).\ \varphi)\ t}{\Gamma; \Sigma, P \mapsto \varphi; n \vdash \varphi[X \mapsto t]}$$

Until we *run out of gas* or cannot compute the formula further, resorting to *naive* checking:

$$\frac{\Gamma \vDash t_1 = t_2 \qquad \Gamma; \Sigma \vdash \varphi_1 \equiv \varphi_2}{\Gamma; \Sigma \vdash \varphi_1\ t_1 \equiv \varphi_2\ t_2}$$

$$\frac{\begin{array}{c} X \notin \mathrm{FV}(\Gamma; \Sigma) \\ \Gamma; \Sigma \vdash \varphi_1[X_1 \mapsto X] \equiv \varphi_2[X_2 \mapsto X]\end{array}}{\Gamma; \Sigma \vdash \lambda_T X_1.\varphi_1 \equiv \lambda_T X_2.\varphi_2}$$

$$\frac{\begin{array}{c} P \notin \mathrm{FV}(\Gamma; \Sigma) \qquad\qquad X \notin \mathrm{FV}(\Gamma; \Sigma) \\ \Gamma; \Sigma \vdash \varphi_1[P_1 \mapsto P, X_1 \mapsto X] \equiv \varphi_2[P_2 \mapsto P, X_2 \mapsto X]\end{array}}{\Gamma; \Sigma \vdash \text{fix } P_1(X_1).\ \varphi_1 \equiv \text{fix } P_2(X_2).\ \varphi_2}$$

Note that we allow *different terms* in equivalent formulas as long as constraints-enviroment Γ ensures their equality is provable. Quantifiers are handled the same way as function above — as they all are a form of bind. (TODO: Also we do constraints-equivalence).

Chapter 5

Model

Definition of a model of our logic is bit involved, due to presence of subkinding relation. We will proceed in two steps. First, for each kind κ we define its *domain* $\mathcal{D}_\kappa$. Then we will interpret each kind as a predicate on elements of its domain. We fix some Heyting algebra $\mathcal{H}$ in which we will interpret propositions. Then kind domains are defined in the following way.

$$\begin{array}{rcl}
\mathcal{D}_\star & = & \mathcal{H} \\
\mathcal{D}_{\kappa_1 \to \kappa_2} & = & \mathcal{D}_{\kappa_1} \to \mathcal{D}_{\kappa_2} \\
\mathcal{D}_{\forall_A a.\kappa} & = & \mathcal{A} \to \mathcal{D}_\kappa \\
\mathcal{D}_{\forall_T X.\kappa} & = & \mathcal{T} \to \mathcal{D}_\kappa \\
\mathcal{D}_{[c]\kappa} & = & \mathcal{D}_\kappa
\end{array}$$

And kind interpretation like this:

$$\begin{array}{rcl}
[\![\star]\!]_\rho & = & \{\bot, \top\} \\
[\![\kappa_1 \to \kappa_2]\!]_\rho & = & \{f \mid \forall P \in [\![\kappa_1]\!]_\rho.\ f(P) \in [\![\kappa_2]\!]_\rho\} \\
[\![\forall_A a.\kappa]\!]_\rho & = & \{f \mid \forall A \in \mathcal{A}.\ f(A) \in [\![\kappa]\!]_{\rho[a \mapsto A]}\} \\
[\![\forall_T X.\kappa]\!]_\rho & = & \{f \mid \forall T \in \mathcal{T}.\ f(T) \in [\![\kappa]\!]_{\rho[X \mapsto T]}\} \\
[\![[c]\kappa]\!]_\rho & = & \{x \mid \rho \vDash c \implies x \in [\![\kappa]\!]_\rho\}
\end{array}$$

And finally the kind derivation model:

$$\begin{array}{rcl}
\left[\!\!\left[\dfrac{}{\Gamma \vdash \top : \star} \right]\!\!\right]_\rho & = & \top \\
\left[\!\!\left[\dfrac{}{\Gamma \vdash P : \Gamma(P)} \right]\!\!\right]_\rho & = & \rho(P) \\
\left[\!\!\left[\dfrac{}{\Gamma \vdash c : \star} \right]\!\!\right]_\rho & = & \texttt{if } \rho \vDash c \texttt{ then } \top \texttt{ else } \bot
\end{array}$$

$$\left\llbracket \frac{\begin{array}{c} D_1 : \Gamma \vdash \varphi_1 : \star \\ D_2 : \Gamma \vdash \varphi_2 : \star \end{array}}{\Gamma \vdash \varphi_1 \wedge \varphi_2 : \star} \right\rrbracket_\rho = \llbracket D_1 \rrbracket_\rho \wedge_{\mathcal{H}} \llbracket D_2 \rrbracket_\rho$$

$$\left\llbracket \frac{\begin{array}{c} D_1 : \Gamma \vdash \varphi_1 : \star \\ D_2 : \Gamma \vdash \varphi_2 : \star \end{array}}{\Gamma \vdash \varphi_1 \vee \varphi_2 : \star} \right\rrbracket_\rho = \llbracket D_1 \rrbracket_\rho \vee_{\mathcal{H}} \llbracket D_2 \rrbracket_\rho$$

$$\left\llbracket \frac{\begin{array}{c} D_1 : \Gamma \vdash \varphi_1 : \star \\ D_2 : \Gamma \vdash \varphi_2 : \star \end{array}}{\Gamma \vdash \varphi_1 \Rightarrow \varphi_2 : \star} \right\rrbracket_\rho = \llbracket D_1 \rrbracket_\rho \Rightarrow_{\mathcal{H}} \llbracket D_2 \rrbracket_\rho$$

$$\left\llbracket \frac{D : \Gamma \vdash \varphi : \star}{\Gamma \vdash \forall_T X.\varphi : \star} \right\rrbracket_\rho = \bigwedge_{T \in Term} \llbracket D \rrbracket_{\rho[X \mapsto T]}$$

$$\left\llbracket \frac{D : \Gamma \vdash \varphi : \star}{\Gamma \vdash \forall_A a.\varphi : \star} \right\rrbracket_\rho = \bigwedge_{A \in Atom} \llbracket D \rrbracket_{\rho[a \mapsto A]}$$

$$\left\llbracket \frac{D : \Gamma \vdash \varphi : \star}{\Gamma \vdash \exists_T X.\varphi : \star} \right\rrbracket_\rho = \bigvee_{T \in Term} \llbracket D \rrbracket_{\rho[X \mapsto T]}$$

$$\left\llbracket \frac{D : \Gamma \vdash \varphi : \star}{\Gamma \vdash \exists_A a.\varphi : \star} \right\rrbracket_\rho = \bigvee_{A \in Atom} \llbracket D \rrbracket_{\rho[a \mapsto A]}$$

$$\left\llbracket \frac{D : \Gamma, c \vdash \varphi : \star}{\Gamma \vdash [c] \wedge \varphi : \star} \right\rrbracket_\rho = \texttt{if } \rho \vDash c \texttt{ then } \llbracket D \rrbracket_\rho \texttt{ else } \bot$$

$$\left\llbracket \frac{D : \Gamma, c \vdash \varphi : \star}{\Gamma \vdash [c] \Rightarrow \varphi : \star} \right\rrbracket_\rho = \texttt{if } \rho \vDash c \texttt{ then } \llbracket D \rrbracket_\rho \texttt{ else } \top$$

$$\left\llbracket \frac{D : \Gamma \vdash \varphi : \kappa_2}{\Gamma \vdash \lambda P.\ \varphi : \kappa_1 \Rightarrow \kappa_2} \right\rrbracket_\rho = \lambda\,(Q : \llbracket \kappa_1 \rrbracket_\rho).\ \llbracket D \rrbracket_{\rho[P \mapsto Q]}$$

$$\left\llbracket \frac{D : \Gamma \vdash \varphi : \kappa}{\Gamma \vdash \lambda a.\ \varphi : \forall_A a.\kappa} \right\rrbracket_\rho = \lambda\,(A : Atom).\ \llbracket D \rrbracket_{\rho[a \mapsto A]}$$

$$\left\llbracket \frac{D : \Gamma \vdash \varphi : \kappa}{\Gamma \vdash \lambda X.\ \varphi : \forall_T X.\kappa} \right\rrbracket_\rho = \lambda\,(T : Term).\ \llbracket D \rrbracket_{\rho[X \mapsto T]}$$

$$\left\llbracket \frac{\begin{array}{c} D_1 : \Gamma \vdash \varphi_1 : \kappa' \Rightarrow \kappa \\ D_2 : \Gamma \vdash \varphi_2 : \kappa' \end{array}}{\Gamma \vdash \varphi_1\ \varphi_2 : \kappa} \right\rrbracket_\rho = \llbracket D_1 \rrbracket_\rho\ \llbracket D_2 \rrbracket_\rho$$

$$\left\llbracket \frac{D : \Gamma \vdash \varphi : \forall_A a.\kappa}{\Gamma \vdash \varphi(\alpha) : \kappa\{a \mapsto \alpha\}} \right\rrbracket_\rho = \llbracket D \rrbracket_\rho\ \llbracket \alpha \rrbracket_\rho$$

$$\left\llbracket \frac{D : \Gamma \vdash \varphi : \forall_T X.\kappa}{\Gamma \vdash \varphi(t) : \kappa\{X \mapsto t\}} \right\rrbracket_\rho = \llbracket D \rrbracket_\rho\ \llbracket X \rrbracket_\rho$$

$$\left[\!\!\left[\frac{D : \Gamma, X : \forall z.\, [z < X']\, \kappa\{z \mapsto X'\} \vdash \varphi : \kappa}{\Gamma \vdash \text{fix } X(X').\, \varphi : \forall X'.\, \kappa} \right]\!\!\right]_\rho = \lim_{n\to\infty} f_n$$

$$\texttt{where } f_0(t) = \bot$$

$$\texttt{and } f_{n+1}(t) = [\![D]\!]_{\rho[X \mapsto f_n, X' \mapsto t]}$$

$$\left[\!\!\left[\frac{D : \Gamma, c \vdash \varphi : \kappa}{\Gamma \vdash \varphi : [c]\kappa} \right]\!\!\right]_\rho = \texttt{if } \rho \vDash c \texttt{ then } [\![D]\!]_\rho \texttt{ else } "\bot"$$

$$\left[\!\!\left[\frac{\begin{array}{c} D : \Gamma \vdash \varphi : \kappa \\ \Gamma \vdash \kappa \leq \kappa' \end{array}}{\Gamma \vdash \varphi : \kappa'} \right]\!\!\right]_\rho = [\![D]\!]_\rho$$

5.1 Fundamental Theorem

For any formula φ, any kind κ, and any environment Γ, for any kind derivation $D : \Gamma \vdash \varphi : \kappa$ under any interpretation $\rho \in [\![\Gamma]\!]$, we have that

$$[\![D]\!]_\rho \in [\![\kappa]\!]_\rho$$

In other words, each kind derivation D has a semantic witness that inhabits the semantic interpretation of κ.

Chapter 6

Proof assistant

...

Chapter 7

Case study: Progress and Preservation of STLC

. . .

Chapter 8

Conclusion and future work

...

Bibliography

[1] . . .