Substitution systems revisited*

Benedikt Ahrens and Ralph Matthes

Institut de Recherche en Informatique de Toulouse, Université Paul Sabatier
{ahrens,matthes}@irit.fr

Abstract

Matthes and Uustalu (TCS 327(1–2):155–174, 2004) presented a categorical description of substitution systems capable of capturing syntax involving binding which is independent of whether the syntax is made up from least or greatest fixed points. We extend this work in two directions: we continue the analysis by creating more categorical structure, in particular by organizing substitution systems into a category and studying its properties, and we develop the proofs of the results of the cited paper and our new ones in UniMath, a recent library of univalent mathematics formalized in the Coq theorem prover.

In previous work, Matthes and Uustalu [6] define a notion of "heterogeneous substitution system", the purpose of which is to axiomatize substitution and its desired properties. Such a substitution system is given by an algebra of a signature functor, equipped with an operation—which is to be thought of as substitution—that is compatible with the algebra structure map in a suitable sense. The term "heterogeneous" refers to the fact that the underlying notion of signature encompasses variable binding constructions and also explicit substitution a. k. a. flattening. More precisely, the signature is based on a rank-2 functor H (an endofunctor on a category of endofunctors) for the respective domain-specific signature, to which a monadic unit is explicitly added. The latter corresponds to the inclusion of variables into the elements that are considered as terms (in a quite general sense) over their variable supplies. The name "rank-2 functor" stems from the rank of the type operator that transforms type transformations into type transformations—hence has kind $(\mathsf{Set} \to \mathsf{Set}) \to (\mathsf{Set} \to \mathsf{Set})$—which may be seen as backbone of H in case the base category is Set. In this rank-2 setting, the carrier of the algebra is an endofunctor, and since a monadic unit is already present, a natural question is if one obtains a monad. In that paper, it is then shown that for any heterogeneous substitution system this is indeed the case; multiplication of the monad is derived from the "substitution" operation which is parameterized by a morphism f of pointed endofunctors and consists in asking for a unique solution that makes a certain diagram commute. Monad multiplication and one of the monad laws is obtained from the existence of a solution in the case that f is the identity, while the other monad laws are derived from uniqueness for two other choices of f.

Furthermore, it is shown there that "substitution is for free" for both initial algebras as well as—maybe more surprisingly—for (the inverse of) final coalgebras: if the initial algebra, resp. terminal coalgebra, of a given signature functor exists, then it, resp. its inverse, can be augmented to a substitution system (for the former case, and in order to easily use generalized iteration [4], it is assumed that the functor $- \cdot Z$ has a right adjoint for every endofunctor Z). Indeed, it was one of the design goals of the axiomatic framework of heterogeneous substitution systems to be applicable to *non-wellfounded* syntax as well as to wellfounded syntax, whereas related work (e.g., [5, 2]) frequently only applies to wellfounded syntax.

Examples of substitution systems are thus given by the lambda calculus, with and without explicit flattening, but also by languages involving typing and *infinite* terms.

*The work of Benedikt Ahrens was partially supported by the CIMI (Centre International de Mathématiques et d'Informatique) Excellence program ANR-11-LABX-0040-CIMI within the program ANR-11-IDEX-0002-02 during a postdoctoral fellowship.

The goal of the present work is to extend and to formalize the work by Matthes and Uustalu [6]; in particular, we introduce a natural notion of *morphisms* of heterogeneous substitution systems, thus arranging them into a category. We then show that the construction of a monad from a heterogeneous substitution system from [6] extends functorially to morphisms.

Moreover, we prove that the substitution system obtained in [6] by equipping the initial algebra with a substitution operation, is initial in the corresponding category of substitution systems. This makes use of a general fusion law for generalized iteration [4]. However, we will show why a similar result cannot be expected for final coalgebras.

As an example of the usefulness of our results, we intend to express the resolution of explicit flattening of the lambda calculus as a(n initial) morphism of substitution systems.

We are in the process of formalizing our results in univalent foundations, more specifically basing ourselves on the UniMath library [1]. This basis of our formalization is suitable in that it provides extensionality (functional and propositional) in a natural way and hereby avoids the use of setoids that would otherwise be inevitable; indeed, since our results are not about categories *in abstracto* but use general categorical concepts in more concrete instances such as the endofunctor category over a given category or its extension by a "point", we need extensionality axioms for the instantiation. Functional extensionality is a consequence of the univalence axiom that we do not use directly in our formalization. We profit from the existing category theory library in UniMath. It is well-known that the handling of rank-2 functors needs universe polymorphism (otherwise, one would need a number of copies of the categorical concepts in the Coq vernacular, since, e. g., the endofunctor category has a higher universe level than its base category). In UniMath, this universe polymorphism is provided by switching off the control of universe levels (setting $\mathsf{Type} : \mathsf{Type}$). Still, this is not extensional type theory—there is a big difference between convertibility and equality. Already for the purpose of type-checking of statements taken from the paper version, we need to reformulate some laws to fit into the type-theoretic framework and to slightly reformulate the statements (in concrete instances, the differences would plainly disappear). The culprit here is that convertibility of types of natural transformations is not preserved under application of an abstract rank-2 functor H.

At the time of writing this abstract, we have not yet finished the formalization; what we do have is a full formalization of the main result of Matthes and Uustalu [6, Theorem 10] that yields a monad for every heterogeneous substitution system, and moreover our refinement stating that this gives rise to a faithful functor into the category of monads. The formalization can be consulted at [3]. We hope that we will have formalized the other new theoretical results before presenting this work at the TYPES conference.

References

[1] UniMath. `http://unimath.org`.

[2] Benedikt Ahrens. Extended Initiality for Typed Abstract Syntax. *Logical Methods in Computer Science*, 8(2):1 – 35, 2012.

[3] Benedikt Ahrens and Ralph Matthes. Substitution systems revisited: Formalization in Coq. `https://github.com/benediktahrens/substitution`.

[4] Richard S. Bird and Ross Paterson. Generalised folds for nested datatypes. *Formal Asp. Comput.*, 11(2):200–222, 1999.

[5] Marcelo Fiore, Gordon Plotkin, and Daniele Turi. Abstract syntax and variable binding. In *Proceedings of the IEEE Symposium on Logic in Computer Science*, LICS '99, pages 193–202, 1999.

[6] Ralph Matthes and Tarmo Uustalu. Substitution in non-wellfounded syntax with variable binding. *Theor. Comput. Sci.*, 327(1-2):155–174, 2004.