Subtype Universes

Harry Maclean and Zhaohui Luo

Royal Holloway, University of London, U.K.
harryjmaclean@gmail.com
zhaohui.luo@hotmail.co.uk

Introduction. Martin-Löf [9] introduced the concept of a universe in order to describe collections of types. The universe Type_0 contains all small types, Type_0 itself is contained in Type_1, and so on. This predicative hierarchy of universes is designed to provide expressive power (we can quantify over collections of types) whilst avoiding paradoxes.

In this paper we explore a new form of universe: a collection of subtypes. For any type A we define a universe $U(A)$ as the collection of all *subtypes* of A. We work in a system UTT[$\mathbb{C}$] which is an extension of UTT [6] with coercive subtyping [8] specified by means of a set $\mathbb{C}$ of coherent coercions. Our subtype universe is defined by the addition of two rules (for brevity we use universe formulation *à la Russell*):

$$\frac{\Gamma \vdash H : \mathbf{Type}}{\Gamma \vdash U(H) : \mathbf{Type}}\ U\text{-formation} \qquad \frac{\Gamma \vdash A \leq H : \mathbf{Type}}{\Gamma \vdash A : U(H)}\ U\text{-introduction}$$

U-formation declares that for type H, $U(H)$ is also a type, and U-introduction says that for any type A which is a subtype of H, A is in the universe $U(H)$. Note that H is always contained in $U(H)$ and, as a degenerate case, $U(H)$ contains only H when $\mathbb{C}$ is empty. We now have a type for "all subtypes of H", and can therefore quantify over this type. For example, $\forall(X : U(H)).P(X)$ is the proposition that a particular property P holds for all subtypes of H.

This subtype universe neatly models bounded quantification (of the form $\Pi(A \leq H).P(A)$) whilst avoiding the type checking issues traditionally associated with it [10]. We show that a specific form of this construction is a logically consistent extension of UTT[$\mathbb{C}$], explore applications in programming and natural language semantics, and discuss possible further work.

Application to programming. Subtype universes provide an alternative model for bounded quantification [3]. For example, we can define a polymorphic identity function over all subtypes of H as $id_H = \lambda X.\lambda x.x : \Pi(X : U(H)).X \to X$. In a system without subtype universes or bounded quantification the equivalent function would be $id_H = \lambda x.x : H \to H$. Given an object $h : H'$ where $H' < H$, the expression $id_H(h)$ is well-typed via subtype subsumption, but the result will be an object of type H - we have lost the information that h is of type H', simply by passing it through the identity function. By specifying the argument type and not relying on subsumption, both bounded quantification and subtype universes avoid this problem.

As a more compelling example, consider a type of (non-dependent) records, representing heterogeneous sets of labelled values. We write a record type as $\{x : A, y : B, ...\}$ where x, y are field labels and A, B are field types. We define a subtyping relation on records as follows:

$$\frac{\Gamma \vdash A : \mathbf{Type} \qquad \Gamma \vdash R : RType}{\Gamma \vdash \{R, x : A\} \leq R : \mathbf{Type}}\ (x \notin R)$$

where R is the type of records and $\{R, x : A\}$ denotes the extension of R by a field x of type A, under the assumption that R does not have a field labelled x. $RType$ is the kind of record types, as described in [7]. We can write the type of a function that extracts the value named "length" as

$$getLength : \Pi(R : U(\{length : Int\})).R \to Int$$

where $U(\{length : Int\})$ is the universe of subtypes of the record type $\{length : Int\}$. $getLength$ can be applied to any record which has a field labelled "length" of type Int.

Unlike bounded quantification, subtype universes can appear anywhere in a type. For example the codomain of the function $f : U(H) \to U(H) \to U(H)$ is a universe, and $U(H) \times A$ is a product type containing a universe.

Application to natural language semantics. In formal semantics based on type theories [11, 5, 4], common nouns are interpreted as types and subtype universes are useful in semantic constructions. For example, they can be employed to model gradable adjectives (words such as "tall"). Gradable adjectives map their arguments on to a totally ordered set of degrees, or scale. For "tall", this scale is height. Consider a universe T of base types containing objects for which we can determine their height, which may contain $Human : T$ and $Building : T$. We can define the type of "tall" as $\text{tall} : \Pi(H : T).\Pi(A : U(H)).(A \to Prop)$.

In words, "tall" is a predicate on subtypes of types in T, and $U(H)$ is the subtype universe for H. Thus if we have a type $Man < Human$ and an object $socrates : Man$ then $\text{tall}(Human, Man, socrates) : Prop$. An example definition is $\text{tall}(H, A, x) \equiv height(x) \geq \xi(H, A)$, where $\xi : \Pi(H : T).(U(H) \to \mathbb{N})$ is a function that calculates the "threshold" height for any subtype of a type in T.

Metatheoretic correctness. We have proved that the extension of UTT[$\mathbb{C}$] with the two universe formation rules is not problematic in a simplified case of the more general system. We pick a specific (but arbitrary) small type H and a single subtyping rule:

$$\frac{\Gamma, x : A \vdash P : Prop}{\Gamma \vdash \Sigma(x : A).P \leq_{\pi_1} A : \mathbf{Type}}$$

where $\pi_1 : \Sigma(x : A).P \to A$ is the first projection for pairs. We prove that all theorems of this simplified system (written $\text{UTT}_H[\mathbb{C}]$) are derivable in UTT[$\mathbb{C}$] via a transformation δ, which maps $U(H)$ to Type_0 but leaves terms otherwise unchanged. A corollary to this is that $\text{UTT}_H[\mathbb{C}]$ is logically consistent, as UTT[$\mathbb{C}$] is a conservative extension of UTT, which is itself a consistent system.

Recent work has extended this result to an arbitrary but coherent set of coercions obeying a restriction that we define here informally: for any coercion $c : A \to B$ we require $A : \text{Type}_i$, $B : \text{Type}_j$ and $i \leq j$. Care must be taken to ensure that coercions cannot themselves mention subtype universes, as otherwise we can derive potentially paradoxical judgements. For example, from the coercion $c : U(H) \to H$ we can derive a judgement $\Gamma \vdash U(H) : U(H)$. The relationship between subtype universes and the existing predicative universe hierarchy is ambiguous and in need of further investigation.

Conclusion. Subtype universes provide a powerful mechanism for quantifying over collections of subtypes without the undecidability issues of bounded quantification. We give several example applications in programming and natural language semantics; there are surely others.

Subtype universes bear similarities to Cardelli's power type [2] Power(A), a type containing all subtypes of A. Cardelli's formulation uses structural subtyping and a system with the logically inconsistent Type : Type, whereas our system is built on the logically consistent UTT. Aspinall's λ_{Power} [1] is a predicative and simplified alternative to Cardelli's system, but it has been difficult to prove some of its metatheoretic properties (such as subject reduction).

So far we have focused on a specific (but arbitrary) type $H : \text{Type}_0$. We are hopeful that the result can be extended to any type, and this will be the focus of future work.

References

[1] David Aspinall. Subtyping with power types. In *International Workshop on Computer Science Logic*, pages 156–171. Springer, 2000.

[2] Luca Cardelli. Structural subtyping and the notion of power type. *POPL*, 1988.

[3] Luca Cardelli and Peter Wegner. On understanding types, data abstraction, and polymorphism. *ACM Computing Surveys (CSUR)*, 17(4):471–523, 1985.

[4] S. Chatzikyriakidis and Z. Luo. *Formal Semantics in Modern Type Theories*. Wiley & ISTE Science Publishing Ltd., 2020. (to appear).

[5] Z. Luo. Formal semantics in modern type theories with coercive subtyping. *Linguistics and Philosophy*, 35(6):491–513, 2012.

[6] Zhaohui Luo. *Computation and reasoning*, volume 20. Oxford University Press, 1994.

[7] Zhaohui Luo. Dependent record types revisited. In *Proceedings of the 1st Workshop on Modules and Libraries for Proof Assistants*, pages 30–37. ACM, 2009.

[8] Zhaohui Luo, Sergei Soloviev, and Tao Xue. Coercive subtyping: theory and implementation. *Information and Computation*, 223, 2013.

[9] Per Martin-Löf. *Intuitionistic type theory*. Bibliopolis Naples, 1984.

[10] Benjamin C Pierce. Bounded quantification is undecidable. *Information and Computation*, 112(1), 1994.

[11] A. Ranta. *Type-Theoretical Grammar*. Oxford University Press, 1994.