Guarded cubical type theory

Lars Birkedal[1], Aleš Bizjak[1], Ranald Clouston[1], Hans Bugge Grathwohl[1], Bas Spitters[1], and Andrea Vezzosi[2]

[1] Department of Computer Science, Aarhus University
[2] Department of Computer Science and Engineering, Chalmers University of Technology

Guarded dependent type theory [1] is a dependent type theory with guarded recursive types, which are useful for building models of program logics, and as a tool for programming and reasoning with coinductive types. This is done via a modality $\triangleright$, pronounced 'later', with a constructor next, and a *guarded* fixed-point combinator $\mathsf{fix} : (\triangleright A \to A) \to A$. This combinator is used both to define guarded recursive types as fixed-points of functions on universes, as well as to define functions on these types.

Cubical type theory [2] is an extension of Martin-Löf type theory with the goal of obtaining a computational interpretation of the univalence axiom. The important novel idea in cubical type theory is that the identity type is not inductively defined. Instead the identity type[1] $\mathsf{Id}_A(x, y)$ is defined to be the type of *paths* starting from x to y. These paths are defined via an abstract interval $\mathbb{I}$ with endpoints $0, 1$. Elements of the identity type are then introduced using path abstraction: if t is a term of type A in context $\Gamma, i : \mathbb{I}$, then $\langle i \rangle\, t$ is a term of type $\mathsf{Id}_A(t[0/i], t[1/i])$ in context Γ. The elimination rule for the identity type is application: given an element i of $\mathbb{I}$ and a proof $p : \mathsf{Id}_A(x, y)$ the term $p\, i$ is of type A, with two judgemental equalities: $p\, 0 \equiv x$ and $p\, 1 \equiv y$. Several extensionality properties are derivable in this type theory. In particular, function extensionality is provable with the term

$$\mathsf{funext} = \lambda f g p.\, \langle i \rangle\, \lambda a.(p a)\, i : \prod_{f g : A \to B} \prod_{x : A} (\mathsf{Id}_B(f x, g x) \to \mathsf{Id}_{A \to B}(f, g)). \tag{1}$$

It is hoped that this type theory has decidable type-checking and satisfies canonicity.

We propose *guarded cubical type theory*, a combination of the two type theories with the goal of obtaining a type theory with a later modality and guarded fixed-point combinator that has decidable type-checking and satisfies canonicity. In previous work on guarded dependent type theory the focus was on designing the rules of the type theory so that it is possible to work with guarded recursive types. In particular, the type theory in [1] is an extensional type theory, i.e., there is the equality reflection rule for the identity type, and there is the judgemental equality

$$\mathsf{fix}\, f \equiv f(\mathsf{next}(\mathsf{fix}\, f)). \tag{2}$$

Both of these prevent decidable type-checking. Moreover, the type theory in [1] also includes

$$\mathsf{com} : \triangleright \mathsf{Id}_A(x, y) \to \mathsf{Id}_{\triangleright A}(\mathsf{next}\, x, \mathsf{next}\, y)$$

which is the inverse to the canonical term of type $\mathsf{Id}_{\triangleright A}(\mathsf{next}\, x, \mathsf{next}\, y) \to \triangleright \mathsf{Id}_A(x, y)$. We found that com is crucial for proving properties of guarded recursive types. Such a term com is not definable in Martin-Löf type theory when the identity type is defined in the usual way and so we need to introduce it axiomatically, which leads to the loss of canonicity. Note the formal resemblance of the type of com to the type of the axiom of function extensionality in (1).

[1] We do not distinguish between the types Id and Path in the interest of presentation.

Guarded cubical type theory Using the constructs from cubical type theory we can address both of these difficulties. First, because the identity type has a much more flexible introduction rule, the term com becomes definable as

$$\mathsf{com} = \lambda\,(p : \triangleright\,\mathsf{Id}_A\,(x, y))\,.\,\langle i\rangle\,\mathsf{next}\,[p' \leftarrow p]\,.p'i,$$

which should be compared to the term funext in (1). Here $[p' \leftarrow p]$ is a *delayed substitution*. Delayed substitutions, introduced in [1], are a generalisation of the applicative functor structure of $\triangleright$ to dependent types.

Interestingly, and rather surprisingly, we can utilise the *face lattice* $\mathbb{F}$ of cubical type theory to control fixed-point unfolding. We omit the fixed-point unfolding rule (2) from the type theory and instead decorate the fixed-point with a face which specifies when the fixed-point should be unfolded. The typing rule for fix becomes (omitting some details for this abstract):

$$\frac{\Gamma \vdash t : \triangleright A \to A \qquad \Gamma \vdash \phi : \mathbb{F}}{\Gamma \vdash \mathsf{fix}^{\phi}\,t}$$

with the only judgemental equality rule being $\mathsf{fix}^{1_{\mathbb{F}}}\,t \equiv f\,\left(\mathsf{next}\,\left(\mathsf{fix}^{0_{\mathbb{F}}}\,t\right)\right)$, where $1_{\mathbb{F}}$ and $0_{\mathbb{F}}$ are the top and bottom elements of $\mathbb{F}$. This judgemental equality rule allows us to unfold the fixed-point on demand. The face ϕ associated to the fixed-point can then be used to prove

$$\langle i\rangle\,\mathsf{fix}^{i=1}\,f : \mathsf{Id}_A\,\left(\mathsf{fix}^{0_{\mathbb{F}}}\,f, f\,\left(\mathsf{next}\,\left(\mathsf{fix}^{0_{\mathbb{F}}}\,f\right)\right)\right).$$

Prototype implementation Our prototype implementation of the type theory[2] extends the cubical type checker[3] with $\triangleright$ modality and guarded fixed-points. One of the motivations behind cubical type theory is to have some extensionality, while preserving the strong metatheoretic properties of intensional type theory. Our experiments with the prototype implementation show that the examples described in [1] can indeed be expressed in guarded cubical type theory, albeit with some more manual rewriting, since fixed-point unfolding is no longer a judgemental equality, but only a propositional one. This is a confirmation that the cubical type theory is relevant not only for mathematical applications, but also in areas of computer science.

Semantics We are working on a semantics of guarded cubical type theory based on an axiomatic version of the cubical model in the internal logic of cubical sets [3, 4]. Our axioms include a presheaf topos with an internal De Morgan algebra having the disjunction property and an internal operator $\forall$. The verification that these axioms actually suffice is nearly done.

The concrete model we use is based on presheaves over the product of the category of cubes (as used in the model of cubical type theory) and the preorder ω (as used in the model of guarded dependent type theory), and this presheaf topos satisfies our axioms.

References

[1] Aleš Bizjak, Hans Bugge Grathwohl, Ranald Clouston, Rasmus Ejlers Møgelberg, and Lars Birkedal. Guarded dependent type theory with coinductive types. In *FoSSaCS 2016*, 2016.

[2] Cyril Cohen, Thierry Coquand, Simon Huber, and Anders Mörtberg. Cubical type theory: a constructive interpretation of the univalence axiom. Unpublished, 2016.

[3] Thierry Coquand. Internal version of the uniform Kan filling condition. Unpublished, 2015.

[4] Bas Spitters. Cubical sets as a classifying topos. *TYPES'15*, 2015.

[2] https://github.com/hansbugge/cubicaltt/tree/gcubical

[3] https://github.com/mortberg/cubicaltt