Efficient Evaluation for Cubical Type Theories

András Kovács

Eötvös Loránd University
kovacsandras@inf.elte.hu

Currently, cubical type theories are the only known systems which support computational univalence. We can use computation in these systems to shortcut some proofs, by appealing to definitional equality of sides of equations. However, efficiency issues in existing implementations often preclude such computational proofs, or it takes a large amount of effort to find definitions which are feasible to compute. In this abstract we investigate the efficiency of the ABCFHL [ABC+21] Cartesian cubical type theory with separate homogeneous composition (hcom) and coercion (coe), although most of our findings transfer to other systems.

Cubical normalization-by-evaluation

In variants of non-cubical Martin-Löf type theory, definitional equalities are specified by reference to a substitution operation on terms. However, well-known efficient implementations do not actually use term substitution. Instead, *normalization-by-evaluation* (NbE) is used, which corresponds to certain *environment machines* from a more operational point of view. In these setups, there is a distinction between syntactic terms and semantic values. Terms are viewed as immutable program code that supports evaluation into the semantic domain but no other operations.

In contrast, in cubical type theories interval substitution is an essential component of computation which seemingly cannot be removed from the semantics. Most existing implementations use NbE for ordinary non-cubical computation, but also include interval substitution as an operation that acts on semantic values. Unfortunately, a naive combination of NbE and interval substitution performs poorly, as it destroys the implicit sharing of work and structure which underlies the efficiency of NbE in the first place. We propose a restructured cubical NbE which handles interval substitution more gracefully. The basic operations are the following.

1. *Evaluation* maps from syntax to semantics like before, but it additionally takes as input an interval environment and a cofibration.
2. *Interval substitution* acts on values, but it has trivial cost by itself; it only shallowly stores an explicit substitution.
3. *Forcing* computes a value to weak head form by sufficiently computing previously stored delayed substitutions.

On canonical values, forcing simply pushes substitutions further down, incurring minimal cost. But on neutral values, since neutrals are not stable under substitution, forcing has to potentially perform arbitrary computation. Here we take a hint from the formal cubical NbE by Sterling and Angiuli [SA21], by annotating neutrals by stability information. This allows us to quickly determine whether a neutral is stable under a given substitution. When a neutral value is stable, forcing does not have to look inside it.

It turns out that there is only a single computation rule in the ABCFHL theory which can trigger interval substitution with significant cost: the coercion rule for the Glue type former. In every other case, only a *weakening* substitution may be created, but all neutral values are stable under weakening, so forcing by weakening always has trivial cost.

Using canonicity in closed evaluation

In non-cubical type theories, evaluation of closed terms can be more efficient than that of open terms. For instance, when we evaluate an $\mathsf{if}-\mathsf{then}-\mathsf{else}$ expression, we know that exactly one branch will be taken. In open evaluation, the Bool scrutinee may be neutral, in which case both branches may have to be evaluated.

In the cubical setting, *systems of partial values* can be viewed as branching structures which make case distinctions on cofibrations. Importantly, there are computation rules which scrutinize *all* components of a cubical system. These are precisely the homogeneous composition rules (hcom) for strict inductive types. For example:

$$\mathsf{hcom}_{\mathbb{N}}^{r \to r'}\,[\psi \mapsto i.\,\mathsf{suc}\,t]\,(\mathsf{suc}\,b) = \mathsf{suc}\,(\mathsf{hcom}_{\mathbb{N}}^{r \to r'}\,[\psi \mapsto i.\,t]\,b)$$

When we only have interval variables and a cofibration in the context, we do not have to compute every system component to check for suc. In this case, which we may call "closed cubical", we can use the canonicity property of the theory. Here $\mathsf{suc}\,b$ in the hcom base implies that every system component is headed by suc as well. Hence, we can use the following rule instead:

$$\mathsf{hcom}_{\mathbb{N}}^{r \to r'}\,[\psi \mapsto i.\,t]\,(\mathsf{suc}\,b) = \mathsf{suc}\,(\mathsf{hcom}_{\mathbb{N}}^{r \to r'}\,[\psi \mapsto i.\,\mathsf{pred}\,t]\,b)$$

Here, pred is a metatheoretic function which takes the predecessor of a value which is already known to be definitionally suc. The revised rule assumes nothing about the shape of t on the left hand side, so we can compute pred lazily in the output. These lazy projections work analogously for all non-higher inductive types. For higher-inductive types, hcom is a canonical value, so there is no efficiency issue to begin with.

Summary

- Costly interval substitution can only arise from computing with Glue types.
- In closed cubical evaluation, no computation rule forces all components of a system.

We are optimistic that an implementation with these properties would yield significant performance improvements. We are currently in the process of developing this system and adapting existing benchmarks to it.

References

[ABC+21] Carlo Angiuli, Guillaume Brunerie, Thierry Coquand, Robert Harper, Kuen-Bang Hou (Favonia), and Daniel R. Licata. Syntax and models of cartesian cubical type theory. *Math. Struct. Comput. Sci.*, 31(4):424–468, 2021. doi:10.1017/S0960129521000347.

[SA21] Jonathan Sterling and Carlo Angiuli. Normalization for cubical type theory. In *36th Annual ACM/IEEE Symposium on Logic in Computer Science, LICS 2021, Rome, Italy, June 29 - July 2, 2021*, pages 1–15. IEEE, 2021. doi:10.1109/LICS52264.2021.9470719.