A Uniform Programming Language for Implementing XML Standards

Pavel Labath¹ and Joachim Niehren²

¹ Comenius University, Bratislava
labath@dcs.fmph.uniba.sk

² Inria, Lille

Abstract. We propose X-Fun, a core language for implementing various XML standards in a uniform manner. X-Fun is a higher-order functional programming language for transforming data trees based on node selection queries. It can support the XML data model and XPath queries as a special case. We present a lean operational semantics of X-Fun based on a typed lambda calculus that enables its in-memory implementation on top of any chosen path query evaluator. We also discuss compilers from XSLT, XQuery and XProc into X-Fun which cover the many details of these standardized languages. As a result, we obtain in-memory implementations of all these XML standards with large coverage and high efficiency in a uniform manner from Saxon’s XPath implementation.

Keywords: XML transformations, database queries, functional programming languages, compilers.

1 Introduction

A major drawback of query-based functional languages with data trees so far is that they either have low coverage in theory and practice or no lean operational semantics. Theory driven languages are often based on some kind of macro tree transducers [3,5,11], which have low coverage, in that they are not closed under function composition [4] and thus not Turing complete (for instance type checking is decidable [12]). The W3C standardised languages XQuery [13] and XSLT [7], in contrast, have large coverage in practice (string operations, data joins, arithmetics, aggregation, etc.) and in theory, since they are closed by function composition and indeed Turing complete [8]. The definitions of these standards, however, consist of hundreds of pages of informal descriptions. They neither explain how to build a compiler in a principled manner nor can they be used as a basis for formal analysis.

A second drawback is the tower of languages approach, adopted for standardised XML processing languages. What happened in the case of XML was the development of a separate language for each class of use cases, which all host the XPath language for querying data trees based on node navigation. XSLT serves for use cases with recursive document transformations such as HTML publishing, while XQuery was developed for use cases in which XML databases are

© Springer-Verlag Berlin Heidelberg 2015
queried. Since the combination of both is needed in most larger applications, the JSON pipeline language XPROC [16,17,18] was developed and standardised again by the W3C. This resulted in yet another functional programming language for processing data trees based on XPATH.

For resolving the above two drawbacks, the question is whether there exists a uniform core language for processing data trees that can cover the different XML standards in a principled manner. It should have a lean and formal operational semantics, support node selection queries as with XPATH and it should be sufficiently expressive in order to serve as a core language for implementing XQUERY, XSLT, and XPROC in a uniform manner.

Related work. An indicator for the existence of a uniform core language for XML processing is that the omnipresent Saxon system [14] implements XSLT and XQUERY on a common platform. However, there is no formal description of this platform as a programming language, and it does not support the XML pipeline language XPROC so far. Instead, the existing implementations of XPROC, CALABASH [16] and QuiXPROC [18], are based on Saxon’s XPATH engine directly.

The recent work from Castagna et al. [2] gives further hope that our question will find a positive answer. They present an XPATH-based functional programming language with a lean formal model based on the lambda calculus, which thus satisfies our first two conditions above and can serve as a core language for implementing a subset of XQUERY 3.0. We believe that relevant parts of XSLT and XPROC can also be compiled into this language, even though this is not shown there. The coverage, however, will remain limited, in particular on the XPATH core (priority is given to strengthening type systems). Therefore, our last requirement is not satisfied.

Contributions. In this paper, we present the first positive answer to the above question based on X-Fun. This is a new purely functional programming language. X-Fun is a higher-order language and it supports the evaluation of path-based queries that select nodes in data trees. The path queries are mapped to X-Fun expressions, whose values can be computed dynamically. In contrast to most previous interfaces between databases and programming languages, we overload variables of path queries with variables of X-Fun. In this manner, the variables in path queries are always bound to tree nodes, before the path query is evaluated itself. We note in particular, that path queries are not simply mapped to X-Fun expressions of type string.

The formal model of the operational semantics of X-Fun is a lambda calculus with a parallel call-by-value reduction strategy. Parallel evaluation is possible due to the absence of imperative data structures. The main novelty in X-Fun admission of tree nodes as values of type node. Which precise nodes are admitted depends on a tree store. New nodes can be created dynamically by adding new trees to the tree store. The same tree can be added twice to the store but with different nodes. How nodes are represented internally can be freely chosen by the X-Fun implementation and is hidden from the programmer.