Generalized Constraint Solving over Differential Algebras

Andreas Dolzmann and Thomas Sturm

University of Passau, Germany
dolzmann@uni-passau.de
sturm@uni-passau.de
www.fmi.uni-passau.de/~dolzmann
www.fmi.uni-passau.de/~sturm

Abstract. We describe an algorithm for quantifier elimination over differentially closed fields and its implementation within the computer logic package REDLOG of the computer algebra system REDUCE. We give various application examples, which on the one hand demonstrate the applicability of our software to non-trivial problems, and on the other hand give a good impression of the possible range of applications of our work. Essentially, our elimination technique dates back to Seidenberg. It has been made much more explicit on the basis of the common axioms for differentially closed fields in lectures on differential algebra by Weispfenning. In this explicit form, which we use and describe here, it had remained unpublished so far.

1 Introduction

Since the 1960's REDUCE has permanently been among the most widely accepted computer algebra systems on the market. From the outset it had mainly focused on users and applications in physics. Nowadays, with several modern competitors on the market, it is again at the first place this scientific community of physicists, which highly estimates the superior efficiency of the actual REDUCE packages and, even more important, the highly optimized Portable Standard Lisp compiler.

At the beginning of the 1990's the authors started to develop within REDUCE their computer logic package REDLOG [6]. Meanwhile the authors belong to the permanent REDUCE development group, and REDLOG is an integral part of the REDUCE distribution. The basic idea of REDLOG is to combine methods of computer algebra with on the one hand with symbolic logic on the other hand. REDLOG is a quite comprehensive system including numerous convenient tools for handling and processing first-order formulas. The central and in fact extremely general method is effective quantifier elimination on the background of temporarily fixed languages and theories in the sense of model theory.

In the REDLOG framework, a fixed combination of a language and a theory as mentioned above is called a *context*. A most prominent example for this is the language $(0, 1, +, -, \cdot, \leq)$ of *ordered rings* in combination with the theory of *real closed fields*. This context was actually the starting point for REDLOG. The current version of REDLOG comprises implementations of three regular quantifier elimination procedures for the reals, viz. *partial cylindrical algebraic decomposition* [4], *virtual substitution methods* [25, 28, 5], and *Hermitian quantifier elimination*, which is based on parametric real root counting [29].

During the past ten years, REDLOG has been augmented by several other contexts: algebraically closed fields (complex numbers) [27], discretely valued fields (p-adic numbers) [25, 23, 8, 10], initial Boolean algebras (quantified propositional logic) [22], and Presburger arithmetic (additive theory of the integers) [26]. In addition, there are first steps done in incorporating free term algebras [24].

REDLOG thus covers a comparatively wide range of—mainly commutative—algebra. On rather comprehensive and most fruitful branch of computer algebra had, however, been passed over so far: *differential algebra*. The present paper is closing this gap for the following setup: Basic *constraints* are ordinary differential equations and inequalities. From these we obtain *systems* of constraints by constructing arbitrary Boolean combinations and in addition admitting universal and existential quantification. The language is thus $(0, 1, +, -, \cdot, ')$, where $'$ is a unary differential operator. The theory is that of *differentially closed fields* [18]. Returning to our initial remarks, it is obvious that suitable tools in the area of differential algebra are of high interest in particular to the area of physics, which REDUCE is so closely related to.

In Section 2 we are going to illuminate the notion of differentially closed fields, which are unfortunately less natural than algebraically closed fields or real closed fields. In Section 4 we outline the elimination algorithm. In Section 5, we describe our new REDLOG context DCFSF, which goes considerably beyond the mere implementation of the elimination procedure. In Section 6 we give various application examples, which on the one hand demonstrate the applicability of our software to non-trivial problems, and on the other hand give a good impression of the possible range of applications of our work. In Section 7 we finally summarize our results.

2 The Notion of Differentially Closed Fields

Differential algebra dates back to Ritt [17]. Ritt had the initial idea to treat differential equations to a large extent in a purely algebraic framework, and developed a corresponding algebraic framework. As a major result, he proved his differential Nullstellensatz, which is a perfect analogue to Hilbert's Nullstellensatz for algebraically closed fields [11].

The first major algorithmic contribution in differential algebra was Seidenberg's elimination theory [21]. It provided an elimination theorem with a perfectly algorithmic proof. We give a formulation of this result from our logical point of view:

Theorem 1 (Seidenberg, 1956). *Let K be a differential field. Consider for differential polynomials $f_1, \dots, f_m, g \in K\{y_1, \dots, y_n, u_1, \dots, u_k\}$ the formula $\varphi \equiv \exists y_1 \dots \exists y_n \psi$, where*

$$\psi \equiv \bigwedge_{i=1}^{m} f_i(y_1, \dots, y_n, u_1, \dots, u_k) = 0 \land g(y_1, \dots, y_n, u_1, \dots, u_k) \neq 0.$$

Then there exists $\hat{\varphi} \equiv \bigvee_{j=1}^{l} \psi_j$, where

$$\psi_j \equiv \bigwedge_{i=1}^{m_j} f_{ij}(u_1, \dots, u_k) = 0 \land g_j(u_1, \dots, u_k) \neq 0, \quad f_{1j}, \dots, f_{s_j j}, g_j \in K\{u_1, \dots, u_k\},$$

and some differential extension field $K' \supseteq K$ such that $K' \models \varphi \longleftrightarrow \hat{\varphi}$. Moreover $\hat{\varphi}$ can be effectively constructed from φ. □

By means of prenex normal form computation, successive elimination of quantifier blocks from the inside to the outside, the equivalence between $\forall y_1 \dots \forall y_n \psi$ and $\neg\exists y_1 \dots \exists y_n \neg\psi$, and disjunctive normal form computations, this theorem can be extended to arbitrary first-order input formulas. Note that in the theorem, the extension K' of K depends on the input formula φ, such that it does not really provide a quantifier elimination procedure for any fixed structure or theory.

On the basis of Seidenberg's work Robinson introduced in 1959 the notion of a *differentially closed field* [18]. He axiomatized the class of differentially closed fields by combining the following sets of axioms:

1. The field axioms.
2. The Leibniz axioms for the derivative.
3. For each existential formula the equivalence between this formula and the corresponding quantifier-free formula obtained according to Seidenberg.

Thus while Seidenberg provided sort of a dynamic process providing equivalents in differential extension fields, Robinson switched from the outset to a sufficiently large fields such that no extension is necessary. From a model theoretic point of view, these differentially closed fields are perfect analogues of algebraically closed fields. Unfortunately, there is no natural example for such fields which could play the role of the complex numbers for algebraically closed fields. Robinson's main result was the *model completeness* of the class of differentially closed fields. It is obvious that Seidenberg's procedure is an effective quantifier elimination procedure for the class of differentially closed fields. Consequently, beyond model completeness, differentially closed fields even have the stronger property of *substructure completeness*, which is equivalent to the existence of quantifier elimination. Interestingly, Robinson who had just one year before discussed this stronger phenomenon [19]—without introducing the notion of substructure completeness, however—did not refer to this at all.

In 1968 Blum reanalyzed Seidenberg's proof wrt. the assumptions on the differential extension fields made there [2, 3]. By indirect model theoretic methods, viz. saturated models, she found a natural axiomatization of differentially closed fields in contrast to Robinson's pragmatic collection of all possible results of Seidenberg's procedure:

3′. For each pair f, g of univariate differential polynomials with $\operatorname{ord}(f) > \operatorname{ord}(g)$ there is a c in the field such that $f(c) = 0$ and $g(c) \neq 0$.

Note that this is still an infinite set of axioms. In contrast to Robinson's, however, the axioms are very natural. In fact, they nicely resemble the axiomatization of algebraically closed fields. At that time the notion of substructure completeness had been introduced by Sacks [20], and the scientific community was absolutely aware of the fact that differentially closed fields admit quantifier elimination via Seidenberg's procedure.

It is a straightforward idea to come full circle by reformulating Seidenberg's elimination procedure in such a way that exactly Blum's axioms become explicit there. This has actually been done by Weispfenning in 1973. This work has been included in his lectures on differential algebra during the 1980's but remained unpublished in the literature so far. We are going to outline in the following Section 4 our revision of the Weispfenning procedure, which we have used for our implementation.

3 The Practical Relevance of Differentially Closed Fields

Before turning to such technical aspects, we should, however, motivate the practical relevance of the method. We have already mentioned that there is no natural example for a differentially closed field at all. That is, quantifier elimination will certainly never take place in the structures that the users have actually in their minds. It rather takes place in a differentially closed extension field, where there generally exist elements that cannot be interpreted as functions.

Nevertheless, the quantifier elimination results will for many first-order formulations of natural questions provide information also on the differential field actually under consideration. At the first place, this applies to input formulas that are either purely existential or purely universal.

Example 2 (Solvability Conditions for parametric systems). One example for a purely existential question is that for the solvability of a parametric system of differential equations

$$\psi \equiv f_1(x_1, \dots, x_n, u_1, \dots, u_m) = 0 \land \dots \land f_k(x_1, \dots, x_n, u_1, \dots, u_m) = 0,$$

where the $f_1, \dots, f_k \in \mathbb{Q}\{x_1, \dots, x_n, u_1, \dots, u_m\}$ are differential polynomials. We are interested in conditions on the parameters $u_1, \dots, u_m$ for the solvability of the system wrt. $x_1, \dots, x_n$. A corresponding first-order formulation is given by $\varphi \equiv \exists x_1 \dots \exists x_n \psi$. □

For such existential problems, quantifier elimination yields a quantifier-free formula $\hat{\varphi}$ such that for any differentially closed field $\bar{K}$, we have $\bar{K} \models \varphi \longleftrightarrow \hat{\varphi}$. In other words, $\hat{\varphi}$ is a necessary and sufficient condition in the parameters $u_1, \dots, u_m$ for the solvability of ψ in the differentially closed field $\bar{K}$. From this point of view we have in particular that $\hat{\varphi}$ is a necessary condition: $\bar{K} \models \varphi \longrightarrow \hat{\varphi}$, alternatively $\bar{K} \models \forall u_1 \dots \forall u_m(\varphi \longrightarrow \hat{\varphi})$, which can in turn be rewritten as follows:

$$\begin{aligned}
\forall u_1 \dots \forall u_m(\varphi \longrightarrow \hat{\varphi}) &\iff \forall u_1 \dots \forall u_m(\exists x_1 \dots \exists x_n(\psi) \longrightarrow \hat{\varphi}) \\
&\iff \forall u_1 \dots \forall u_m\big(\neg\exists x_1 \dots \exists x_n(\psi) \lor \hat{\varphi}\big) \\
&\iff \forall u_1 \dots \forall u_m\big(\forall x_1 \dots \forall x_n(\neg\psi) \lor \hat{\varphi}\big) \\
&\iff \forall u_1 \dots \forall u_m \forall x_1 \dots \forall x_n(\neg\psi \lor \hat{\varphi}).
\end{aligned}$$

So the fact that $\hat{\varphi}$ is a *necessary* condition can be expressed as a universal sentence, and thus holds in all subfields of $\bar{K}$, in particular in the field actually under consideration.

Applied to our Example 2, the quantifier elimination result $\hat{\varphi}$ will thus possibly identify certain choices of parameters for which the considered system has no solutions. Moreover, we expect the case distinctions on the parameters made in $\hat{\varphi}$, to be typical for the input problem rather than for the considered differential field. They would then provide a certain structural insight into the problem modeled by the parametric system ψ. The first point is a fact, which we have proved above. The second point is not mathematically precise and has to be substantiated by convincing computation examples in Section 6.

Example 3 (Conditions on the solutions of systems). As an example for a purely universal question consider the differential equation ${x'}^2 + x = 0$. By taking the derivative we obtain

$$0 = ({x'}^2 + x)' = 2x'x'' + x' = x'(2x'' + 1).$$

It is thus necessary for solutions x that $x' = 0$ or $x'' = -1/2$.[1] We can ask for such conditions by means of a universal formula:

$$\varphi \equiv \forall x \psi, \quad \text{where} \quad \psi \equiv {x'}^2 + x = 0 \longrightarrow x' = a \lor x'' = b.$$

As a quantifier-free equivalent our procedure delivers $\hat{\varphi} \equiv a = 0 \land 2b + 1 = 0$. □

As for Example 2 we formally have $\bar{K} \models \varphi \longleftrightarrow \hat{\varphi}$. This time the direction $\bar{K} \models \hat{\varphi} \longrightarrow \varphi$ corresponds to a universal sentence:

$$\forall a \forall b(\hat{\varphi} \longrightarrow \varphi) \iff \forall a \forall b(\hat{\varphi} \longrightarrow \forall x \psi) \iff \forall a \forall b(\neg\hat{\varphi} \lor \forall x \psi) \iff \forall a \forall b \forall x(\neg\hat{\varphi} \lor \psi).$$

That is, the quantifier-free condition $\hat{\varphi}$ on the parameters is *sufficient* for φ in all subfields of $\bar{K}$, in particular in the field actually under consideration.

Applying this observation to Example 3 is a bit puzzling at first: We ask for a necessary condition on $a = x'$ and $b = x''$ for being a solution of the considered equation ${x'}^2 + x = 0$. According to the discussion above we may, however, only conclude that the obtained result is—in any reasonable differential field—a *sufficient* condition on $a = x'$ and $b = x''$ for being a *necessary* condition as required in the formulation of the input formula. We see that, in general, it requires a certain intuition about mathematical logic to deal with the results of our procedure. Here, the situation can be resolved as follows: From $\bar{K} \models \varphi \longleftrightarrow \hat{\varphi}$ it follows that in particular $\bar{K} \models (\varphi \longleftrightarrow \hat{\varphi})\big[a/0, b/-\frac{1}{2}\big]$. That is,

$$\bar{K} \models \forall x\Big({x'}^2 + x = 0 \longrightarrow x' = 0 \lor x'' = -\frac{1}{2}\Big) \longleftrightarrow \Big(0 = 0 \land 2 \cdot \Big(-\frac{1}{2}\Big) + 1 = 0\Big).$$

Equivalently, $\bar{K} \models \forall x\big({x'}^2 + x = 0 \longrightarrow x' = 0 \lor x'' = -\frac{1}{2}\big)$, which as a universal formula holds also in the subfield of $\bar{K}$ actually under consideration.[2]

4 The Elimination Algorithm

We start with some basic definitions. Consider a differential polynomial $f \in \mathbb{Z}\{y, u_1, \dots, u_k\}$, where y plays the special role of being quantified. We introduce some notions, which depend on this variable y, where y is not mentioned explicitly anymore. The *order* $\operatorname{ord}(f)$ of f is the largest $s \in \mathbb{N}$ such that the s-th derivative $y^{(s)}$ occurs in f. For our next definitions, we consider f of order s as a univariate polynomial in

$$\mathbb{Z}\{u_1, \dots, u_k\}[y, y^{(1)}, \dots, y^{(s-1)}][y^{(s)}].$$

The *degree* of this univariate polynomial is denoted by $\deg(f)$. Its leading coefficient is called the *initial* $I(f)$ of f. Its *reductum* $\operatorname{red}(f)$ is f without the leading monomial. Its partial derivation $\partial f / \partial y^{(s)}$ is called the *separant* $S(f)$ of f. Whenever we are going to apply polynomial pseudo-division in our elimination procedure, we also have in mind this univariate representation.

[1] This very instructive problem has been suggested by E. Pankratiev at the MEXMAT faculty of Moscow State University during the second author's stay there.

[2] This pragmatic treatment for the considered example has been suggested by A. Seidl. In general, it will be interesting to develop a general theory for transferring results for certain types of input formulas. It is, however, beyond the scope of this work.

Example 4. Consider $f = y'y''^3 + u_1y'' + y'' + u_1$. Then we have $\operatorname{ord}(f) = 2$. Rewritten as a univariate polynomial, this is

$$y'y''^3 + (u_1 + 1)y'' + u_1.$$

From this we see that $\deg(f) = 3$, $I(f) = y'$, $\operatorname{red}(f) = (u_1+1)y''+u_1$, and $S(f) = 3y'y''^2+u_1+1$. □

At some points, however, we take the alternative view of f as a distributive multivariate polynomial in

$$\mathbb{Z}\{u_1, \dots, u_k\}[y, y^{(1)}, \dots, y^{(s)}].$$

From that point of view, very polynomial has a representation of the form $f = \sum_{t\in \mathrm{T}(f)} \mathrm{C}(f, t) \cdot t$, where $\mathrm{T}(f)$ is the set of distributive terms in f, and $\mathrm{C}(f, t)$ is the coefficient of a given term $t \in \mathrm{T}(f)$ of f.

Example 5. Resuming Example 4 with $f = y'y''^3 + (u_1 + 1)y'' + u_1$, we have $\mathrm{T}(f) = \{y'y''^3, y'', 1\}$, and $\mathrm{C}(f, y'y''^3) = 1$, $\mathrm{C}(f, y'') = u_1 + 1$ and $\mathrm{C}(f, 1) = u_1$. □

We want to be able to eliminate quantifiers from an arbitrary first-order formula over the language $(0, 1, +, -, \cdot, ')$. In view of our discussion after Theorem 1, it suffices to consider input formulas of the form

$$\exists y\Big(\bigwedge_{i=1}^{m} f_i = 0 \wedge g \neq 0\Big), \quad \text{where} \quad f_1, \dots, f_m, g \in \mathbb{Z}\{y, u_1, \dots, u_k\}.$$

Our quantifier elimination algorithm is recursive. We start our discussion with 5 base cases, in which there occurs no recursion. Then we proceed to the actual recursion.

Base Cases

Base Case 1 If $g = 0$, then the elimination result is obviously false.

Base Case 2 If we have $m = 0$, i.e., there is no equation in our input, then the elimination result is the following disjunction of coefficients stating that g is not the zero polynomial:

$$\bigvee_{t\in \mathrm{T}(g)} \mathrm{C}(g, t) \neq 0.$$

Base Case 3 If $m = 1$ and $g \in \mathbb{Z}\setminus\{0\}$, then the elimination result is the following formula specifying that f is either not constant or the constant zero polynomial:

$$\mathrm{C}(f_1, 1) = 0 \vee \bigvee_{t\in \mathrm{T}(f_1)\setminus\{1\}} \mathrm{C}(f_1, t) \neq 0.$$

Base Case 4 If g is of the form $I(f_1) \cdot \hat{g}$ and $\operatorname{ord}(\hat{g}) < \operatorname{ord}(f_1)$, i.e., if we know that $I(f_1) \neq 0$, then the elimination result is

$$\bigvee_{t\in \mathrm{T}(f_1)} \mathrm{C}(f_1, t) \neq 0 \wedge \bigvee_{t\in \mathrm{T}(\hat{g})} \mathrm{C}(\hat{g}, t) \neq 0.$$

Base Case 5 If $m = 1$ and $g = I(f_1) \cdot \hat{g}$ and $\operatorname{ord}(\hat{g}) = \operatorname{ord}(f_1)$, then we proceed as follows: We compute the remainder r of the polynomial pseudo-division of $I(f_1)^{de} \cdot \hat{g}^d$ by f where $d = \deg(f_1)$ and $e = \deg(\hat{g})$. Our elimination result is similar to the previous case with r instead of $\hat{g}$:

$$\bigvee_{t\in \mathrm{T}(f_1)} \mathrm{C}(f_1, t) \neq 0 \wedge \bigvee_{t\in \mathrm{T}(r)} \mathrm{C}(r, t) \neq 0.$$

Recursion

Let $s_i = \operatorname{ord}(f_i)$ and $d_i = \deg(f_i)$ for $i \in \{1, \dots, m\}$. Then we can assume wlog. that $(s_1, d_1) \geq \cdots \geq (s_m, d_m)$ wrt. the lexicographic order on $\mathbb{N}^2$.

Recursion Case 1 If $\deg(f_m) = 0$, i.e., if y does not occur in f_m, then we recursively apply our algorithm to

$$\exists y\Big(\bigwedge_{i=1}^{m-1} f_i = 0 \land g \neq 0\Big).$$

This yields a quantifier-free equivalent $\hat{\varphi}$. The elimination result is then $f_m = 0 \land \hat{\varphi}$.

Recursion Case 2 In view of Recursion Case 1, we can assume that y occurs in f_m. In a first step we construct a case distinction on the initial of f_m being zero or not. That is, we replace our considered formula

$$\exists y\Big(\bigwedge_{i=1}^{m} f_i = 0 \land g \neq 0\Big)$$

by $\varphi_1 \lor \varphi_2$, where

$$\varphi_1 = \exists y\Big(\bigwedge_{i=1}^{m-1} f_i = 0 \land \operatorname{red}(f_m) = 0 \land I(f_m) = 0 \land g \neq 0\Big),$$
$$\varphi_2 = \exists y\Big(\bigwedge_{i=1}^{m} f_i = 0 \land I(f_m) \cdot g \neq 0\Big).$$

For φ_1 we can by recursion obtain a quantifier-free equivalent $\hat{\varphi}_1$. For the elimination of φ_2 yielding $\hat{\varphi}_2$ we are going to distinguish cases once more. The final elimination result will be $\hat{\varphi}_1 \lor \hat{\varphi}_2$.

Recursion Case 2.1 If $m = 1$ and $\operatorname{ord}(g) \leq \operatorname{ord}(f_1)$, then we proceed as described in the base cases 3–5 discussed above.

Recursion Case 2.2 Assume $m = 1$ and $\operatorname{ord}(g) > \operatorname{ord}(f_1)$. We compute the remainder r of the polynomial pseudo-division

$$S(f_1)^{d_1} g : f_1^{(s'-s_1)}, \quad \text{where} \quad s' = \operatorname{ord}(g).$$

If $\deg(f_1) = 1$, then $\hat{\varphi}_2$ can be recursively computed from $\exists x(f_1 = 0 \land I(f_1) \cdot r \neq 0)$. Otherwise we once more perform a polynomial pseudo-division computing the remainder f of the division

$$S\big(S(f_1)\big)^{d_1} f_1 : S(f_1)^{(s_1-\hat{s})}, \quad \text{where} \quad \hat{s} = \operatorname{ord}\big(S(f_1)\big).$$

The partial elimination result $\hat{\varphi}_2$ is then obtained by recursively applying our procedure to the two constituents of the following case distinction on the possible vanishing of $S(f_1)$:

$$\exists y\big(f_1 = 0 \land S(f_1) \cdot I(f_1) \cdot r \neq 0\big) \lor \exists y\big(S(f_1) = 0 \land f = 0 \land I(f_1) \cdot r \neq 0\big).$$

Recursion Case 2.3 Assume $m > 1$. We compute the remainder r of the polynomial pseudo-division

$$I\big(f_m^{(s_{m-1}-s_m)}\big)^{d_{m-1}} f_{m-1} : f_m^{(s_{m-1}-s_m)}.$$

The partial elimination result $\hat{\varphi}_2$ is then computed by recursively applying our method to

$$\exists y\Big(\bigwedge_{i=1}^{m-2} f_i = 0 \land f_m = 0 \land r = 0 \land I(f_m) \cdot g \neq 0\Big).$$

5 Implementation

The procedure described in the previous section has been implemented in REDLOG, which stands for "REDUCE logic" system [6]. It provides an extension of the computer algebra system REDUCE to a computer logic system implementing symbolic algorithms on first-order formulas wrt. temporarily fixed first-order languages and theories. Such a choice of language and theory is called a *context*. So far, the following REDLOG contexts had been available:

- OFSF (Ordered fields, standard form representation of terms). These are real closed fields such as the real numbers with ordering. This context was the original motivation for REDLOG. It is still the most important and sophisticated one.
- ACFSF (Algebraically closed fields, standard form representation of terms). These are algebraically closed fields such as the complex numbers.
- PASF (Presburger Arithmetic, standard form representation of terms). These are the integers with the addition, negation, and congruences wrt. fixed moduli.
- DVFSF (Discretely valued fields, standard form representation of terms). The most prominent example for discretely valued fields are p-adic numbers for some prime p with abstract divisibility relations.
- IBALP (Initial Boolean Algebras, Lisp prefix representation of terms). These are Boolean algebras with two elements, which are uniquely determined up to isomorphisms.

The work discussed here establishes another such context DCFSF:

- DCFSF (Differentially closed fields, standard form representation of terms). Our context for dealing with differentially closed fields. As mentioned above there is, unfortunately, no natural example for a differentially closed field. As shown in this paper, however, one can still obtain relevant and interpretable results also for reasonable differential fields.

The idea of REDLOG is to combine methods from computer algebra with logic thus extending the computer algebra system REDUCE to a computer logic system. In this extended system both the algebraic side and the logic side greatly benefit from each other in numerous ways.

We give a short overview of the REDLOG functions currently available. Details can be found in the REDLOG user manual [9].

Before turning to the quantifier elimination `rlqe`, which is the main subject of this paper, we discuss the other methods available in the new context DCFSF: For this, we classify them into three groups: Functions for simplifying formulas, normal form computations, and utility functions.

We start with the functions for simplification. The techniques used for simplifications have been described for real closed fields in [7], and adapted to differentially closed fields for our implementation:

`rlsimpl`: This is the very fast standard simplifier applying Boolean simplifications rules and some few algebraic simplification rules.
`rlgsc`, `rlgsd`, `rlgsn`: These functions simplify a formula using Gröbner bases for discovering algebraic dependencies between terms. The resulting formulas are in either conjunctive or disjunctive normal form.
`rltab`, `rlitab`, `rlatab`: These functions construct systematic case distinctions wrt. certain terms, thus possibly simplifying the formula.

Some applications required normal form computations. These result in formulas which typically have a comprehensible Boolean structure:

`rlnnf` computes a negation normal form of a formula, i.e. a formula containing only the Boolean operators $\wedge$, $\vee$, in particular no negation $\neg$.
`rlpnf` computes a prenex normal form.
`rldnf` computes a conjunctive normal form. During the computation we apply simplification techniques extending the ideas of Quine [14–16] and McClusky [13] for propositional calculus to our algebraic situation.
`rlcnf` computes a conjunctive normal form in analogy to `rldnf`.

The utility functions allow to comfortably construct, manipulate, and access formulas or parts of formulas:

`rlmatrix` returns the matrix, i.e. the quantifier-free part, of a prenex formula.
`rlall` binds all free variables of a formula with a prenex universal quantifier.
`rlex` binds all free variables of a formula with an prenex existential quantifier.
`rlatl` computes the set of all contained atomic formulas.

`rlatml` computes the set of all contained atomic formulas together with the number of their occurrences.
`rlterml` computes the set of all contained terms.
`rltermml` computes the set of all contained terms together with the number of their occurrences.
`rlfvarl` computes the list of all variables x that occur at some place where they are not bound by any quantifier.
`rlbvarl` computes the list of all variables bound by some quantifier somewhere.
`rlvarl` computes a list containing the return values of both `rlfvarl` and of `rlbvarl`.
`rlatnum` computes the number of atomic formulas contaid in a formula, counting multiplicities.
`rlqnum` computes the number of quantifiers occurring in a formula.
`for ... mkand` systematically constructs a conjunction.
`for ... mkor` systematically constructs a disjunction.

The elimination algorithm described in the previous section has been implemented in the REDLOG function `rlqe`. This function has two arguments, where the second one is optional: The first one is the formula to eliminate quantifiers from; the second one specifies an *external theory.* Such external theories, which have been introduced in [7], are a general concept present in REDLOG. Formally, an external theory is a set of atomic formulas considered as a conjunction. All elimination results are equivalent to the input formula for parameter values satisfying the external theory. In other words, if $\hat{\varphi}$ is the result of eliminating φ wrt. the external theory ϑ in K', then we have

$$K' \models \bigwedge \vartheta \longrightarrow (\varphi \longleftrightarrow \hat{\varphi}).$$

In the case of `rlqe` in DCFSF, such theories ϑ are exploited as follows: Whenever during the elimination procedure the derivation operator is applied to some $y^{(n)}$ a check is performed whether there is an equation $y^{(n+1)} = t$ contained in ϑ. In the positive case, t is used instead of $y^{(n+1)}$. This allows in particular to specify via $a' = 0$ that a is a constant, and via $t' = 1$ that t is essentially the independent variable. Our examples in the next section are going to demonstrate that this can greatly support the elimination procedure.

The current version REDLOG 3.0 including our work described here is an integral part of the computer algebra system REDUCE 3.8.

6 Application Examples

All our computations have been carried out on a 2 GHz Intel Pentium 4 using 128 MB of RAM.

Example 6 (Example 3 revisited). We start by revisiting Example 3. For the input formula

$$\forall x\big(x'^2 + x = 0 \longrightarrow x' = a \vee x'' = b\big),$$

we obtain the quantifier-free equivalent $\hat{\varphi} \equiv a = 0 \wedge 2b + 1 = 0$ discussed there in less than 10 ms. □

Example 7 (A Benchmark Sequence). In the previous Example 6, we have seen that it is necessary for the solvability of $x'^2 + x = 0$ that $x' = 0$ or $x'' = -1/2$. In either case, it follows for the solutions that $x^{(s)} = 0$ for $s > 2$. This motivates the following sequence of benchmark examples

$$\varphi_s \equiv \exists x\big(x'^2 + x = 0 \wedge x^{(s)} \neq 0\big)$$

for increasing $s \in \mathbb{N}$. The following table collects the obtained quantifier-free equivalents φ_s' and the computation times of our procedure applied to φ_s for some values of s:

s	0	1	2	3	10	20	30	31	32	33	34	35	36
φ_s'	true	true	true	false	false	false	false	false	false	false	false	false	false
time (ms)	< 10	< 10	< 10	< 10	50	440	2480	2870	3440	4240	5280	6490	9030

For $s = 37$ our implementation exceeds the available memory of 128 MB. □

Example 8 (Inhomogeneous System with Polynomial Coefficients). This example has been adapted from Example 7.2 in [1]. The following system is discussed there:

$$y' = Ay + b, \quad \text{where} \quad A = \begin{pmatrix} 0 & 2t \\ -2t & 0 \end{pmatrix}, \quad b = r\begin{pmatrix} 2t\cos(t^2) \\ 2t\sin(t^2) \end{pmatrix}.$$

It is furthermore specified that r is a constant and t is the independent variable, i.e. $r' = 0$ and $t' = 1$. Note that the coefficient matrix A is thus polynomial.

We are interested in deriving conditions on the parameters r and t for the solvability of the system wrt. y. We introduce a new indeterminate a for modeling the trigonometric functions: $a := \sin(t^2)$, and it follows that $a' = 2t\cos(t^2)$. Substitution yields

$$b = r\begin{pmatrix} a' \\ 2ta \end{pmatrix}.$$

For constructing our input formulas, we formulate the system as a quantifier-free formula:

$$\psi \equiv y_1' = 2ty_2 + ra' \wedge y_2' = -2ty_1 + 2rta.$$

We formulate the conditions on r and t as an external theory $\vartheta = \{r' = 0, t' = 1\}$, which we use for all our computations. For

$$\exists y_1 \exists y_2 (\psi \wedge y_1 \neq 0 \wedge y_2 \neq 0)$$

we obtain "true" wrt. ϑ in less than 10 ms. A more interesting result is obtained for

$$\exists y_1 \exists y_2 (\psi \wedge y_1 \neq 0 \wedge y_2 \neq 0 \wedge y_2' \neq 2ty_1).$$

This yields after 50 ms the quantifier-free equivalent wrt. ϑ:

$$t = 0 \vee (a''t - a' + 4at^3 = 0 \wedge a' \neq 0 \wedge a \neq 0 \wedge r \neq 0).$$

Note that $t = 0$ is "false" wrt. ϑ; here the simplifier requires some improvement. It is well-known, that the most general solution for the equations in ψ is of the form

$$y_1 = C_1 \sin(t^2) + C_2 \cos(t^2) + r\sin(t^2), \quad y_2 = C_1 \cos(t^2) - C_2 \sin(t^2).$$

The implicit condition $y_2' \neq 2ty_1$ has thus indeed the consequence $r \neq 0$. Moreover, the fundamental equation $a''t - a' + 4at^3 = 0$ is a homogeneous differential equation for $a = \sin(t^2)$. Accordingly, resubstituting $\sin(t^2)$ for our artificial parameter a yields

$$t = 0 \vee \big(2t\cos(t^2) \neq 0 \wedge \sin(t^2) \neq 0 \wedge r \neq 0\big).$$

For comparison of efficiency we finally give result and timing for the elimination without specifying the external theory ϑ. We then we obtain

$$t = 0 \vee (a''rt - a't'r - r''at + r't'a + 4art^3 = 0 \wedge a'r - r'a \neq 0 \wedge a \neq 0 \wedge r \neq 0),$$

which requires 3160 ms. □

Example 9 (Equilibrium Points of an Electric Circuit). This example is taken from [12]. The following system φ describes a nonlinear electric circuit:

$$\begin{aligned} \varphi \equiv\ L_1 i_1' &= v_6 + v_4 - v_3 \wedge \\ L_2 i_2' &= v_5 - v_4 \wedge \\ C_3 v_3' &= I_{01} + i_1 - \tilde{f}_3(v_3) \wedge \\ C_2 v_4' &= -I_0 - i_1 + i_2 - \tilde{f}_2(v_4) \wedge \\ C_4 v_5' &= I_{02} - i_2 - \tilde{f}_4(v_5) \wedge \\ C_1 v_6' &= -I_0 - i_1 - \tilde{f}_1(v_6), \end{aligned}$$

where

$$\tilde{f}_1(v_6) = 9u_1^2 g_1 v_6 - 6u_1 g_1 v_6^2 + g_1 v_6^3, \qquad \tilde{f}_2(v_4) = 9u_2^2 g_2 v_4 - 6u_2 g_2 v_4^2 + g_2 v_4^3,$$
$$\tilde{f}_3(v_3) = 9u_3^2 g_3 v_3 - 6u_3 g_3 v_3^2 + g_3 v_3^3, \qquad \tilde{f}_4(v_5) = 9u_4^2 g_4 v_5 - 6u_4 g_4 v_5^2 + g_4 v_5^3.$$

Capital letters indicate parameters: $C_1, \ldots, C_4$ are constant capacities, L_0 and L_1 are constant inductances, and I_0, I_{01}, I_{02} are constant current sources. The polynomials $\tilde{f}_1, \ldots, \tilde{f}_4$ are cubic Lagrange polynomials interpolating the nonlinear voltage-current characteristics $f_1, \ldots, f_4$ of corresponding resistors. For $i \in \{1, \ldots, 4\}$, u_i denotes the first extremum of f_i, and $g_i = f_i/(4u_i^3)$.

In the original work [12], the left hand sides of the equations, which contain the derivation operator are set to zero in order to determine the equilibrium points of the circuit. This is, however, a purely algebraic problem then.

In order to get an impression of the current limits of our implementation, we instead try to eliminate as many currents and voltages from the original system in order to derive necessary relations between the parametric quantities. We are able to eliminate the currents i_1 and i_2 and the voltage v_3 wrt. the theory

$$\vartheta = \{C_1' = 0, \ldots, C_4' = 0, L_1' = 0, L_2' = 0, I_0' = 0, I_{01}' = 0, I_{02}' = 0\}.$$

For $\exists i_1 \exists i_2 \exists v_3 \varphi$ and ϑ, we obtain within 80 ms

$$\begin{aligned} 9g_4' L_2 u_4^2 v_5 - 6g_4' L_2 u_4 v_5^2 + g_4' L_2 v_5^3 + 18u_4' L_2 g_4 u_4 v_5 - 6u_4' L_2 g_4 v_5^2 \\ + v_5'' C_4 L_2 + 9v_5' L_2 g_4 u_4^2 - 12 v_5' L_2 g_4 u_4 v_5 + 3v_5' L_2 g_4 v_5^2 - v_4 + v_5 = 0 \land \\ v_4' C_2 + v_5' C_4 - v_6' C_1 - I_{02} - 9g_1 u_1^2 v_6 + 6g_1 u_1 v_6^2 - g_1 v_6^3 \\ + 9g_2 u_2^2 v_4 - 6g_2 u_2 v_4^2 + g_2 v_4^3 + 9g_4 u_4^2 v_5 - 6g_4 u_4 v_5^2 + g_4 v_5^3 = 0 \land \\ \chi = 0, \end{aligned}$$

where χ is a huge irreducible differential polynomial with 1441 monomials in its distributive representation. The elimination of more variables exceeds our memory of 128 MB. Without specifying ϑ the elimination above takes 160 ms. The result then also consists of three equations. The first two of these equations are only slightly more complicated than the displayed ones, while the polynomial corresponding to χ grows to 2543 monomials. □

7 Conclusions

We have described a quantifier elimination procedure for differentially closed fields. For this purpose, we have carefully motivated and introduced the notion of a differentially closed field. We furthermore have motivated that although these structures are not at all natural, results obtained there are in general of high relevance for natural differential fields. Our method is an optimized version of a variant of Seidenberg's famous elimination method. It is implemented in the logic package REDLOG of the computer algebra system REDUCE. This implementation goes far beyond only providing quantifier elimination. Instead it provides a rich experimentation and implementation environment for logic algorithms in differential fields and many other algebraic structures. We have finally demonstrated the applicability of our implementation to non-trivial problems taken from the contemporary scientific literature on computer algebra.

Acknowledgment

This work has been supported by numerous most fruitful discussions with E. Pankratiev at Moscow State University and with A. Seidl and V. Weispfenning. In addition, we wish to point out once more that our version of Seidenberg's elimination in Section 4 is based on notes of lectures on differential algebra by V. Weispfenning.

References

1. Hirokazu Anai and Volker Weispfenning. Reach set computations using real quantifier elimination. Technical Report MIP-0012, FMI, Universität Passau, D-94030 Passau, Germany, October 2004.
2. Lenore Blum. *Generalized Algebraic Structures: A Model Theoretical Approach*. Ph.D. thesis, MIT, Cambridge, MA, 1968.
3. Lenore Blum. Differentially closed fields: A model theoretic tour. In H. Bass, P. J. Cassidy, and J. Kovacic, editors, *Contributions to Algebra. A Collection of Papers Dedicated to Ellis Kolchin*, pages 37–61. Academic Press, New York, 1977.
4. George E. Collins and Hoon Hong. Partial cylindrical algebraic decomposition for quantifier elimination. *Journal of Symbolic Computation*, 12(3):299–328, September 1991.
5. Andreas Dolzmann. *Algorithmic Strategies for Applicable Real Quantifier Elimination*. Doctoral dissertation, Department of Mathematics and Computer Science. University of Passau, Germany, D-94030 Passau, Germany, March 2000.
6. Andreas Dolzmann and Thomas Sturm. Redlog: Computer algebra meets computer logic. *ACM SIGSAM Bulletin*, 31(2):2–9, June 1997.
7. Andreas Dolzmann and Thomas Sturm. Simplification of quantifier-free formulae over ordered fields. *Journal of Symbolic Computation*, 24(2):209–231, August 1997.
8. Andreas Dolzmann and Thomas Sturm. P-adic constraint solving. In Sam Dooley, editor, *Proceedings of the 1999 International Symposium on Symbolic and Algebraic Computation (ISSAC 99), Vancouver, BC*, pages 151–158. ACM Press, New York, NY, July 1999.
9. Andreas Dolzmann and Thomas Sturm. Redlog user manual. Technical Report MIP-9905, FMI, Universität Passau, D-94030 Passau, Germany, April 1999. Edition 2.0 for Version 2.0.
10. Andreas Dolzmann and Thomas Sturm. Parametric systems of linear congruences. In V. G. Ganzha, E. W. Mayr, and E. V. Vorozhtsov, editors, *Computer Algebra in Scientific Computing. Proceedings of the CASC 2000*, pages 149–166. Springer, Berlin, 2001. To appear.
11. David Hilbert. Über die vollen Invarianzsysteme. *Mathematische Annalen*, 42:313–373, 1893.
12. Valentin Irtegov and Tatyana Titorenko. On modeling the qualitative investigation of nonlinear systems with the aid of computer algebra. In Victor G. Ganzha, Ernst W. Mayr, and E. V. Vorozhtsov, editors, *Computer Algebra in Scientific Computing. Proceedings of the CASC 2003*, pages 199–212. TUM München, 2003.
13. E. J. McCluskey. Minimization of Boolean functions. *Bell Systems Technical Journal*, 35:1417–1444, April 1956.
14. W. V. Quine. The problem of simplifying truth functions. *American Mathematical Monthly*, 59:521–531, November 1952.
15. W. V. Quine. A way to simplify truth functions. *American Mathematical Monthly*, 62:627–631, November 1955.
16. W. V. Quine. On cores and prime implicants of truth functions. *American Mathematical Monthly*, 66:755–760, November 1959.
17. Joseph F. Ritt. *Differential Algebra*, volume 33 of *AMS Colloquium Publications*. American Mathematical Society, New York, NY, 1950.
18. Abraham Robinson. On the concept of a differentially closed field. In H. J. Kreisler, S. Körner, W. A. Luxemburg, and A. D. Young, editors, *Abraham Robinson. Selected Papers*, volume 1, pages 440–455. Yale University Press, New Haven and London, 1979. Originally in Bull. Res. Council Israel 8F (1959), 113–128. M.R. 23 #2323.
19. Abraham Robinson. Relative model-completeness and the elimination of quantifiers. In H. J. Kreisler, S. Körner, W. A. Luxemburg, and A. D. Young, editors, *Abraham Robinson. Selected Papers*, volume 1, pages 146–159. Yale University Press, New Haven and London, 1979. Originally in Dialectica 12 (1958) 394–407. M.R. 21 #1265.
20. Gerald E. Sacks. *Saturated Model Theory*. Mathematics Lecture Note Series. W. A. Benjamin, Inc., Reading, MA, 1972.
21. Abraham Seidenberg. An elimination theory for differential algebra. *University of California Publications in Mathematics. New Series*, 3:31–66, 1956.
22. Andreas Seidl and Thomas Sturm. A generic projection operator for partial cylindrical algebraic decomposition. In Rafael Sendra, editor, *Proceedings of the 2003 International Symposium on Symbolic and Algebraic Computation (ISSAC 03), Philadelphia, Pennsylvania*, pages 240–247. ACM Press, New York, NY, 2003.
23. Thomas Sturm. Linear problems in valued fields. *Journal of Symbolic Computation*, 30(2):207–219, August 2000.
24. Thomas Sturm and Volker Weispfenning. Quantifier elimination in term algebras. The case of finite languages. In Victor G. Ganzha, Ernst W. Mayr, and E. V. Vorozhtsov, editors, *Computer Algebra in Scientific Computing. Proceedings of the CASC 2002*, pages 285–300. TUM München, 2002.

25. Volker Weispfenning. The complexity of linear problems in fields. *Journal of Symbolic Computation*, 5(1&2):3–27, February–April 1988.
26. Volker Weispfenning. The complexity of almost linear diophantine problems. *Journal of Symbolic Computation*, 10(5):395–403, November 1990.
27. Volker Weispfenning. Comprehensive Gröbner bases. *Journal of Symbolic Computation*, 14:1–29, July 1992.
28. Volker Weispfenning. Quantifier elimination for real algebra—the quadratic case and beyond. *Applicable Algebra in Engineering Communication and Computing*, 8(2):85–101, February 1997.
29. Volker Weispfenning. A new approach to quantifier elimination for real algebra. In B.F. Caviness and J.R. Johnson, editors, *Quantifier Elimination and Cylindrical Algebraic Decomposition*, Texts and Monographs in Symbolic Computation, pages 376–392. Springer, Wien, New York, 1998.