Normal Forms in Function Fields

Karl Aberer,

ETH Zürich

Abstract

We consider function fields of functions of one variable augmented by the binary operation of composition of functions. It is shown that the straightforward axiomatization of this concept allows the introduction of a normal form for expressions denoting elements in such fields. While the description of this normal form seems relatively intuitive, it is surprisingly difficult to prove this fact. We present an algorithm for the normalization of expressions, formulated in the symbolic computer algebra language *mathematica*. This allows us to effectively decide compositional identities in such fields. Examples are given.

1 Introduction

One of the fundamental questions in symbolic computation is to find normal forms for expressions where certain identities are considered to be valid for these expressions. Function fields are structures appearing frequently as domains for symbolic computations, one has to think only of the rational functions, elementary functions, special functions etc. . Such function fields satisfy, besides the identities resulting from their property of being a field, additional identities resulting from the properties of the composition operator. These properties are known very well, and important contributions to the investigation of spaces equipped with composition as an additional operation were especially made by K. Menger [6]. It now seems natural to ask for normal form algorithms in such spaces, in our case function fields, to be able e.g. to computationally decide the equivalence problem for terms. In the following example let a, F, j be unary function symbols, where additionally $j(x) = x$, for all x. It is quite straightforward to simplify the term

$$((F+1)(1+a((j+1)^2)))(x-1)$$

to

$$F(1+a(x^2))+1.$$

But first how do we know that this is the simplest or an otherwise outstanding form of that term and second what to do when the terms get longer and computation by hand gets error prone or even impracticable? For example: does the following expression have a simple equivalent "normal" form:

$$\begin{aligned}&(((F+1)(1+a((1+j)^2)))(a-1)+\\&\quad j\cdot(1+F(1+a(a^2))))(F(a+a^2)(x^2-1)-\\&\quad (F+1)(a(j^2-1)(x)\cdot(1+a(j\cdot(2+j))(x-1)))).\end{aligned}$$

(with multiplications indicated by $\cdot$.) When expressions get a certain size, and as we will see the normal forms can grow exponential in length, the help of computers is unavoidable. With the help of the computer algebra system *mathematica* we will be able to implement the normal forms for expressions like the ones given above. The goal of the algorithm is to reduce every term to a simple form of which we can show by construction of an appropriate substitution of the function symbols that it is not identical to zero iff the expression itself is nonzero. This form is the desired normal form.

This work is part of a project to develop a system to compute in combinatory analytic structures and languages, e.g. fields or vector spaces with operators appropriate to represent programs. For a short motivation compare with section 6 where the terms of the first of the two examples above are interpreted as programs. Extensions of our algorithm to the case where differentiation (differential fields) or programming constructs like $if..then..else$ are involved were made [1] and are used as a part of a system to compute in combinatory differential fields [3].

0

2 Definitions

First we give a general definition of the notion of normal form. Let T be a set of terms and $=_T$ an equivalence relation on the terms.

Definition 1 (Normal form) *A normal form is given by a normal form function*

$$N : T \to NT \subseteq T, \quad \tau \mapsto N(\tau),$$

with the properties

1. $N(N(\tau)) \equiv N(\tau)$,
2. $a =_T b$ *iff* $N(a) \equiv N(b)$.

The relation $\equiv$ means syntactically equal. The set NT is the set of terms in normal form.

Next we define a language $\mathcal{C}$ in which we describe the properties of function fields and for whose terms we intend to construct normal forms. The terms are built up from the individual constants 0, 1 and ι, variables $x_1, x_2, x_3 \ldots$, the field operations $+$, $\cdot$, $-$, $^{-1}$, and the composition operation $\circ$. The predicates are $=$ and *const*. We characterize the structure of function fields by the following first order theory formulated in the language $\mathcal{C}$, using $\sigma, \tau, \ldots$ as metavariables for terms.

Definition 2 (Field axioms A_R) *A function field is a field of characteristic 0 with neutral element 0 for addition and 1 for multiplication. A subset of elements x with const(x) true are called the constant elements and form a subfield.*

$$const(0),\ const(1),$$
$$const(\sigma) \wedge const(\tau) \to const(\sigma + \tau) \wedge const(\sigma \cdot \tau),$$
$$const(\tau) \to const(-\tau),$$
$$const(\tau) \wedge \tau \neq 0 \to const(\tau^{-1})$$

Definition 3 (Axioms of composition A_C)

$$\begin{aligned}
(\tau_1 + \tau_2) \circ \sigma &= \tau_1 \circ \sigma + \tau_2 \circ \sigma,\\
(\tau_1 \cdot \tau_2) \circ \sigma &= \tau_1 \circ \sigma \cdot \tau_2 \circ \sigma,\\
(-\tau) \circ \sigma &= -(\tau \circ \sigma),\\
(\tau^{-1}) \circ \sigma &= (\tau \circ \sigma)^{-1},\\
\tau_1 \circ (\tau_2 \circ \tau_3) &= (\tau_1 \circ \tau_2) \circ \tau_3,\\
\tau \circ \iota &= \iota \circ \tau = \tau,\\
const(\sigma) \to \sigma \circ \tau &= \sigma.
\end{aligned}$$

We introduce different classes of terms.

Definition 4 (Term classes)

(1) Constant Terms Q: Terms built up with the constants 0, 1 and operations $+$, $\cdot$, $-$, $^{-1}$ containing no variables.

(2) Rational Terms $R(\sigma_1, \ldots, \sigma_k)$: Terms built up with the constants 0, 1 and operations $+$, $\cdot$, $-$, $^{-1}$ using terms $\sigma_1, \ldots, \sigma_k$.

(3) Composition Terms $C(x_1, \ldots, x_k)$: Terms built up with the constants 0, 1, ι and operations $+$, $\cdot$, $-$, $^{-1}$, $\circ$ using variables $x_1, \ldots, x_k$.

Example 1

$$R(\iota, x) \subseteq C(x).$$
$$Q \subseteq R(x, x \circ x, x \circ (x^2), \ldots) \subseteq C(x).$$

On these classes we define the following equivalence relations, where we use "$\vdash$" to denote provability in first-order logic with equality.

Definition 5 (Equivalence relations on terms)

(1) $\tau =_R \sigma$ for $\tau, \sigma \in C(x_1, x_2, \ldots)$ iff $A_R \vdash \tau = \sigma$.

(2) $\tau =_C \sigma$ for $\tau, \sigma \in C(x_1, x_2, \ldots)$ iff $A_R \cup A_C \vdash \tau = \sigma$.

Remark. Hence we have $\tau \neq_R \sigma$ iff $A_R \nvdash \tau = \sigma$ and $\tau \neq_C \sigma$ iff $A_R \cup A_C \nvdash \tau = \sigma$. Our goal will be to construct a normal form function N_C under the equivalence relation $=_C$.

3 Rational Terms

The obvious normal form for Q under $=_R$ is given by:

$$\begin{aligned}
NQ = \{\tfrac{p}{q} \mid\ & p \equiv \pm(1 + 1 + \ldots + 1),\\
& q \equiv 1 + 1 + \ldots + 1,\ gcd(p, q) = 1,\ p, q \in \mathbb{Z}\}.
\end{aligned}$$

Similarly for rational terms in the class $R(x_1, \ldots, x_k)$ a class of normal forms under $=_R$ (with NP the class of normal forms of polynomials over $\mathbb{Z}$, i.e. normal forms for elements in $\mathbb{Z}[x_1, \ldots, x_k]$) is given by:

$$\begin{aligned}
NR(x_1, \ldots, x_k) = \{\tfrac{p}{q} \mid\ & p, q \in NP(x_1, \ldots, x_k),\\
& lc(q) > 0,\ gcd(p, q) = 1\}.
\end{aligned}$$

Normal forms for polynomials are trivial, one could take the polynomial expanded with equal powers collected and sorted in a canonical order. We want to summarize in the following lemma.

Lemma 1 *For the class Q there exists a normal form function N_Q which computes the unique normal form for terms in Q under the equivalence relation $=_R$. For the class $R(x_1, \ldots, x_k)$ there exists a normal form function N_R which computes the unique normal form for terms in $R(x_1, \ldots, x_k)$ under the equivalence relation $=_R$.*

4 Composition Terms

Remark. To increase the readability we want to use in the following lower case latin letters for rational terms and lower case greek letters for terms in C. Nevertheless the type of a term always will be obvious from the context. In the proofs we therefore often suppress argument lists.

Lemma 2 *For terms $\tau \in C$ and $t(\iota, x_1, \ldots, x_k) \in R(\iota, x_1, \ldots, x_k)$ the following equation holds.*

$$t(\iota, x_1, \ldots, x_k) \circ \tau =_C t(\tau, x_1 \circ \tau, \ldots, x_k \circ \tau).$$

Proof. Induction on the structure of the term t. ∎

Remark. We will use the following obvious properties:

(1) If $\sigma =_R \tau$ then $\sigma =_C \tau$ and the contraposition, if $\sigma \neq_C \tau$ then $\sigma \neq_R \tau$.

(2) The converse we can only state for $\sigma \in Q$. If $\sigma \in Q$ and $\sigma \neq_R 0$ then $\sigma \neq_C 0$. This property follows immediately from the field axiom $\neg(1 = 0)$ and the fact that all constant terms in Q have a normal form in NQ.

The first non-trivial problem is to specify a subclass of composition terms that will serve as normal forms. We propose the following definition.

Definition 6 (Classes of composition terms) *Let $V(x_1, \ldots, x_k)$ and $T(x_1, \ldots, x_k)$ be defined recursively as follows.*

$$\begin{aligned}
V_0(x_1, \ldots, x_k) &:= \{x_1, \ldots, x_k\},\\
T_n(x_1, \ldots, x_k) &:= \{t(\iota, v_1, \ldots, v_l) \mid \\
&\qquad v_1, \ldots, v_l \in \bigcup_{j \leq n} V_j(x_1, \ldots, x_k),\\
&\qquad t \in NR(\iota, v_1, \ldots, v_l)\},\\
V_{n+1}(x_1, \ldots, x_k) &:= \{x_i \circ \tau \mid i = 1, \ldots, k,\\
&\qquad \tau \in T_n(x_1, \ldots, x_k),\ \tau \not\equiv \iota\},\\
V(x_1, \ldots, x_k) &:= \bigcup_{j \in \mathbb{N}} V_j(x_1, \ldots, x_k),\\
T(x_1, \ldots, x_k) &:= \bigcup_{j \in \mathbb{N}} T_j(x_1, \ldots, x_k).
\end{aligned}$$

Remark. We have $T_0 \subset T_1 \subset T_2 \ldots$ and $V_0 \subset V_1 \subset V_2 \ldots$.

Lemma 3 *If $v \in V_n(x_1, \ldots, x_k)$ and $\tau \in T_m(x_1, \ldots, x_k)$, $\tau \not\equiv \iota$, then there exists a term $u \in V_{n+m+1}(x_1, \ldots, x_k)$ such that $v \circ \tau =_C u$.*

Proof. Induction on the structure of v.
Base step. Let $v \in V_0$, $v \equiv x_i$. Then take $u :\equiv x_i \circ \tau \in V_{m+1}$ for $\tau \in T_m$.

Induction step. Let $v \in V_n$, $n > 0$, $v \equiv x_i \circ \rho \equiv x_i \circ r(\iota, v_1, \ldots, v_h)$ with $r \in NR(\iota, v_1, \ldots, v_h)$ and $\rho \in T_{n-1}$. Then $v_1, \ldots, v_h \in \bigcup_{j \leq n-1} V_j$. Furthermore let $\tau \in T_m$, $\tau \equiv s(\iota, w_1, \ldots, w_l) \in NR(\iota, w_1, \ldots, w_l)$. Then again $w_1, \ldots, w_l \in \bigcup_{j \leq m} V_j$. We observe $v \circ \tau =_C (x_i \circ r) \circ \tau =_C x_i \circ (r \circ \tau)$. By application of lemma 2 we get

$$\begin{aligned}
v \circ \tau =_C\ & x_i \circ (r(\iota, v_1, \ldots, v_h) \circ \tau) =_C\\
& x_i \circ r(\tau, v_1 \circ \tau, \ldots, v_h \circ \tau) =_C\\
& x_i \circ r(s(\iota, w_1, \ldots, w_l), v_1 \circ \tau, \ldots, v_h \circ \tau) =_C\\
& x_i \circ t(\iota, w_1, \ldots, w_l, v_1 \circ \tau, \ldots, v_h \circ \tau),
\end{aligned}$$

where $t \in R(\iota, w_1, \ldots, w_l, v_1 \circ \tau, \ldots, v_h \circ \tau)$. We take $u :\equiv x_i \circ N_R(t)$ and then $u \in V_{n+m+1}$ because $w_1, \ldots, w_l \in \bigcup_{j \leq m} V_j$ and $v_1 \circ \tau, \ldots, v_h \circ \tau \in \bigcup_{j \leq n+m} V_j$, by induction hypothesis. ∎

Lemma 4 *There exists a function*

$$N_C : C(x_1, \ldots, x_k) \to T(x_1, \ldots, x_k)$$

such that for every composition term $\tau \in C(x_1, \ldots, x_k)$ we have $N_C(\tau) =_C \tau$.

Proof. Induction on the structure of the term τ.
Base step. For $\tau \equiv 0, 1, \iota, x_i$ this is trivial because then $\tau \in T_0 \subseteq T$ and we take $N_C(\tau) :\equiv \tau$.
Induction step, by cases according to the type of operations used:
Case 1. Field operations; let $\tau \equiv \tau_1 + \tau_2$. Then $N_C(\tau_1) \in T_{n_1}$ and $N_C(\tau_2) \in T_{n_2}$. Take

$$N_C(\tau) :\equiv N_R(N_C(\tau_1) + N_C(\tau_2)) \in T_{max(n_1, n_2)};$$

similarly for the other field operations.
Case 2. Composition operator; let $\tau \equiv \tau_1 \circ \tau_2$.
Case 2.1. For the special case $\tau_2 \equiv \iota$ we have $\tau_1 \circ \tau_2 =_C N_C(\tau_1) \equiv: N_C(\tau) \in T$.
Case 2.2. For $\tau_2 \not\equiv \iota$ we have

$$N_C(\tau_1) \equiv r(\iota, v_1, \ldots, v_h) \in NR(\iota, v_1, \ldots, v_h) \subseteq T_n,$$

and

$$N_C(\tau_2) \equiv s(\iota, w_1, \ldots, w_l) \in NR(\iota, w_1, \ldots, w_l) \subseteq T_m.$$

Then we get

$$\begin{aligned}
& N_C(\tau_1) \circ N_C(\tau_2) =_C\\
& r(\iota, v_1, \ldots, v_h) \circ N_C(\tau_2) =_C\\
& r(N_C(\tau_2), v_1 \circ N_C(\tau_2), \ldots, v_h \circ N_C(\tau_2)) =_C\\
& r(s(\iota, w_1, \ldots, w_l), v_1 \circ N_C(\tau_2), \ldots, v_h \circ N_C(\tau_2)) =_C\\
& t(\iota, w_1, \ldots, w_l, v_1 \circ N_C(\tau_2), \ldots, v_h \circ N_C(\tau_2)).
\end{aligned}$$

By lemma 3 we see that $v_1 \circ N_C(\tau_2), \ldots, v_h \circ N_C(\tau_2) \in V_{n+m+1}$.

Hence if we take, $N_C(\tau) :\equiv$
$NR(t(\iota, w_1, \ldots, w_l, v_1 \circ N_C(\tau_2), \ldots, v_n \circ N_C(\tau_2)))$
we get $N_C(\tau) \in T_{n+m+1}$. ∎

Next we want to show, for proving the main theorem later, that $\tau \neq_C 0$ follows from $\tau \neq_R 0$ for all elements $\tau \in T$. The following examples indicate that we need to construct for every term $\tau \in T$ a term $q \in R(\iota)$ and a constant $a \in Q$ such that $\tau|^q_x \circ a \neq_C 0$. They also show that this is not obvious.

Example 2

(1) For $v \in V$ we have $v \equiv x_i \circ \tau$ or $v \equiv x_i$. In both cases we get by substituting $c \in Q,\ c \neq_R 0$, for x_i the desired property $v|^c_{x_i} \neq_C c \neq_C 0$.

(2) For every term $\tau \in T$ of the form $\tau \equiv t(x_1 \circ \tau_1, \ldots, x_k \circ \tau_k)$, where $t \neq_R 0$ we get $\tau \neq_C 0$ by substituting suitable constants $c_1, \ldots, c_k$ for $x_1, \ldots, x_k$.

(3) For $\tau \equiv x \circ \tau_1 - x \circ \tau_2$ we cannot show that $\tau \neq_C 0$ by simply substituting a constant for x.

(4) Furthermore, in the case of the term $\tau \equiv (x \circ 1 - 1) \cdot (x \circ x - x \circ 1)$, we have $\tau|^q_x =_C 0$ for $q \equiv c \in Q$ or $q(\iota) \equiv \iota^n,\ n \in \mathbb{N}$.

Lemma 5 *For a term $\tau \in T(x)$ of the form $\tau \equiv t(\iota, v_1, \ldots, v_l),\ t \in NR(\iota, v_1, \ldots, v_l)$ with $v_1, \ldots, v_l \in V(x)$ we have the property*

$$t(\iota, v_1, \ldots, v_l) \neq_C 0 \text{ iff } t(\iota, v_1, \ldots, v_l) \neq_R 0.$$

Proof. We construct for every $\tau \in T(x)$ a set of equations E_τ for $q \in R(\iota)$ and a constant $a \in Q$ with the following properties:

(1) if q satisfies the equations E_τ we have $\tau|^q_x \circ a \neq_C 0$;

(2) there always exists a polynomial $q \in R(\iota)$ satisfying the equations E_τ.

We introduce now an auxiliary notation for variables. Let α, β be fixed symbols. With these we construct sets of variable-symbols as follows:

$$\begin{aligned} B_0 &:= \{\alpha\}, \\ B_n &:= \{\beta_i \mid i \equiv t(\beta_{i_1}, \ldots, \beta_{i_l}), \\ &\quad t \in NR(\beta_{i_1}, \ldots, \beta_{i_l}),\ \beta_{i_j} \in B_{n-1}, j = 1, \ldots, l\}, \\ B &:= \textstyle\bigcup_{n \in \mathbb{N}} B_n. \end{aligned}$$

Let $\tau \in T(x),\ \tau = t(\iota, v_1, \ldots, v_l) \in NR(\iota, v_1, \ldots, v_l)$ with $t \neq_R 0$. There exists a finite set of equations E_τ for q, a set of variables $V_\tau \subseteq B$, a set of terms $I_\tau \subseteq NR(V_\tau)$, and a term $j_\tau \in NR(V_\tau)$ with the following properties.

(1) The equations of E_τ are of the form $\beta_i =_C q \circ i$, with $i \in NR(V_\tau)$ and $\beta_i \in V_\tau$.

(2) For all $i \in NR(V_\tau)$, appearing as argument of q on the r.h.s. of an equation in E_τ, we have $i \in I_\tau$.

(3) For j_τ we have $j_\tau \notin I_\tau$, $j_\tau \in NR(V_\tau)$ and $j_\tau \neq_R 0$. Furthermore the following equation holds if all equations of E_τ are satisfied.

$$j_\tau =_C \tau|^q_x \circ \alpha.$$

We construct E_τ, V_τ, I_τ and j_τ inductively.
Base step. In the case $\tau \equiv t(\iota, x) \in T_0$ take

$$E_\tau := \{\beta_\alpha =_C q \circ \alpha\}, \quad V_\tau := \{\alpha, \beta_\alpha\}$$

$$I_\tau := \{\alpha\}, \quad j_\tau :\equiv t(\alpha, \beta_\alpha).$$

We have to show (3).

$$t(\iota, x)|^q_x \circ \alpha =_C t(\alpha, q \circ \alpha) =_C t(\alpha, \beta_\alpha) =_C j_\tau.$$

For the case $\tau \equiv t(\iota)$ we have $E_\tau := \emptyset$, $V_\tau := \emptyset$, $I_\tau := \emptyset$ and $j_\tau :\equiv t(\alpha)$.
Induction step. For $\tau \equiv t(\iota, x \circ \tau_1, \ldots, x \circ \tau_l)$ with $\tau_1, \ldots, \tau_l \in T_n(x)$ we define

$$\begin{aligned} E_\tau &:= \textstyle\bigcup_{m=1}^l E_{\tau_m} \cup \bigcup_{m=1}^l \{\beta_{j_{\tau_m}} =_C q \circ j_{\tau_m}\}, \\ V_\tau &:= \textstyle\bigcup_{m=1}^l V_{\tau_m} \cup \bigcup_{m=1}^l \{\beta_{j_{\tau_m}}\}, \\ I_\tau &:= \textstyle\bigcup_{m=1}^l I_{\tau_m} \cup \bigcup_{m=1}^l \{j_{\tau_m}\}, \\ j_\tau &:\equiv t(\alpha, \beta_{j_{\tau_1}}, \ldots, \beta_{j_{\tau_l}}). \end{aligned}$$

(1) and (2) are now satisfied by this definition. We want to show (3).

$$\begin{aligned} \tau|^q_x \circ \alpha =_C\ & t(\iota, q \circ \tau_1|^q_x, \ldots, q \circ \tau_l|^q_x) \circ \alpha =_C \\ & t(\iota \circ \alpha, (q \circ \tau_1|^q_x) \circ \alpha, \ldots, (q \circ \tau_l|^q_x) \circ \alpha) =_C \\ & t(\alpha, q \circ (\tau_1|^q_x \circ \alpha), \ldots, q \circ (\tau_l|^q_x \circ \alpha)) =_C \\ & t(\alpha, q \circ j_{\tau_1}, \ldots, q \circ j_{\tau_l}) =_C \\ & t(\alpha, \beta_{j_{\tau_1}}, \ldots, \beta_{j_{\tau_l}}) =_C j_\tau. \end{aligned}$$

We used the definitions stated above, the induction hypothesis, the assumption that $q \in R(\iota)$ and repeated application of lemma 2. Furthermore $j_\tau \neq_R 0$ because $t \neq_R 0$. Now we proceed as follows. For $\tau \neq_R 0$ construct E_τ, V_τ, I_τ and j_τ. Then define

$$p :\equiv j_\tau \cdot \prod_{\{i,j\}, i \neq_R j,\ i,j \in I_\tau} (i - j) \in R(V_\tau).$$

(Remark: $i \neq_R j$ iff $i \not\equiv j$.) With $j_\tau \neq_R 0$ we see that $p \neq_R 0$. Therefore we can find rational constants $b_{i_1}, \ldots, b_{i_k}, a$ for the variables $\beta_{i_1}, \ldots, \beta_{i_k}, \alpha \in V_\tau$ appearing in p such that $p|^{b_{i_1}, \ldots, b_{i_k}, a}_{\beta_{i_1}, \ldots, \beta_{i_k}, \alpha} \neq_R 0$. Let p^* denote

$p|^{b_{i_1},\ldots,b_{i_k},a}_{\beta_{i_1},\ldots,\beta_{i_k},\alpha}$. From the construction of the polynomial p it follows that for two terms $i, j \in I_\tau$ with $i \neq_R j$

$$i^* \neq_R j^*.$$

Therefore it is easy to construct a polynomial $q \in R(\iota)$ that satisfies the equations

$$b_i =_C q \circ i^*,\ i \in I_\tau,$$

namely

$$q :\equiv \sum_{i \in I_\tau} b_i \cdot \prod_{i,j \in I_\tau,\ j \neq_R i} \frac{(\iota - i^*)}{(j^* - i^*)} \in R(\iota).$$

We get then with (3),

$$\tau|^q_x \circ a =_C t(a, b_{i_1}, \ldots, b_{i_l}) =_C j^*_\tau \neq_R 0.$$

Because of $t(a, b_{i_1}, \ldots, b_{i_l}) \in Q$ we conclude $t(a, b_{i_1}, \ldots, b_{i_l}) \neq_C 0$ and $\tau|^q_x \circ a \neq_C 0$. Since $0|^q_x \circ a =_C 0$ we conclude $\tau \neq_C 0$, which completes the proof. ∎

Example 3 *To illustrate the proof let us compute the polynomial q explicitly for the term given in example 2.4:*

$$\begin{aligned}\tau &\equiv (x \circ 1 - 1) \cdot (x \circ x - x \circ 1) =_C \\ & x \circ 1 \cdot\ x \circ x - x \circ 1^2 + x \circ 1 - x \circ x \equiv \\ & t(x \circ x, x \circ 1).\end{aligned}$$

For the terms x and 1 we get

$$\begin{aligned}E_x = \{\beta_\alpha =_C q \circ \alpha\},\ V_x = \{\alpha, \beta_\alpha\},\\ I_x = \{\alpha\},\ j_x \equiv \beta_\alpha,\\ E_1 = \emptyset,\ V_1 = \emptyset,\ I_1 = \emptyset,\ j_1 \equiv 1.\end{aligned}$$

For $t(x \circ x, x \circ 1)$ we get

$$\begin{aligned}E_{t(x\circ x, x\circ 1)} &= \{\beta_\alpha =_C q \circ \alpha,\ \beta_{\beta_\alpha} =_C q \circ \beta_\alpha,\\ &\qquad \beta_1 =_C q \circ 1\},\\ V_{t(x\circ x, x\circ 1)} &= \{\alpha, \beta_\alpha, 1,\ \beta_{\beta_\alpha}\},\\ I_{t(x\circ x, x\circ 1)} &= \{\alpha, \beta_\alpha, 1\},\\ j_{t(x\circ x, x\circ 1)} &\equiv t(\beta_{\beta_\alpha}, \beta_1).\end{aligned}$$

$$\begin{aligned}p \ &\equiv\ t(\beta_{\beta_\alpha}, \beta_1) \cdot (1 - \beta_\alpha) \cdot (1 - \alpha) \cdot (\beta_\alpha - \alpha)\\ &=_C\ (\beta_{\beta_\alpha} \cdot \beta_1 - \beta_1^2 + \beta_1 - \beta_{\beta_\alpha}) \cdot\\ &\qquad (1 - \beta_\alpha) \cdot (1 - \alpha) \cdot (\beta_\alpha - \alpha).\end{aligned}$$

If we set $a \equiv -1,\ b_\alpha \equiv 0,\ b_{\beta_\alpha} \equiv 1,\ b_1 \equiv 2$ we get for p the value $-2 \neq_R 0$. For the terms in I_τ we get $\alpha^* =_R -1,\ 1^* =_R 1,\ \beta^*_\alpha =_R 0$ which of course are all different. So the computation yields for q

$$\begin{aligned}q =_C\ & 0 \cdot \frac{(\iota - 0)(\iota - 1)}{(-1 - 0)(-1 - 1)} +\\ & 1 \cdot \frac{(\iota - 1)(\iota + 1)}{(0 - 1)(0 + 1)} +\\ & 2 \cdot \frac{(\iota - 0)(\iota + 1)}{(1 - 0)(1 + 1)} =_C \iota + 1.\end{aligned}$$

Finally we want to generalize lemma 5 to the multivariate case.

Lemma 6 *For a term $\tau \in T(x_1, \ldots, x_k)$ of the form $\tau \equiv t(\iota, v_1, \ldots, v_l),\ t \in NR(\iota, v_1, \ldots, v_l)$ with $v_1, \ldots, v_l \in V(x_1, \ldots, x_k)$ we have the property*

$$t(\iota, v_1, \ldots, v_l) \neq_C 0 \text{ iff } t(\iota, v_1, \ldots, v_l) \neq_R 0.$$

Proof. First we show the fact for the special case of $\tau \in T(x_1, x_2)$ with

$$\tau \equiv t(\iota, x_1 \circ \tau_1, \ldots, x_2 \circ \sigma_1, \ldots).$$

We substitute for x_2 a term $\rho(x_1)$ such that $x_2 \circ \sigma_i|^{\rho(x_1)}_{x_2}$ is not equal to one of the arguments $x_1 \circ \tau_j$. This we can force by taking $\rho(x_1) :\equiv x_1 \circ x_1 \circ \ldots \circ x_1$ (sufficiently often) such that the r.h.s. term never appears as subterm in $x_1 \circ \tau_j$. For the general case we repeat this substitution until we get a term in $T(x_1)$ and then we can apply lemma 5. ∎

Theorem 1 *The function N_C is a normal form function for terms in C with respect to the equivalence relation $=_C$. For the set of normal forms we have $NC = T$.*

Proof. The first property to show for a normal form function is $N_C(N_C(\tau)) \equiv N_C(\tau)$. This is clear from the construction of N_C. For the second property assume that $\tau =_C \sigma$ and $N_C(\sigma) \not\equiv N_C(\tau)$. We conclude $N_C(\sigma) \neq_R N_C(\tau)$ since N_C gives rational normal form. Therefore $N_C(\sigma) - N_C(\tau) \neq_R 0$. By definition of N_C we have

$$N_C(N_C(\sigma) - N_C(\tau)) =_R N_R(N_C(\sigma) - N_C(\tau)) \neq_R 0.$$

Since $N_R(N_C(\sigma) - N_C(\tau)) \in T$ we conclude with lemma 6 that $N_R(N_C(\sigma) - N_C(\tau)) \neq_C 0$. On the other hand we conclude from $\tau =_C \sigma$ with the property $N_C(\tau) =_C \tau$ of lemma 4 that $N_R(N_C(\sigma) - N_C(\tau)) =_C 0$ which results in a contradiction. For the set of normal forms remark that if $\sigma \in T$ there is $N_C(\sigma) \equiv \sigma$ and so $NC = T$. ∎

Remark. The size of the normal form can grow exponentially with the size of the input. Consider for example the term

$$(\sigma + \tau) \circ (\sigma + \tau) \circ \ldots \circ (\sigma + \tau). \text{ (see below)}$$

5 Implementation

The implementation of the algorithm in *mathematica* is given by the following compact rule-based program, using the full power of the pattern-matching abilities of *mathematica* [7].

```
NR[r_]:=Together[
         ExpandAll[Cancel[Together[r]]]]
(* NR returns rational normal form *)

Const[n_Integer]:=True
Const[j]=False
Const[-a_]:=Const[a]
Const[1/a_]:=Const[a]
Const[a_+b_]:=Const[a] && Const[b]
Const[a_*b_]:=Const[a] && Const[b]
Const[a_^n_Integer]:=Const[a]
Const[x_]:=False}
(* Const identifies constant terms *)

Comp[a_]=Map[NR,a]
Comp[x___,a_,b_,c___]:=
    Map[NR,Comp[x,a,c]]/;Const[a]
Comp[x___,j,c_,d___]:=
    Map[NR,Comp[x,c,d]]
Comp[x___,c_,j,d___]:=
    Map[NR,Comp[x,c,d]]
Comp[x___,a_+b_,c__]:=
    Map[NR,Comp[x,Comp[a,c]+Comp[b,c]]]
Comp[x___,a_*b_,c__]:=
    Map[NR,Comp[x,Comp[a,c]*Comp[b,c]]]
Comp[x___,-a_,c__]:=
    Map[NR,Comp[x,-Comp[a,c]]]
Comp[x___,1/a_,c__]:=
    Map[NR,Comp[x,1/Comp[a,c]]]
Comp[x___,a_^n_Integer,c__]:=
    Map[NR,Comp[x,Comp[a^(n-1),c]*Comp[a,c]]]
Attributes[Comp]={Flat}
(* Comp may take any number of arguments due
to the associativity. This is expressed by
the Attribute Flat which means that e.g.
Comp[a,Comp[b,c]]= Comp[a,b,c].
The axioms of composition are applied to the
arguments of Comp and after every application
Map[NR,_] establishes rational normal form on
all subterms. *)
```

The first example given in the introduction is handled by this program as follows.

```
In[1]:=Comp[Comp[F+1,1+
            Comp[a,(1+j)^2]],x-1]}

Out[1]= 1+Comp[F,1+Comp[a,x^2]]
```

Now let us compute the complicated expression from the introduction.

```
In[2]:=Comp[Comp[Comp[F+1,1+
          Comp[a,(1+j)^2]],a-1]+
          j*(1+Comp[F,1+Comp[a,a^2]]),
          Comp[Comp[F,a+a^2],Comp[x^2-1]]-
          Comp[F+1,Comp[Comp[a,j^2-1],x]*
          (1+Comp[Comp[a,j*(2+j)],x-1])]]}

Out[2]= 0
```

As illustration of the last remark in the previous chapter we compute.

```
In[3]:= Comp[s+t,s+t,s+t,x]}

Out[3]= Comp[s,Comp[s,Comp[s,x]+Comp[t,x]]+
          Comp[t,Comp[s,x]+ Comp[t,x]]]+
          Comp[t,Comp[s,Comp[s,x]+Comp[t,x]]+
          Comp[t, Comp[s, x] + Comp[t, x]]]
```

6 Concluding Remarks

A natural interpretation for composition terms are programs built up by rational operations and function calls. For example the two terms of our first introducing example could be interpreted as the mathematical representations of the following two programs. The first term representing the computation of $f(x)$:

$$
\begin{aligned}
&function\ f(x);\\
&f := x - 1;\\
&f := h(f);\\
&return(f);
\end{aligned}
$$

$$
\begin{aligned}
&function\ h(x);\\
&h := (x+1)^2;\\
&h := 1 + a(h);\\
&h := l(h);\\
&return(h);
\end{aligned}
$$

$$
\begin{aligned}
&function\ l(x);\\
&l := F(x);\\
&l := l + 1;\\
&return(l);
\end{aligned}
$$

The second term representing the computation of $g(x)$:

$$
\begin{aligned}
&function\ g(x);\\
&g := x^2;\\
&g := 1 + a(g);\\
&g := F(g);\\
&g := g + 1;\\
&return(g);
\end{aligned}
$$

These two programs, using unknown subprograms for computing $a(x)$ and $F(x)$, will compute the same results for all $a(x)$, $F(x)$ and inputs x. Now we are not only in the position to tell whether programs of the kind above

always compute the same results, but furthermore we can, when they are different, give by lemma 5 and 6 explicitely instances of the unknown subprograms such that this is the case.

A natural extension of the algorithm would be to the case of multivariate functions. Another extensions could be made by introducing additional functions into the theory by new axioms like $exp \circ (x + y) = (exp \circ x) \cdot (exp \circ y)$.

When interested in the rewriting approach [5], it could be worth to restrict to rings, since it is difficult to see how to do the computation of normal forms in fields, classically based on GCD-computation, by rewriting techniques. In rings Knuth-Bendix-algorithms for computing the normal form are known. On the other hand a pure compositional structure is a monoid also allowing normal form computation by a Knuth-Bendix-algorithm. Therefore it seems quite plausible to extend the Knuth-Bendix-algorithm for rings to a normal form algorithm for function rings, involving rules similar to those in the *mathematica* program given above. In this framework the composition axiom for constants could serve as a test case for conditional term rewriting.

7 Acknowledgements

I wish to thank Prof. E.Engeler for his encouragement and help in improving the paper, Oliver Gloor for carefully reading the paper, and an anonymous referee for his knowledgeable comments.

References

[1] Aberer, K.: "Normal Forms in Combinatory Differential Fields". *ETH-Report No. 89-01*, (1989).

[2] Davenport, J.H., Siret, Y. and Tournier, E.: "Computer Algebra". *Academic Press, N.Y.*, (1988).

[3] Engeler, E.: "Combinatory Differential Fields". to appear in *Theoretical Computer Science*, (1990).

[4] Geddes, K.O., Labahn, G., Czapor, S.R.: "Algorithms for Computer Algebra". preprint, (1989).

[5] Le Chenadec, P.: "Canonical Forms in Finitely Presented Algebras", *Research Notes in Theoretical Computer Science, Pitman,* (1986)

[6] Menger, K.: "Function Algebra and Propositional Calculus". *Self-Organizing Systems, Spartan Books*, (1962), p. 525ff.

[7] Wolfram, S.: "Mathematica". *Addison-Wesley Publishing Company*, (1988).