A Purely Functional Computer Algebra System Embedded in Haskell

Hiromi Ishii

Doctoral Program in Mathematics,
University of Tsukuba, Tsukuba, Ibaraki 305-8571, Japan
h-ishii@math.tsukuba.ac.jp

Abstract. We demonstrate how methods in *Functional Programming* can be used to implement a computer algebra system. As a proof-of-concept, we present the `computational-algebra` package. It is a computer algebra system implemented as an embedded domain-specific language in *Haskell*, a purely functional programming language. Utilising methods in functional programming and prominent features of Haskell, this library achieves safety, composability, and correctness at the same time. To demonstrate the advantages of our approach, we have implemented advanced Gröbner basis algorithms, such as Faugère's F_4 and F_5, in a composable way.

Keywords: Gröbner basis; signature-based algorithms; computational algebra; functional programming; Haskell; type system; formal methods; property-based testing; implementation report.

1 Introduction

In the last few decades, the area of computational algebra has grown larger. Many algorithms have been proposed, and there have emerged plenty of computer algebra systems. Such systems must achieve *correctness*, *composability* and *safety* so that one can implement and examine new algorithms within them. More specifically, we want to achieve the following goals:

Composability means that users can easily implement algorithms or mathematical objects so that they work seamlessly with existing features.

Safety prevents users and implementors from writing "wrong" code. For example, elements in different rings, e.g. $\mathbb{Q}[x, y, z]$ and $\mathbb{Q}[w, x, y]$, should be treated differently and must not directly be added. Also, it is convenient to have handy ways to convert, inject, or coerce such values.

Correctness of algorithms, with respect to prescribed formal specifications, should be guaranteed with a high assurance.

We apply methods in the area of *functional programming* to achieve these goals. As a proof-of-concept, we present the `computational-algebra` package [12]. It is implemented as an embedded domain-specific language in the