Towards Specifying Symbolic Computation*

Jacques Carette and William M. Farmer

Computing and Software, McMaster University, Canada
http://www.cas.mcmaster.ca/~carette
http://imps.mcmaster.ca/wmfarmer

Abstract. ??

1 Introduction

2 Rational Expressions, Rational Functions

2.1 The Problem

Let e be an expression in the language $\mathcal{L}$ of the field $\mathbb{Q}(x)$, that is, a well-formed expression built from the symbols $x, 0, 1, +, *, -, ^{-1}$, elements of $\mathbb{Q}$ and parentheses (as necessary). For greater readability, we will take the liberty of using fractional notation for $^{-1}$ and the exponential notation x^n for $x * \cdots * x$ (n times). e can be something simple like $\frac{x^4-1}{x^2-1}$ or something more complicated like

$$\frac{\frac{1-x}{3/2x^{18}+x+17}}{\frac{1}{9834*x^{19393874}-1/5}} + 3 * x - \frac{12}{x}.$$

The meaning of e, written $[\![e]\!]_{\mathbb{Q}(x)}$, is the element of $\mathbb{Q}(x)$ that e denotes. If e is an expression like $x/0$, then the meaning of e is undefined and e is said to be *undefined*. We will call members of $\mathcal{L}$ *rational expressions*. Two rational expressions e and e' are *equal*, written $e = e'$, if they are both defined and $[\![e]\!]_{\mathbb{Q}(x)} = [\![e']\!]_{\mathbb{Q}(x)}$ and are *quasi-equal*, written $e \simeq e'$, if either $e = e'$ or they are both undefined.

We are taught that, like for members of $\mathbb{Q}$ (such as $5/15$), there is a *normal form* for rational expressions. This is typically defined to be an expression p/q for two polynomials $p, q \in \mathbb{Q}[x] \subseteq \mathbb{Q}(x)$ such that p and q are themselves in polynomial normal form and $\mathsf{gcd}(p, q) = 1$. The motivation for the latter property is that we usually want to write $\frac{x^4-1}{x^2-1}$ as $x^2 + 1$ just as we usually want to write $5/15$ as $1/3$. Thus, the normal forms of $\frac{x^4-1}{x^2-1}$ and $\frac{x}{x}$ are $x^2 + 1$ and 1, respectively. This definition of normal form is based on the characteristic that the elements of the *field of fractions* of a ring R can be written as quotients r/s of elements of R where $r_0/s_0 = r_1/s_1$ if and only if $r_0 * s_1 = r_1 * s_0$ in R.

* This research is supported by NSERC.

SBMA should be defined in the Introduction.

Every computer algebra system implements a function that *normalizes* expressions that denote elements of $\mathbb{Q}$, $\mathbb{Q}[x]$, and $\mathbb{Q}(x)$. Let $\mathsf{normRatExpr}$ be the name of an algorithm that implements this function for $\mathbb{Q}(x)$. It should certainly have the signature $\mathsf{normRatExpr} : \mathcal{L} \to \mathcal{L}$ and satisfy the invariant $\mathsf{normRatExpr}(e) \simeq e$ — in other words, be *meaning preserving* — for all $e \in \mathcal{L}$. $\mathsf{normRatExpr}$ is an example of an SBMA. Note that merely giving the signature of $\mathsf{normRatExpr}$ and saying that it is meaning preserving is a (tremendously) incomplete specification of the *computational behavior* of $\mathsf{normRatExpr}$.

A rational expression $e \in \mathbb{Q}(x)$ can be interpreted as a *function* $f = \lambda\, x : \mathbb{Q}\ .\ e$. Such functions are typically called *rational functions*. However, equality in $\mathbb{Q}(x)$ and in $\mathbb{Q} \to \mathbb{Q}$ differ. For example, one might think that the rational functions $\lambda\, x : \mathbb{Q}\ .\ x/x$ and $\lambda\, x : \mathbb{Q}\ .\ 1$ should be equal since x/x and 1 are equal as rational expressions, or that $\lambda\, x : \mathbb{Q}\ .\ 1/x - 1/x$ and $\lambda\, x : \mathbb{Q}\ .\ 0$ are too. But they are not since both $\lambda\, x : \mathbb{Q}\ .\ x/x$ and $\lambda\, x : \mathbb{Q}\ .\ 1/x - 1/x$ are undefined at 0, while both $\lambda\, x : \mathbb{Q}\ .\ 1$ and $\lambda\, x : \mathbb{Q}\ .\ 0$ are defined everywhere. Calling $\lambda\, x : \mathbb{Q}\ .\ x/x$ and $\lambda\, x : \mathbb{Q}\ .\ 1$ is frequently justified by invoking the concept of *removable singulaties* – but this reasoning is less clear as a justification that $\lambda\, x : \mathbb{Q}\ .\ 1/x - 1/x$ and $\lambda\, x : \mathbb{Q}\ .\ 0$ are equal.

Why is this an issue? Mainly because CAS make little distinction between the two. For example, one can always *evaluate* an expression for its free variables, or even convert it to a function. In Maple[1], these are done respectively via `eval(e, x = 0)` and *unapply(e, x)*. We can exhibit the problematic behaviour as follows:

insert some Maple code with output here

In fact, there is an even more pervasive, one could even say *obnoxious* way of doing this: as the underlying language is *imperative*, it is possible to do

```
e := (x^4-1)/(x^2-1);
# many, many more lines of 'code'
x := 1;
# try to use 'e'
```

Hence, if an expression e is interpreted as a function, then it is not valid to simplify the function by applying $\mathsf{normRatExpr}$ to e, but CASs let the user do exactly this because usually there is no distinction made between e as a rational expression and e as representing a rational function, as shown above. We need to know the *mathematical meaning* of $\mathsf{normRatExpr}$ applied to rational functions to be able to avoid unsound applications of $\mathsf{normRatExpr}$.

I don't know why we need to say this: Of course, given some symbol y, $f(y)$ is in $\mathcal{L}$.

We are thus interested in the following questions:

1. What should the specification of the computational behavior of $\mathsf{normRatExpr}$ be?
2. What is the mathematical meaning of $\mathsf{normRatExpr}$ be when $\mathsf{normRatExpr}$ is applied to the body of a rational function?
3. What features of a logic are needed to express $\mathsf{normRatExpr}$'s specification and mathematical meaning?

[1] Mathematica has similar commands

4. What features of a logic would make expressing normRatExpr's specification and mathematical meaning relatively straightforward?

2.2 The Specification of normRatExpr

2.3 The Mathematical Meaning of normRatExpr

3 Conclusion

Todo list