A Combined VTS/Lazard Quantifier Elimination Method

James H. Davenport[1], Zak P. Tonks[1] and Ali K. Uncu[1,2]

[1] *University of Bath, Faculty of Science, Department of Computer Science, Bath UK*

[2] *Johann Radon Institute for Computational and Applied Mathematics, Austrian Academy of Science, Altenbergerstraße 69, A-4040 Linz, Austria*

Abstract

Many nonlinear optimisation problems involving systems of multivariate polynomial constraints can be expressed as Quantifier Elimination problems. In this regard, the need to eliminate quantifiers presents itself in many real problems such as in economics, mechanics, and motion planning. We focus on using Cylindrical Algebraic Decomposition and Virtual Term Substitution simultaneously. In this paper, we introduce the state-of-the-art theory behind an implementation of these methods produced in collaboration with Maplesoft .

Keywords

Quantifier Elimination, Cylindrical Algebraic Decomposition, Virtual Term Substitution, Equational Constraints

1. Introduction

Quantifier Elimination (QE) over the real numbers refers to problems in Computer Algebra of producing a quantifier-free equivalent representation of a logical expression defined by polynomial constraints. Quantifier Elimination is highly relevant and necessary for many applications from different fields of study. Nonlinear optimisation problems and conditions on solvability and satisfiability of systems of multivariate polynomial equations and inequalities can canonically be presented as QE problems. QE problems are known to arise in economics [1, 2], mechanics [3], mathematical biology [4, 5, 6], reaction networks [7], AI to pass mathematical exams [8], and motion planning [9].

In this paper, we introduce the theoretic background that is inside the QE implementation of the second author as a part of his Ph.D. studies under the guidance of the first author [10]. This Maple implementation, the package `QuantifierElimination`, is now being integrated to the main program by the second author and it will be available to all Maple users without the need of downloading and installing a third party package. One main aim of this implementation is to mitigate the high running times of QE problems by avoiding unnecessary and/or complex computations by using a poly-algorithmic approach to QE to pick the route of expected least

6^{th} *International Workshop on Satisfiability Checking and Symbolic Computation, 19–20 Aug 2021, College Station U.S.*
masjhd@bath.ac.uk (J. H. Davenport); zak.p.tonks@bath.edu (Z. P. Tonks); aku21@bath.ac.uk (A. K. Uncu)
https://people.bath.ac.uk/masjhd/ (J. H. Davenport); https://people.bath.ac.uk/zpt20/ (Z. P. Tonks);
http://akuncu.com (A. K. Uncu)
0000-0002-3982-7545 (J. H. Davenport); 0000-0001-6019-1775 (Z. P. Tonks); 0000-0001-5631-6424 (A. K. Uncu)

resistance. Moreover, the poly-algorithm offers full incrementality as a whole. Interested readers can find more about the implementation and its design decisions in [10, 11]. In particular [10, §7.4] has the benchmarking data.

2. Definitions

A logical formula of integral multivariate polynomial constraints, $p_i(x_1, \ldots, x_n)\sigma_i 0$ for some $i \in \mathbb{N}$ and $\sigma_i \in \{=, \neq, >, \geq, <, \leq\}$, joined together with logical connectives $\wedge$, and $\vee$ is called a *Tarski formula*. A single polynomial constraint in a Tarski formula is called an *atom*. Finally, the language of all semi-algebraic propositions is called a *Tarski-language*.

Given a Tarski formula $F(x_1, \ldots, x_n)$, and a list of quantifiers on $x_1, \ldots, x_k$ for some integer $1 \leq k \leq n$. A (prenex) *quantified* Tarski formula is

$$Q_1 x_1\ Q_2 x_2 \ldots Q_k x_k F(x_1, \ldots, x_k, \ldots, x_n), \tag{1}$$

where $Q_i = \forall$ or $\exists$. The QE problem is then two-fold: Is there a quantifier-free Tarski formula $G(x_{k+1}, \ldots, x_n)$ equivalent to (1)? And, if so, can it be computed?

In 1951, Tarski showed that the real quantifier elimination problem is decidable for Tarski formulas involving multivariate polynomials with rational coefficients. Moreover, he gave an algorithm (insuperable complexity) to calculate the equivalent quantifier-free formula.

Collins [12] produced a QE algorithm through calculating a sign-invariant *Cylindrical Algebraic Decomposition* (CAD) in 1975, and proved that, starting with a Tarski formula with m degree d polynomials in n variables, CAD computations have doubly exponential complexity: $d\hat{\ }2\hat{\ }(2n + \mathcal{O}(1))$. This is due to the need of squaring the degree d of the polynomials and their number m at each one of the n variable elimination steps. McCallum [13] showed that, under some assumptions that certain polynomials don't vanish on some varieties, the doubly exponential complexity of calculating a order-invariant CAD of a given Tarski formula can be lowered to $d\hat{\ }2\hat{\ }(n + \mathcal{O}(1))$. In 1994, Lazard [14] suggested a new procedure for projecting and lifting for lex-least invariant CAD calculations that steers clear of the assumptions needed in McCallum's procedure, and in [15], McCallum–Parusiński–Paunescu justified the correctness of Lazard's procedure, and the same $d\hat{\ }2\hat{\ }(n + \mathcal{O}(1))$ complexity. The first author and Heintz [16] have shown that doubly-exponential complexity is intrinsic to quantifier elimination, admittedly with a double exponent lower bound of $n/5 + \mathcal{O}(1)$, later improved to $n/3 + \mathcal{O}(1)$ by [17].

We should also note that the degrees of the polynomials and the number of polynomials in a Tarski formula are not the only sources of the doubly exponential complexity of quantifier elimination. Given a quantified Tarski formula, let a be the number of alterations of quantifiers. For example, $a = 3$ if we have the quantifiers in the $\forall\forall\exists\forall\exists\exists\exists$ order. Brown and the first author [17] have provided an example involving only linear polynomials that manifests a doubly exponential behaviour in a, caused (in the case of VTS) by the repeated conversion between conjunctive and disjunctive normal form forced by the alternations: see the remark at the start of §4.

Starting from a different origin, Weispfenning introduced the technique of *Virtual Term Substitution* (VTS) for purely existential QE problems for linear [18], quadratic [19] and cubic [20] polynomials. Later in his doctoral studies Košta [21] wrote down explicit algorithms for VTS for all degrees $d \leq 3$.

3. Quantifier Elimination by Projection and Lifting

This was originated in [12]. An improvement, but which could occasionally deliver a result flagged as "potentially incomplete", was in [13]. A solution was suggested by Lazard in [14], but only proved in [15]. It uses lexicographically-least (lex-least) invariance rather than McCallum's order-invariance. The projection phases of both approaches are similar. A development, skipping trailing coefficients in projection in many cases, was introduced in [22].

3.1. Lazard Projection and Lifting

The lex-least invariance means that we require polynomials to have a certain expansion on varieties. For a given multivariate polynomial $f(x_1, \ldots, x_n)$ and a point $\mathbf{a} = (a_1, \ldots, a_n)$, we want the lex-least monomial of f to have the expansion

$$c_{\mathbf{a}}(x_1 - a_1)^{\alpha_1}(x_2 - a_2)^{\alpha_2} \ldots (x_n - a_n)^{\alpha_n}, \tag{2}$$

where $c_{\mathbf{a}}$ is a non-zero integer and α_is are non-negative integers. This selection creates a big difference in the lifting phase. If f becomes zero for a trail of variable assignments $\mathbf{a}$, i.e. f vanishes on some variety that $\mathbf{a}$ is in, the McCallum procedure would need to come to a halt. On the other hand, for Lazard lifting, one can use the lex-least invariance and replace f with $c_{\mathbf{a}}f$ divided by (2). The outcome polynomial still has integer coefficients and it is in the Tarski language. It is easy to see that the outcome polynomial is non-zero at $\mathbf{a}$ and one can proceed with the lifting procedure.

The guaranteed lex-least factorisation and replacing a nullified polynomial by its zero-free part is a clear advantage of Lazard lifting. However, this should not create a false image. There are still cases where nullification of polynomials are a problem, especially when we have equational constraints (§3.2) in the Tarski formulae. More on these special cases and relevant research can be found in [23, 24].

Most projection-lifting[1] CAD implementations available to the public today use McCallum's projection and lifting procedure to calculate CADs. To the best of our knowledge, the second author's implementation is the first QE implementation that uses Lazard's procedure in CAD calculations. This is certainly true for an implementation made in the Maple language.

3.2. Equational Constraints

In this section we assume that we start with a Tarski formula made up of m polynomial constraints where exactly c of these constraints are equations:

$$f_1 = 0 \wedge f_2 = 0 \wedge f_c = 0 \wedge g_1\sigma_1 0 \wedge g_2\sigma_2 0 \wedge \cdots \wedge g_{m-c}\sigma_{m-c}0, \tag{3}$$

where all of the polynomials are from the polynomial ring $\mathbb{Z}[x_1, \ldots, x_n]$ and $\sigma_i \in \{\neq, <, \leq, >, \geq\}$. The atoms in the Tarski formula with the equations are called *equational constraints*.

Collins [26] raised the question of whether an single equational constraint ($c = 1$ in (3)) in a Tarski formula be used to our advantage in CAD calculations. In 1999 for $c = 1$ case and in 2001

[1] As opposed to, say, Regular Chains approaches [25].

for general c, McCallum [27, 28] showed that the presence of c equational constraints reduces the double exponential complexity $m\hat{}2\hat{}(n + \mathcal{O}(1))$ drastically to $m\hat{}2\hat{}(n - c + \mathcal{O}(1))$.

We would also like to note that more improvements and extensions has been made on treating equational constraints. The first author in collaboration with others [29] generalised the use of equational constraints when only a part of the formula has equational constraints. In 2020, it was shown that Gröbner bases can be used rather than just using iterated resultants [30] and that this approach reduces the polynomial degree growth. Generically, this lowers the doubly exponential complexity in the polynomial degrees to $d\hat{}2\hat{}(n - c + \mathcal{O}(1))$.

These results make a clear case for their benefit. In essence the equational constraint treatment reduces the doubly exponential complexity of CAD calculations significantly by removing the number of such equalities from the Tarski formula. It should still be noted that all these mentioned results are for the McCallum projection and lifting procedure and they require well-orientedness assumptions.

The second author's implementation, on the other hand, uses the Lazard's procedure and therefore cannot yet enjoy all the simplifications that equational constraints have to offer. The research on equational constraints while using Lazard's approach is at its early stages. At the time of implementation, by the Ph.D. work of Nair [24] we only knew how to process a single equational constraint in Lazard-based CAD computations (this has now been extended). Moreover, we require the polynomial in the equational constraint to satisfy some non-vanishing properties on varieties. The QE implementation of the second author handles Tarski formulae with a single equational constraint, and it has the capabilities to notice whether this equational constraints satisfies the assumptions necessary coming from Nair's conditions. Moreover, the `QuantifierElimination` package implements bespoke routines Nair outlined [24] for the recovery of problematic cases once such a problem is identified. On top of it all, this software takes full advantage of the avoidance and recovery of the problematic cases by employing Collins and Hong's partial CAD [31] construction.

4. Virtual Term Substitution (VTS)

Virtual Term Substitution (VTS) is a method with geometric roots to handle QE problems where the Tarski formulae are purely existentially quantified: see [18]. One can still apply VTS to a universally quantified problem by logically negating the QE problem first: $\forall x P(x) \equiv \neg\exists x \neg P(x)$. This is a step that can give rise to complexity growth: $\exists x \neg P(x)$ is a disjunction over all the guards (see (4)), which will be turned by $\neg$ into a conjunction, and conversion to disjunctive normal form can cause exponential blowup.

Consider an existentially quantified Tarski formula $\exists x F(\mathbf{u}, x)$, where explicit negations in F have been eliminated by De Morgan's Laws.. The aim of VTS is to compute a finite list of conditions (called *guards*) and relative virtual substitution points for the existentially quantified variable x so that QE can be performed by turning the problem replacing x under the conditions of the guards. More precisely, VTS tries to split the x-axis into all relevant intervals and tries to identify a finite set $E = \{(\gamma_i, e_i)\}$ of pairs where γ_i are guard conditions and e_i are sample points from each interval on the real axis. Under these conditions we can see the following correspondence

$$\exists x F(\mathbf{u}, x) \Leftrightarrow \bigvee_{(\gamma_i, e_i) \in E} (\gamma_i \wedge F(\mathbf{u}, x // e_i)) . \tag{4}$$

Definition 1 (Virtual Substitution). *[21, Section 2.3] The virtual substitution of a structural test point $T(\mathbf{u})$ for x into a quantifier free relation $g(\mathbf{u}, x)\sigma 0$ is $F(\mathbf{u}) := g[x//T]$ a quantifier free formula such that for any parameter values $\mathbf{a} \in \mathbb{R}^{n-1}$, if $\mathbf{a}$ satisfies the guard of T, then $\mathbf{a}$ satisfies F if and only if $g(\mathbf{a}, T(\mathbf{a}))\sigma 0$ is true. $T(\mathbf{a})$ is well-defined because the guard is satisfied.*

In addition, we have to deal with virtually substituting $\pm\infty$: see [21, p. 36].

4.1. Example

We would like to exemplify this idea on a simple case with a single existential quantifier. Assume that we start with the simple quantified Tarski formula

$$\exists x\, (ax - b = 0 \wedge cx + d > 0) . \tag{5}$$

Naïvely, one can solve this QE problem by algebraically (and carelessly) solving the first atom of the Tarski formula as $x = b/a$ (to be our virtual substitution point) and by plugging this x value in the second atom:

$$c(b/a) + d > 0. \tag{6}$$

This simple approach ignores many possibilities that requires attention. As it is given, (6) is only meaningful under the guard that $a \neq 0$. Furthermore, the Tarski language does not include rational values therefore clearing any denominators is necessary. Even at this step one needs to consider that a is either negative or positive, therefore clearing denominators might change the inequality. We can instead clear the denominators by multiplying each side of (6) with a^2 to keep the constraint unchanged. These considerations give us one of the right-hand sides cases in (4) as

$$(a \neq 0 \wedge acb + a^2 d > 0). \tag{7}$$

Remark that (7) is not equivalent to (5). For that one also needs other conditions such as

$$(a = 0 \wedge b = 0 \wedge c \neq 0) \vee (a = 0 \wedge b = 0 \wedge c = 0 \wedge d > 0). \tag{8}$$

The Tarski formulae (7) and (8) put together with disjunction is logically equivalent to (5) and it is free of the quantified variable x.

4.2. VTS in general

It is easy to see that once the VTS points E are established, an $\exists x$ quantifier is eliminated by moving from the problem on the left-hand side of (4) to the right-hand side. Having said that, it is not always possible to identify the set E. Weispfenning [18, 19] gave us E for the linear and quadratic cases, and [20] outlined the cubic case.

Including Košta's [21] detailed algorithms for cubic VTS and the corrections in Appendix A, this method is implemented in second author's QE package. This QE implementation is believed to be the first in Maple, and certainly the first in Maple to include the degree 3 VTS algorithms.

4.3. Degree Growth in VTS

Although at the start all the variables in the Tarski formula (5) appeared linearly, after the suggested virtual substitution and simplification (7), we saw that the degree of variable a increased. This hints shows successive runs of VTS might affect the applicability of VTS since this method requires the degrees of the variables to be ≤ 3. For example, let's start with the quantified Tarski formula

$$\exists a \exists x \left(a^3 x - b = 0 \wedge cx + d > 0\right) \tag{9}$$

similar to (5). The application of VTS to the variable x, as it was done for (5), yields

$$\begin{aligned} \exists a \big((a^3 \neq 0 \wedge a^3 cb + a^6 d > 0) \vee (a^3 = 0 \wedge b = 0 \wedge c \neq 0) \\ \vee (a^3 = 0 \wedge b = 0 \wedge c = 0 \wedge d > 0)\big) \,. \end{aligned} \tag{10}$$

The degree of the variable a in the first Tarski formula of (10) is 6. Therefore, without a change of variables, VTS can no longer be employed for elimination of the existential quantifier on a. Such a problem can be avoided by a better variable ordering. We look at the following Tarski formula:

$$\exists x \exists a \left(a^3 x - b = 0 \wedge cx + d > 0\right) , \tag{11}$$

which is equivalent to (9). After applying VTS for variable a, we get

$$\exists x \left((x \neq 0 \wedge cx + d > 0) \vee (x = 0 \wedge b = 0 \wedge cx + d > 0)\right) . \tag{12}$$

This time it is possible to apply VTS to all parts of (10) to eliminate the quantifier on x (as this variable appears with degree ≤ 3 in the output). Hence, we see that the variable ordering can have an effect on the applicability of VTS. This is a characteristic difference between VTS and CAD; in CAD, the order of variables can affect the running time heavily but the variable ordering never renders this method inapplicable.

5. Conclusion and Future Directions

The poly-algorithmic QE implementation is a promising tool that is at its early stages. With its various variable ordering strategies and QE methods it already is a sophisticated system that offers Maple users a lot of options. Moreover, any research done, especially on Lazard's CAD construction procedure, would directly benefit this implementation. For that, it is of interest to us to compare and understand the strengths of McCallum and Lazard's procedures against each other and to find ways to identify the situations where these approaches fail to yield answers. It is also a top priority to develop theory around equational constraints using Lazard's projection/lifting that is analogous to the theory already developed around McCallum's procedure.

Neither the guards nor the virtual sample points are unique; we are optimistic that a different splitting of a given problem (such as (5)) can help to keep the variable degrees low. Not only that, similar to the CAD case, the variable ordering in VTS can be relevant when the QE problem has multiple existential constraints in a block (see §4.3). More experiments on different guard selections and variable orderings are needed to make full use of virtual term substitutions.

6. Acknowledgement

The authors are grateful to Marek Košta and Jürgen Gerhard for correspondence about Appendix A. The first author would like to thank the EPSRC grant number EP/T015713/1 for supporting his research. The second author would like to thank EPSRC and Maplesoft for funding his research studentship. The third author would like to thank the EPSRC grant number EP/T015713/1 and the FWF grant P-34501N for partially supporting his research.

References

[1] C. Mulligan, R. Bradford, J. Davenport, M. England, Z. Tonks, Quantifier Elimination for Reasoning in Economics, https://arxiv.org/abs/1804.10037, 2018.

[2] C. Mulligan, J. Davenport, M. England, TheoryGuru: A Mathematica Package to Apply Quantifier Elimination Technology to Economics, in: J. Davenport, M. Kauers, G. Labahn, J. Urban (Eds.), Proceedings Mathematical Software — ICMS 2018, 2018, pp. 369–378.

[3] N. Ioakimidis, Sharp bounds based on quantifier elimination in truss and other applied mechanics problems with uncertain, interval forces/loads and other parameters, Technical Report TR-2019-Q7 University of Patras, 2019.

[4] C. Brown, M. El Kahoui, D. Novotni, A. Weber, Algorithmic methods for investigating equilibria in epidemic modeling, J. Symbolic Comp. 41 (2006) 1157–1173.

[5] G. Röst, A. Sadeghimanesh, Exotic Bifurcations in Three Connected Populations with Allee Effect, International Journal of Bifurcation and Chaos Issue 13 art. no. A36 31 (2021).

[6] C. Chauvin, M. Müller, A. Weber, An application of quantifier elimination to mathematical biology, In Computer Algebra in Science and Engineering (Fleischer (1995) 287–296.

[7] H. Rahkooy, T. Sturm, Parametric Toricity of Steady State Varieties of Reaction Networks, in: Proceedings CASC 2021: Computer Algebra in Scientific Computing, 2021, pp. 314–333.

[8] Y. Wada, T. Matsuzaki, A. Terui, N. Arai, An Automated Deduction and Its Implementation for Solving Problem of Sequence at University Entrance Examination, in: Proceedings ICMS 2016, 2016, pp. 82–92.

[9] D. Wilson, R. Bradford, J. Davenport, M. England, A "Piano Movers" Problem Reformulated, Technical Report CSBU-2013-03 Department of Computer Science University of Bath, 2013.

[10] Z. Tonks, Poly-algorithmic Techniques in Real Quantifier Elimination, Ph.D. thesis, University of Bath, 2021. URL: https://researchportal.bath.ac.uk/en/studentTheses/poly-algorithmic-techniques-in-real-quantifier-elimination.

[11] Z. Tonks, A Poly-algorithmic Quantifier Elimination Package in Maple, in: J. Gerhard, I. Kotsireas (Eds.), Maple in Mathematics Education and Research 2019, volume 1125 of *Communications in Computer and Information Science*, 2020, pp. 171–186.

[12] G. Collins, Quantifier Elimination for Real Closed Fields by Cylindrical Algebraic Decomposition, in: Proceedings 2nd. GI Conference Automata Theory & Formal Languages, 1975, pp. 134–183.

[13] S. McCallum, An Improved Projection Operation for Cylindrical Algebraic Decomposition, Ph.D. thesis, University of Wisconsin-Madison Computer Science, 1984.

[14] D. Lazard, An Improved Projection Operator for Cylindrical Algebraic Decomposition, in:

C. Bajaj (Ed.), Proceedings Algebraic Geometry and its Applications: Collections of Papers from Shreeram S. Abhyankar's 60th Birthday Conference, 1994, pp. 467–476.
[15] S. McCallum, A. Parusiński, L. Paunescu, Validity proof of Lazard's method for CAD construction, J. Symbolic Comp. 92 (2019) 52–69.
[16] J. Davenport, J. Heintz, Real Quantifier Elimination is Doubly Exponential, J. Symbolic Comp. 5 (1988) 29–35.
[17] C. Brown, J. Davenport, The Complexity of Quantifier Elimination and Cylindrical Algebraic Decomposition, in: C. Brown (Ed.), Proceedings ISSAC 2007, 2007, pp. 54–60.
[18] V. Weispfenning, The Complexity of Linear Problems in Fields, J. Symbolic Comp. 5 (1988) 3–27.
[19] V. Weispfenning, Quantifier elimination for real algebra — the quadratic case and beyond, AAECC 8 (1997) 85–101.
[20] V. Weispfenning, Quantifier elimination for real algebra — the cubic case, in: Proceedings ISSAC 1994, 1994, pp. 258–263.
[21] M. Košta, New concepts for real quantifier elimination by virtual substitution, Ph.D. thesis, Universität des Saarlandes, 2016.
[22] C. Brown, S. McCallum, Enhancements to Lazard's Method for Cylindrical Algebraic Decomposition, in: F. Boulier, M. England, T. Sadykov, E. Vorozhtsov (Eds.), Computer Algebra in Scientific Computing CASC 2020, volume 12291 of *Springer Lecture Notes in Computer Science*, 2020, pp. 129–149. doi:https://doi.org/10.1007/978-3-030-60026-6_8.
[23] A. Nair, J. Davenport, G. Sankaran, Curtains in CAD: Why Are They a Problem and How Do We Fix Them?, in: A. Bigatti, J. Carette, J. Davenport, M. Joswig, T. de Wolff (Eds.), Mathematical Software — ICMS 2020, volume 12097 of *Springer Lecture Notes in Computer Science*, Springer, 2020, pp. 17–26.
[24] A. Nair, Curtains in Cylindrical Algebraic Decomposition, Ph.D. thesis, University of Bath, 2021. URL: https://researchportal.bath.ac.uk/en/studentTheses/curtains-in-cylindrical-algebraic-decomposition.
[25] R. Bradford, C. Chen, J. Davenport, M. England, M. Moreno Maza, D. Wilson, Truth Table Invariant Cylindrical Algebraic Decomposition by Regular Chains, in: Proceedings CASC 2014, 2014, pp. 44–58.
[26] G. Collins, Quantifier elimination by cylindrical algebraic decomposition — twenty years of progess, in: B. Caviness, J. Johnson (Eds.), Quantifier Elimination and Cylindrical Algebraic Decomposition, Springer Verlag, Wien, 1998, pp. 8–23.
[27] S. McCallum, On Projection in CAD-Based Quantifier Elimination with Equational Constraints, in: S. Dooley (Ed.), Proceedings ISSAC '99, 1999, pp. 145–149.
[28] S. McCallum, On Propagation of Equational Constraints in CAD-Based Quantifier Elimination, in: B. Mourrain (Ed.), Proceedings ISSAC 2001, 2001, pp. 223–230.
[29] R. Bradford, J. Davenport, M. England, S. McCallum, D. Wilson, Truth table invariant cylindrical algebraic decomposition, J. Symbolic Comp. 76 (2016) 1–35.
[30] M. England, R. Bradford, J. Davenport, Cylindrical Algebraic Decomposition with Equational Constraints, in: J. Davenport, M. England, A. Griggio, T. Sturm, C. Tinelli (Eds.), Symbolic Computation and Satisfiability Checking: special issue of J. Symbolic Computation, volume 100, 2020, pp. 38–71.
[31] G. Collins, H. Hong, Partial Cylindrical Algebraic Decomposition for Quantifier Elimina-

tion, J. Symbolic Comp. 12 (1991) 299–328.

A. Correction to [21]

There is a correction necessary with respect to [21, p. 58]. The setting is that f is a cubic, with a root a defined by f and an index S. We wish to investigate $g(a)$, where g is linear or quadratic in a (the highest degrees necessary when a is defined by a cubic). Let ρ^* be $\leq$ if ρ is $<$, and $<$ if ρ is $\leq$.

Step 3.1 the value assigned to ψ_- should be $\neg(-g\rho^*0)[x//(f,S)]$ in the cases ρ is not $=$.

Step 4 has a typo: instead of $\rho \in \{=,<,\leq\}$ it should be $\rho \in \{\neq,>,\geq\}$.

Step 4.1 An analogous change to 3.1, but the way the code in [10] is written, this case is never reached, provided that the $g \neq 0$ case is handled separately.