SUBJECT REDUCTION FOR PURE TYPE SYSTEMS

Charlotte Marchal | ????????? Dashiell Rich | 261002837
Milton Rosenbaum | 260972050 Zhaoshen Zhai | 261003108

Abstract. Following [GN91] and [Bar91], we study the basics of *pure type systems*, which abstract many of the constructs found in the eight systems of the λ-*cube*. We start with a brief introduction to the systems of the λ-cube, discuss their expressibilities, and introduce pure type systems as a uniform framework in which they can be studied. We then give a detailed proof of subject reduction for arbitrary pure type systems.

Introduction. Subject reduction is a crucial property of a type system that guarantees its 'computational consistency' by ensuring that reductions of a well-typed expression remains well-typed, and which supports the slogan that 'well-typed programs do not go wrong'. It is thus desirable that we can prove it uniformly across many different type systems, and this is the goal of the present note.

To this end, **TODO:** review λ-cube.

Definition 1. A *pure type system* is **TODO**

Definition 2. The β-*reduction* relation is the least relation on σ-terms satisfying the following for all σ-terms A, A', A'': the *principal reduction rule* $(\lambda x : A.A')A'' \to_\beta A'[A''/x]$, and the *congruence rules* $A\,A' \succ A\,A''$, $A'\,A \succ A''\,A$, $\lambda x:A.A' \succ \lambda x:A.A''$, $\lambda x:A'.A \succ \lambda x:A''.A$, $\Pi x:A.A' \succ \Pi x:A.A''$, and $\Pi x:A'.A \succ \Pi x:A''.A$.

Notation 3. We write $\twoheadrightarrow_\beta$ for the reflexive and transitive closure of $\to_\beta$, and $=_\beta$ for the equivalence relation generated by $\twoheadrightarrow_\beta$. A σ-term of the form $(\lambda x:A.A')A''$ is called a β-*redex*.

Theorem 4 (Subject Reduction)**.** *Let* Γ, Γ' *be* σ*-contexts,* M, M', N *be* σ*-terms, and suppose that* $\Gamma \vdash M : N$*.*

1. *If* $M \twoheadrightarrow_\beta M'$*, then* $\Gamma \vdash M' : N$*.*
2. *If* $\Gamma \twoheadrightarrow_\beta \Gamma'$*, then* $\Gamma' \vdash M : N$*.*

Proof. **TODO** ■

A. Technical Lemmas

Lemma (Substitution Lemma)**.**

Lemma (Stripping Lemma)**.**

References

[GN91] H. Geuvers and M. Nederhof, *Modular proof of strong normalization for the calculus of constructions*, Journal of Functional Programming **1** (1991), no. 2, 155-189.

[Bar91] H. Barendregt, *Introduction to Generalized Type Systems*, Journal of Functional Programming **1** (1991), no. 2, 125-154.

Date: April 20, 2025.
Extended abstracted for the final project for Comp527: Logic and Computation taught by Professor Brigitte Pientka.
Link to presentation: **TODO**