A Bibliography About Formalized Binding

Randy Pollack

October 19, 2011

Some questions to consider when you look at these papers.

Is the representation general? Most representations only treat unitary binding; what about binding a list at one time? What about binding a set at one time? Can simultaneous substitution be represented? Explicit substitution? Some of the representations are even more special: [BHKM] only treats simply typed languages. Some Lambda-free logical frameworks [Ada08] can only represent object systems with specific finite orders.

Every approach mentioned below has something to say about instantiation of abstractions and substitution. Most have something to say about induction over the structure of the object language. You probably also need to define relations inductively, (e.g. typing, reduction) and functions recursively (e.g. height of a term, set of free variables). You will want to do rule induction (e.g. weakening of a typing relation). Maybe you need unification respecting binding over the language.

In some cases you might choose a representation tailored to the problem at hand, but I am interested in formalizing large parts of mathematics in the long term, and any special purpose representation will turn out to be inadequate for some naturally occurring example. Generality and expressiveness are important for me.

Is the representation easy to use? Basically, you need tool support to do significant reasoning about languages with binding. For example some object languages have several syntactic classes of variable: term variables, type variables, module variables, . . . . Defining substitution by hand for term variables in terms, for type variables in terms, . . . , may become not just annoying, but prohibitive. Tool support is needed. In [BHKM], using new tool support, some extremely smart people do some rather specialized examples that no one had been able to do before. At the moment [PSR11] is a beautiful representation, but with no tool support. The similar representation [ACP + 08] has a useful library of metatheory and tactics. Nominal Isabelle [UBN07, Urb08, BU08] has several person-years of development and good tool support. Similarly Twelf [HL07] is highly developed, with many users. Newer systems like Abella [GMN09, Gac08] may have excellent ideas behind them, but not be ready for large scale use. Systems with very few users [HMS01, FM09] are likely hard to get into.
1 First-order Representations

The type of lambda terms contains no actual binding in these representations; “binding” is in how terms are used.

1.1 Local representations

These use two species of names: local bound variables vs. global or free variables (parameters), following classical practice of Frege, Gentzen and Prawitz. The original locally named development [MP99, MP93] is not a canonical representation. Locally nameless [Gor93, ACP+08, Cha] was introduced to get a canonical representation. Finally a canonical named representation [PSR11] developed.


1.2 Concrete or quotiented representation

Very few suggest unquotiented concrete representation [VB03, Sto88]. With extensional logics such as HOL4 and Isabelle/HOL it is possible to work with quotients [Nor06].
1.3 Pure de Bruijn representation

Rather than binding by name equality, use natural numbers to index binding depth. Original paper \[dB72\]. Some applications \[NV07, Alt93, Ber11, Vou11\]. A priori it seems like a clumsy representation (consider object theorems about permutation of typing contexts, for example), but its proponents often manage concise and elegant examples.

---


2 Nominal representation

Nominal set theory [GP02]. Nominal logic as a theory [Pit03, Pit06]. An up-to-date review of nominal unification [Urb10]. Nominal representation implemented in Isabelle/HOL [UBN07, Urb08, BU08]. Second generation nominal Isabelle [HU10, UK11] explicitly supports binding of sets and lists of names.

---


3 Higher-order Representations

Higher order properties of the meta language, such as function spaces or recursive datatypes, are used in these representations.
3.1 Higher Order Abstract Syntax (HOAS)

Original LF paper [HHP93]. Read [HL07] to understand LF and TWELF, the modern implementation of LF. Formal proof of adequacy of representation [CNV]. Many other HOAS approaches [PGO10, FP10, Mil00]. Two-layer approaches are interesting [GMN09, Gac08, FM09]. Another variant: Weak HOAS [HMS01, Chl08]. Lambda-free logical frameworks [Ada08, Luo03] are designed to have good algebraic properties, and elementary metatheory so that the proof of adequacy of representation of object systems is simple.

---


3.2 Nested datatypes

[BP99, Ada06, BHKM] [AR99] is for category theorists. [Chl08], categorized as Weak HOAS above, also uses nested datatypes.

---


4 Theories of binding

We already saw nominal logic as a first order theory [Pit03]. Here are more first order or axiomatic approaches [PR09, GM96, DHK02, PG11].


5 The POPLmark challenge

A set of challenge problems involving reasoning about $F_\subseteq$. See

https://alliance.seas.upenn.edu/~plclub/cgi-bin/popmark

for solutions showing formal reasoning about binding in several proof tools (Twelf, Coq, Isabelle/HOL, Matita, Abella) using several representations (pure de Bruijn, HOAS, locally nameless, concrete first order, nominal, two level HOAS, nested datatypes).

6 Programming with binders

Logic programming [CU08, NM88]. Functional programming in simply-typed languages [MM04, WYS11, Pot06, BH94, Che05]. Dependently typed languages [Pou11, PS09, Pie10, PD10, PC12, LZH08, LH09].


