

# First-class Polymorphism: Type Inference and Extensions

by

Ningning Xie  
(谢宁宁)



A thesis submitted in partial fulfillment of the requirements for  
the degree of Doctor of Philosophy  
at The University of Hong Kong

February 2021



Abstract of thesis entitled  
“First-class Polymorphism: Type Inference and Extensions”

Submitted by  
Ningning Xie

for the degree of Doctor of Philosophy  
at The University of Hong Kong  
in February 2021

---



# DECLARATION

I declare that this thesis represents my own work, except where due acknowledgment is made, and that it has not been previously included in a thesis, dissertation or report submitted to this University or to any other institution for a degree, diploma or other qualifications.

.....

Ningning Xie

February 2021



## ACKNOWLEDGMENTS





# CONTENTS

DECLARATION	I
ACKNOWLEDGMENTS	III
LIST OF FIGURES	VII
LIST OF TABLES	IX
I PROLOGUE	1
1 INTRODUCTION	3
1.1 Contributions . . . . .	3
1.2 Organization . . . . .	4
2 BACKGROUND	5
II TECHNICAL CONTRIBUTIONS	7
3 HIGHER RANK TYPE INFERENCE WITH THE APPLICATION MODE	9
4 HIGHER RANK GRADUAL TYPES	11
5 TYPE PROMOTION	13
III RELATED AND FUTURE WORK	15
6 RELATED WORK	17
7 FUTURE WORK	19
	 v

## Contents

IV	EPILOGUE	21
8	CONCLUSION	23
	BIBLIOGRAPHY	25

## LIST OF FIGURES



## LIST OF TABLES



# PART I

## PROLOGUE





# 1 INTRODUCTION

## 1.1 CONTRIBUTIONS

In summary the contributions of this thesis are:

- Section 3 proposes a new design for type inference of higher-ranked types.
  - We design a variant of bi-directional type checking where the inference mode is combined with a new, so-called, application mode. The application mode naturally propagates type information from arguments to the functions.
  - With the application mode, we give a new design for type inference of higher-ranked type, which generalizes the HM type system, supports a polymorphic let as syntactic sugar, and infers higher rank types. We present a syntax-directed specification, an elaboration semantics to System F, and an algorithmic type system with completeness and soundness proofs.
- Section 4 presents the type system GPC, which extends predicative implicit higher-rank polymorphism with gradual types.
  - We define a framework for consistent subtyping with a new definition of consistent subtyping that subsumes and generalizes that of Siek and Taha [2007] and can deal with polymorphism and top types; and a syntax-directed version of consistent subtyping that is sound and complete with respect to our definition of consistent subtyping, but still guesses instantiations.
  - Based on consistent subtyping, we present GPC. We prove that our calculus satisfies the static aspects of the refined criteria for gradual typing Siek et al. [2015], and is type-safe by a type-directed translation to  $\lambda B$  Ahmed et al. [2009].
  - We present a sound and complete bidirectional algorithm for implementing the declarative system based on the design principle of Garcia and Cimini [2015].
- We present a new design of the context for ???, where ???.
  - We come up with a strategy called promotion that resolves the dependency between types.

## 1 Introduction

- We further explore the design of promotion by an application to kind inference for datatypes.
  - We formalize Haskell98’ s datatype declarations, providing both a declarative specification and syntax-driven algorithm for kind inference. We prove that the algorithm is sound and observe how Haskell98’ s technique of defaulting unconstrained kinds to  $\lambda$  leads to incompleteness. We believe that ours is the first formalization of this aspect of Haskell98. Its inclusion in this paper both sheds light on this historically important language and also prepares us for the more challenging features of modern Haskell.
- A comprehensive Coq mechanization of all metatheory, including type safety, coherence, algorithmic soundness and completeness, etc.<sup>1</sup> This has notably revealed several missing lemmas and oversights in Pierce’s manual proof of BCD’s algorithmic subtyping [?]. As a by-product, we obtain the first mechanically verified BCD-style subtyping algorithm with coercions.

### 1.2 ORGANIZATION

This thesis is largely based on the publications by the author [Xie et al. 2018, 2019a,b; Xie and Oliveira 2018], as indicated below.


????: Ningning Xie and Bruno C. d. S. Oliveira. 2018. “Let Arguments Go First”. In European Symposium on Programming (ESOP).

????: Ningning Xie, Xuan Bi, and Bruno C. d. S. Oliveira. 2018. “Consistent Subtyping for All”. In European Symposium on Programming (ESOP).

????: Ningning Xie, Xuan Bi, Bruno C. d. S. Oliveira, and Tom Schrijvers. 2019. “Consistent Subtyping for All”. In ACM Transactions on Programming Languages and Systems (TOPLAS).

????: Ningning Xie, Richard Eisenber and Bruno C. d. S. Oliveira. 2020. “Kind Inference for Datatypes”. In Kind Inference for Datatypes.

---

<sup>1</sup>For convenience, whenever possible, definitions, lemmas and theorems have hyperlinks (click ) to their Coq counterparts. Also since  $F_i^+$  completely subsumes  $\lambda_i^+$ , to save work, for  $\lambda_i^+$  metatheory we provide cross references to the corresponding  $F_i^+$  Coq definitions, instead.

## 2 BACKGROUND



## PART II

## TECHNICAL CONTRIBUTIONS



# 3 HIGHER RANK TYPE INFERENCE WITH THE APPLICATION MODE





## 4 HIGHER RANK GRADUAL TYPES



# 5 TYPE PROMOTION



## PART III

### RELATED AND FUTURE WORK



## 6 RELATED WORK





## 7 FUTURE WORK



## PART IV

## EPILOGUE



## 8 CONCLUSION



# BIBLIOGRAPHY

[Citing pages are listed after each reference.]

Amal Ahmed, Robert Bruce Findler, Jacob Matthews, and Philip Wadler. 2009. Blame for All. In Proceedings for the 1st Workshop on Script to Program Evolution (STOP '09). Association for Computing Machinery, New York, NY, USA, 1–13. <https://doi.org/10.1145/1570506.1570507> [cited on page 3]

Ronald Garcia and Matteo Cimini. 2015. Principal Type Schemes for Gradual Programs. In Proceedings of the 42nd Annual ACM SIGPLAN-SIGACT Symposium on Principles of Programming Languages (POPL '15). Association for Computing Machinery, New York, NY, USA, 303–315. <https://doi.org/10.1145/2676726.2676992> [cited on page 3]

Jeremy Siek and Walid Taha. 2007. Gradual Typing for Objects. In Proceedings of the 21st European Conference on Object-Oriented Programming (ECOOP'07). Springer-Verlag, Berlin, Heidelberg, 2–27. [cited on page 3]

Jeremy G Siek, Michael M Vitousek, Matteo Cimini, and John Tang Boyland. 2015. Refined criteria for gradual typing. In 1st Summit on Advances in Programming Languages (SNAPL 2015). Schloss Dagstuhl-Leibniz-Zentrum fuer Informatik. [cited on page 3]

Ningning Xie, Xuan Bi, and Bruno C d S Oliveira. 2018. Consistent Subtyping for All. In European Symposium on Programming. Springer, 3–30. [cited on page 4]

Ningning Xie, Xuan Bi, Bruno C. D. S. Oliveira, and Tom Schrijvers. 2019a. Consistent Subtyping for All. *ACM Transactions on Programming Languages and Systems* 42, 1, Article 2 (Nov. 2019), 79 pages. <https://doi.org/10.1145/3310339> [cited on page 4]

Ningning Xie, Richard A. Eisenberg, and Bruno C. d. S. Oliveira. 2019b. Kind Inference for Datatypes. *Proc. ACM Program. Lang.* 4, POPL, Article 53 (Dec. 2019), 28 pages. <https://doi.org/10.1145/3371121> [cited on page 4]

Ningning Xie and Bruno C d S Oliveira. 2018. Let Arguments Go First. In European Symposium on Programming. Springer, 272–299. [cited on page 4]