Satisfiability of Non-Linear Arithmetic over Algebraic Numbers[*]

Harald Zankl René Thiemann Aart Middeldorp
Institute of Computer Science, University of Innsbruck, 6020 Innsbruck, Austria

Abstract

We propose an (incomplete) approach for satisfiability of non-linear arithmetic over algebraic real numbers by reducing the problem to non-linear arithmetic over the rationals/integers.

1 Introduction

The first order theory of non-linear arithmetic over the reals has been shown to be decidable by Tarski [2]. This decision procedure is of non-elementary computational complexity. Improvements of the original procedure are still of double exponential time complexity [1]. Existing implementations of these procedures can only cope with small instances. It is well-known that many termination criteria for term rewriting can be encoded in SAT/SMT. Typically such instances contain hundreds of variables and (tens of) thousands of arithmetic operations. In this note we suggest an alternative method for checking satisfiability of real arithmetic, capable of handling large instances. Our approach cannot be used to deduce unsatisfiability of arithmetic constraints since only (a subset of) algebraic real numbers are covered.

The remainder of this paper is organized as follows. In Section 2 we introduce the syntax of non-linear arithmetic constraints. Then Section 3 lists the encodings for the arithmetic operations while Section 4 shows their soundness (i.e., if the encoding evaluates to true then the original constraint is satisfiable). Section 5 then assesses our contribution while Section 6 concludes.

2 Syntax

In this section we fix the syntax for non-linear arithmetic constraints.

Definition 1. An *arithmetic constraint* φ is described by the BNFs

$$\varphi ::= \bot \mid \top \mid p \mid (\neg\varphi) \mid (\varphi \circ \varphi) \mid (\alpha \star \alpha) \quad \text{and} \quad \alpha ::= \mathrm{a} \mid r \mid (\alpha \diamond \alpha) \mid (\varphi\,?\,\alpha : \alpha)$$

where $\circ \in \{\vee, \wedge, \rightarrow, \leftrightarrow\}$, $\star \in \{>, =\}$, and $\diamond \in \{+, -, \times\}$.

Here $\bot$ ($\top$) denotes contradiction (tautology), p (a) ranges over Boolean (arithmetical) variables, $\neg$ ($\vee$, $\wedge$, $\rightarrow$, $\leftrightarrow$) is logical not (or, and, implication, bi-implication), $>$ ($=$) greater (equal), r ranges over the real numbers, and $+$ ($-$, $\times$) denotes addition (subtraction, multiplication). If-then-else is written as $(\cdot\,?\,\cdot : \cdot)$. The following example shows some (non-)well-formed constraints.

Example 2. The expressions 5, p_{100}, $(p_{10}\,?\,(2.1 \times \mathrm{a}_{12}) : 0)$, and $((((\mathrm{a}_{12} + (\sqrt{2} \times \mathrm{a}_{30})) + 7.2) > (0 - \mathrm{a}_5)) \wedge p_2)$ are well-formed whereas $-\mathrm{a}_{10}$ (unary minus) and $3 + \mathrm{a}$ (parentheses missing) are not.

The binding precedence $\times \succ +,- \succ >,= \succ \neg \succ \vee,\wedge \succ \rightarrow,\leftrightarrow,(\cdot\,?\,\cdot:\cdot)$ allows to save parentheses. Furthermore the operators $+$, $-$, $\times$, $\vee$, $\wedge$, and $\leftrightarrow$ are left-associative while $\rightarrow$ associates to the right. Taking these conventions into account the most complex constraint from the previous example simplifies to $\mathrm{a}_{12} + \sqrt{2} \times \mathrm{a}_{30} + 7.2 > 0 - \mathrm{a}_5 \ \wedge\ p_2$. To obtain smaller constraints already at the time of encoding trivial simplifications like $\varphi \wedge \top \rightarrow \varphi$, $\varphi \wedge \bot \rightarrow \bot$, $\cdots$ are performed.

Example 3. The constraint $\varphi := \mathrm{a}_1 \times \mathrm{a}_1 \times \mathrm{a}_1 = 3 \ \wedge\ \mathrm{a}_2 \times \mathrm{a}_2 = 2 \ \wedge\ \mathrm{a}_3 + \mathrm{a}_3 = 1$ is satisfiable. The assignment α with $\alpha(\mathrm{a}_1) = \sqrt[3]{3}$, $\alpha(\mathrm{a}_2) = \sqrt{2}$, and $\alpha(\mathrm{a}_3) = \frac{1}{2}$ is a model for φ.

[*] This research is supported by FWF (Austrian Science Fund) project P18763.

3 Reducing Reals to Integers

Next we show how to find models for constraints like the one in Example 3 automatically. To this end we extend the approach of dealing with algebraic real numbers from [3]. The idea is to reduce satisfiability of *real* arithmetic to arithmetic over *rational* numbers. A fast approach for the latter is introduced in [3]. To represent a real-valued variable we use pairs $([c_1,\dots,c_n],m)$ (or in short $(\mathbf{c}_n,m)$) where the first element is a list of integer (or rational) valued variables and m is a variable taking integral values larger than one. The intended semantics of $(\mathbf{c}_n,m)$ is $c_1\sqrt[n]{m^0}+_{\mathbb{R}}\cdots+_{\mathbb{R}}c_n\sqrt[n]{m^{n-1}}$. Hence this approach can only represent (a subset of) algebraic numbers. For instance, $\sqrt[3]{1+\sqrt{2}}$ is not represented.

In the sequel we show that arithmetic operations can easily be encoded. To avoid confusion we denote the set of real (rational) numbers by $\mathbb{R}$ ($\mathbb{Q}$) and encodings of such numbers by **R** (**Q**). Similarly a subscript $_\mathbf{R}$ ($_\mathbf{Q}$) is used to denote the encodings of arithmetic operations.

Definition 4. For $(\mathbf{c}_n,m)$ and $(\mathbf{d}_n,m)$ from **R** we define:

$$\begin{aligned}(\mathbf{c}_n,m)+_{\mathbf{R}}(\mathbf{d}_n,m) &:= ([c_1+_{\mathbf{Q}}d_1,\dots,c_n+_{\mathbf{Q}}d_n],m)\\ (\mathbf{c}_n,m)-_{\mathbf{R}}(\mathbf{d}_n,m) &:= ([c_1-_{\mathbf{Q}}d_1,\dots,c_n-_{\mathbf{Q}}d_n],m)\end{aligned}$$

The following example demonstrates addition for encodings of reals. To distinguish numbers from encodings of numbers the latter are represented in boldface.

Example 5. The computation $([\mathbf{1},\mathbf{2}],\mathbf{3})+_{\mathbf{R}}([\mathbf{5},\mathbf{3}],\mathbf{3})=([\mathbf{1}+_{\mathbf{Q}}\mathbf{5},\mathbf{2}+_{\mathbf{Q}}\mathbf{3}],\mathbf{3})=([\mathbf{6},\mathbf{5}],\mathbf{3})$ is valid since the left-hand side encodes $1+2\sqrt{3}+5+3\sqrt{3}$ and reduces to $6+5\sqrt{3}$ corresponding to the right-hand side.

Next we focus on multiplication.

Definition 6. For $(\mathbf{c}_n,m)$ and $(\mathbf{d}_n,m)$ from **R** we define:

$$(\mathbf{c}_n,m)\times_{\mathbf{R}}(\mathbf{d}_n,m) := (((\mathbf{c}_n,m)\cdot d_1)\gg 0)+_{\mathbf{R}}\cdots+_{\mathbf{R}}(((\mathbf{c}_n,m)\cdot d_n)\gg(n-1))$$

where

$$\begin{aligned}(\mathbf{c}_n,m)\cdot d &:= ([c_1\times_{\mathbf{Q}}d,\dots,c_n\times_{\mathbf{Q}}d],m)\\ (\mathbf{c}_n,m)\gg 0 &:= (\mathbf{c}_n,m)\\ (\mathbf{c}_n,m)\gg(i+1) &:= ([m\times_{\mathbf{Q}}c_n,c_1,\dots,c_{n-1}],m)\gg i\end{aligned}$$

Example 7. The computation

$$\begin{aligned}([\mathbf{1},\mathbf{2}],\mathbf{2})\times_{\mathbf{R}}([\mathbf{5},\mathbf{3}],\mathbf{2}) &= (([\mathbf{5},\mathbf{10}],\mathbf{2})\gg 0)+_{\mathbf{R}}(([\mathbf{3},\mathbf{6}],\mathbf{2})\gg 1)\\ &= ([\mathbf{5},\mathbf{10}],\mathbf{2})+_{\mathbf{R}}([\mathbf{12},\mathbf{3}],\mathbf{2})\\ &= ([\mathbf{17},\mathbf{13}],\mathbf{2})\end{aligned}$$

is justified by $(1+2\sqrt{2})\times(5+3\sqrt{2})=5+10\sqrt{2}+3\sqrt{2}+6\sqrt{2}\sqrt{2}=17+13\sqrt{2}$.

Next we encode comparisons. Here $\mathbf{c}_k=[c_1,\dots,c_k]$ whenever $\mathbf{c}_{k+1}=[c_1,\dots,c_k,c_{k+1}]$.

Definition 8. For $(\mathbf{c}_n,m)$ and $(\mathbf{d}_n,m)$ from **R** we define:

$$\begin{aligned}(\mathbf{c}_n,m)=_{\mathbf{R}}(\mathbf{d}_n,m) &:= (c_1=_{\mathbf{Q}}d_1)\wedge\cdots\wedge(c_n=_{\mathbf{Q}}d_n)\\ (\mathbf{c}_n,m)>_{\mathbf{R}}(\mathbf{d}_n,m) &:= (\mathbf{c}_n,m)>^0_{\mathbf{R}}(\mathbf{d}_n,m)\\ (\mathbf{c}_0,m)>^a_{\mathbf{R}}(\mathbf{d}_0,m) &:= a>_{\mathbf{Q}}0\\ (\mathbf{c}_{k+1},m)>^a_{\mathbf{R}}(\mathbf{d}_{k+1},m) &:= c_{k+1}+_{\mathbf{Q}}a\geqslant_{\mathbf{Q}}d_{k+1}\ \wedge\ (\mathbf{c}_k,m)>^{a+(c_{k+1}-d_{k+1})}_{\mathbf{R}}(\mathbf{d}_k,m)\end{aligned}$$

The next example shows that $=_{\mathbf{R}}$ and $>_{\mathbf{R}}$ only approximate $=_{\mathbb{R}}$ and $>_{\mathbb{R}}$, respectively. The intuition behind $>_{\mathbf{R}}$ is also demonstrated in the example.

Example 9. First we show that $=_{\mathbf{R}}$ is only an approximation of $=_{\mathbb{R}}$. The problem is that the representation of numbers need not be canonical, e.g., $([\mathbf{2},\mathbf{0}],\mathbf{4}) \neq_{\mathbf{R}} ([\mathbf{0},\mathbf{1}],\mathbf{4})$ since the lists are not equal componentwise but $2+0\sqrt{4} =_{\mathbb{R}} 0+1\sqrt{4}$. This poses problems when tests for equality appear at negative positions in Boolean formulas (cf. Section 4).

Next we show that $>_{\mathbf{R}}$ is only an approximation of $>_{\mathbb{R}}$ which is not only due to representation issues. Obviously $5+1\sqrt{2} \approx 6.41 >_{\mathbb{R}} 6.24 \approx 2+3\sqrt{2}$ holds. But we have $([\mathbf{5},\mathbf{1}],\mathbf{2}) \not>_{\mathbf{R}} ([\mathbf{2},\mathbf{3}],\mathbf{2})$ since $\mathbf{1} \not>_{\mathbf{Q}} \mathbf{3}$. On the other hand $([\mathbf{2},\mathbf{4}],\mathbf{2}) >_{\mathbf{R}} ([\mathbf{3},\mathbf{2}],\mathbf{2})$ which represents $2+4\sqrt{2} >_{\mathbb{R}} 3+2\sqrt{2}$. Since $4 \geqslant_{\mathbb{R}} 2$ also $4\sqrt{2} \geqslant_{\mathbb{R}} 2\sqrt{2}$. The leftover $2\sqrt{2}$ on the left-hand side is clearly greater than 2 which is used to obtain $2+2 >_{\mathbb{R}} 3$.

We note that $(\mathbf{c}_n,m) >_{\mathbf{R}} (\mathbf{d}_n,m)$ can be solved exactly if $n=2$, i.e., when only square roots occur. The idea is to replace comparisons like $c_1+c_2\sqrt{m} >_{\mathbb{R}} d_1+d_2\sqrt{m}$ by $c_1-d_1+(c_2-d_2)\sqrt{m} >_{\mathbb{R}} 0$. The latter can be encoded exactly based on a case analysis on the operands' signs while squaring the inequality.

In the final definition we exactly characterize $([c_1,c_2],m) >_{\mathbf{R}} 0$.

Definition 10. For the pair $([c_1,c_2],m)$ from $\mathbf{R}$ we define

$$\begin{aligned}([c_1,c_2],m) >_{\mathbf{R}} 0 \;:=\; &(c_1 \geqslant_{\mathbf{Q}} 0 \wedge c_2 >_{\mathbf{Q}} 0) \vee (c_1 >_{\mathbf{Q}} 0 \wedge c_2 \geqslant_{\mathbf{Q}} 0) \vee \\ &(c_1 \geqslant_{\mathbf{Q}} 0 \wedge c_2 <_{\mathbf{Q}} 0 \wedge \varphi) \vee (c_1 \leqslant_{\mathbf{Q}} 0 \wedge c_2 >_{\mathbf{Q}} 0 \wedge \chi)\end{aligned}$$

with $\varphi = c_1 \times_{\mathbf{Q}} c_1 >_{\mathbf{Q}} c_2 \times_{\mathbf{Q}} c_2 \times_{\mathbf{Q}} m$ and $\chi = c_1 \times_{\mathbf{Q}} c_1 <_{\mathbf{Q}} c_2 \times_{\mathbf{Q}} c_2 \times_{\mathbf{Q}} m$.

4 Soundness

In this section we show that our encodings are sound, i.e., that from a satisfying assignment for the encoding, a model for the original constraint can be inferred. By $[\![\cdot]\!]$ we denote the semantic evaluation function of the encodings. Since the encodings for $=_{\mathbb{R}}$ and $>_{\mathbb{R}}$ in general are not exact but only approximations, such constraints may not appear at negative positions in Boolean formulas, e.g. $\mathsf{a} = \mathsf{b} \rightarrow \bot$ is satisfiable if $\mathsf{a} = ([\mathbf{2},\mathbf{0}],\mathbf{4})$ and $\mathsf{b} = ([\mathbf{0},\mathbf{1}],\mathbf{4})$ but a and b both represent 2. We remark that none of the benchmarks from Section 5 contains comparisons at negative positions.

Addition: We have

$$\begin{aligned}[\![(\mathbf{c}_n,m) +_{\mathbf{R}} (\mathbf{d}_n,m)]\!] &= \sum_{1\leqslant j\leqslant n} c_j \sqrt[n]{m^{j-1}} + \sum_{1\leqslant i\leqslant n} d_i \sqrt[n]{m^{i-1}} \\ &= \sum_{1\leqslant i\leqslant n} (c_i+d_i) \sqrt[n]{m^{i-1}} \\ &= [\![([c_1 +_{\mathbf{Q}} d_1,\ldots,c_n +_{\mathbf{Q}} d_n],m)]\!]\end{aligned}$$

Subtraction: Analogous to addition.

Multiplication: First we show that $[\![(\mathbf{c}_n,m) \gg i]\!] = \left(\sum_{1\leqslant j\leqslant n} c_j \sqrt[n]{m^{j-1}}\right) \sqrt[n]{m^i}$ by induction on i. In the base case

$$[\![(\mathbf{c}_n,m) \gg 0]\!] = [\![(\mathbf{c}_n,m)]\!] = \sum_{1\leqslant j\leqslant n} c_j \sqrt[n]{m^{j-1}} = \left(\sum_{1\leqslant j\leqslant n} c_j \sqrt[n]{m^{j-1}}\right) \sqrt[n]{m^0}$$

In the inductive step

$$\begin{aligned}
&[\![(\mathbf{c}_n,m) \gg (i+1)]\!] \\
&\quad = [\![([m \times_{\mathbf{Q}} c_n, c_1, \ldots, c_{n-1}], m) \gg i]\!] = \left(mc_n \sqrt[n]{m^0} + \sum_{2\leqslant j\leqslant n} c_{j-1} \sqrt[n]{m^{j-1}}\right) \sqrt[n]{m^i} \\
&\quad = \left(c_n \sqrt[n]{m^{n-1}} + \sum_{1\leqslant j\leqslant n-1} c_j \sqrt[n]{m^{j-1}}\right) \sqrt[n]{m^{i+1}} = \left(\sum_{1\leqslant j\leqslant n} c_j \sqrt[n]{m^{j-1}}\right) \sqrt[n]{m^{i+1}}
\end{aligned}$$

Using this result multiplication is then justified by

$$\begin{aligned}
&[\![(\mathbf{c}_n,m) \times_{\mathbf{R}} (\mathbf{d}_n,m)]\!] \\
&\quad = \left(\sum_{1\leqslant j\leqslant n} c_j \sqrt[n]{m^{j-1}}\right) \times \left(\sum_{1\leqslant i\leqslant n} d_i \sqrt[n]{m^{i-1}}\right) = \sum_{1\leqslant i\leqslant n} \left(\sum_{1\leqslant j\leqslant n} \left(c_j \sqrt[n]{m^{j-1}}\right)\right) \times d_i \sqrt[n]{m^{i-1}} \\
&\quad = \sum_{1\leqslant i\leqslant n} \left(\left(\sum_{1\leqslant j\leqslant n} c_j \sqrt[n]{m^{j-1}}\right) \times d_i\right) \times \sqrt[n]{m^{i-1}} = [\![\sum_{1\leqslant i\leqslant n} ((\mathbf{c}_n,m) \cdot d_i) \gg (i-1)]\!]
\end{aligned}$$

Equality: We have

$$\begin{aligned}
[\![(\mathbf{c}_n,m) =_{\mathbf{R}} (\mathbf{d}_n,m)]\!] &\iff \sum_{1\leqslant j\leqslant n} c_j \sqrt[n]{m^{j-1}} = \sum_{1\leqslant i\leqslant n} d_i \sqrt[n]{m^{i-1}} \\
&\iff \sum_{1\leqslant j\leqslant n} (c_j - d_j) \sqrt[n]{m^{j-1}} = 0 \\
&\Longleftarrow c_1 = d_1 \text{ and } \cdots \text{ and } c_n = d_n \\
&\iff [\![c_1 =_{\mathbf{Q}} d_1 \wedge \cdots \wedge c_n =_{\mathbf{Q}} d_n]\!]
\end{aligned}$$

Greater: First we give semantics to the $>^a_{\mathbf{R}}$ operator and show its soundness.

$$\begin{aligned}
[\![(\mathbf{c}_{k+1},m) >^a_{\mathbf{R}} (\mathbf{d}_{k+1},m)]\!] &\iff \sum_{1\leqslant j\leqslant k+1} c_j \sqrt[n]{m^{j-1}} + a \sqrt[n]{m^k} >_{\mathbb{R}} \sum_{1\leqslant i\leqslant k+1} d_i \sqrt[n]{m^{i-1}} \\
&\iff \sum_{1\leqslant j\leqslant k} c_j \sqrt[n]{m^{j-1}} + (a + c_{k+1} - d_{k+1}) \sqrt[n]{m^k} >_{\mathbb{R}} \sum_{1\leqslant i\leqslant k} d_i \sqrt[n]{m^{i-1}} \\
&\Longleftarrow \sum_{1\leqslant j\leqslant k} c_j \sqrt[n]{m^{j-1}} + (a + c_{k+1} - d_{k+1}) \sqrt[n]{m^{k-1}} >_{\mathbb{R}} \sum_{1\leqslant i\leqslant k} d_i \sqrt[n]{m^{i-1}} \\
&\qquad \text{and } a + c_{k+1} \geqslant d_{k+1} \\
&\iff [\![(\mathbf{c}_k,m) >^{a+c_{k+1}-d_{k+1}}_{\mathbf{R}} (\mathbf{d}_k,m)]\!]
\end{aligned}$$

Then soundness of $>_{\mathbf{R}}$, i.e., satisfiability of the encoding implies satisfiability of the constraint, follows.

Next we show soundness of the exact encoding (Definition 10), i.e., when $n = 2$. Note that $m > 0$ by assumption.

$$
\begin{aligned}
&[\![([c_1, c_2], m) >_{\mathbf{R}} 0]\!] \\
&\iff c_1 + c_2\sqrt{m} >_{\mathbb{R}} 0 \\
&\iff c_1 >_{\mathbb{R}} -c_2\sqrt{m} \\
&\iff c_1 \geqslant_{\mathbb{Q}} 0 \text{ and } -c_2 <_{\mathbb{Q}} 0 \text{ or } c_1 >_{\mathbb{Q}} 0 \text{ and } -c_2 \leqslant_{\mathbb{Q}} 0 \text{ or} \\
&\qquad c_1 \geqslant_{\mathbb{Q}} 0 \text{ and } -c_2 >_{\mathbb{Q}} 0 \text{ and } c_1^2 >_{\mathbb{Q}} mc_2^2 \text{ or } c_1 \leqslant_{\mathbb{Q}} 0 \text{ and } -c_2 <_{\mathbb{Q}} 0 \text{ and } c_1^2 <_{\mathbb{Q}} mc_2^2 \\
&\iff c_1 \geqslant_{\mathbb{Q}} 0 \text{ and } c_2 >_{\mathbb{Q}} 0 \text{ or } c_1 >_{\mathbb{Q}} 0 \text{ and } c_2 \geqslant_{\mathbb{Q}} 0 \text{ or} \\
&\qquad c_1 \geqslant_{\mathbb{Q}} 0 \text{ and } c_2 <_{\mathbb{Q}} 0 \text{ and } c_1^2 >_{\mathbb{Q}} mc_2^2 \text{ or } c_1 \leqslant_{\mathbb{Q}} 0 \text{ and } c_2 >_{\mathbb{Q}} 0 \text{ and } c_1^2 <_{\mathbb{Q}} mc_2^2 \\
&\iff [\![(c_1 \geqslant_{\mathbf{Q}} 0 \wedge c_2 >_{\mathbf{Q}} 0) \vee (c_1 >_{\mathbf{Q}} 0 \wedge c_2 \geqslant_{\mathbf{Q}} 0) \vee \\
&\qquad (c_1 \geqslant_{\mathbf{Q}} 0 \wedge c_2 <_{\mathbf{Q}} 0 \wedge \varphi) \vee (c_1 \leqslant_{\mathbf{Q}} 0 \wedge c_2 >_{\mathbf{Q}} 0 \wedge \chi)]\!]
\end{aligned}
$$

5 Assessment

We remark that our approach requires arithmetical variables to be based on (roots of) the same base. However, this does not immediately exclude solutions that can use different bases, as shown in the next example.

Example 11. Consider the constraint from Example 3 again. Our approach cannot find a model where $\alpha(\mathrm{a}_1) = \sqrt[3]{3}$ and $\alpha(\mathrm{a}_2) = \sqrt{2}$. However, it can determine the equivalent assignment $\alpha(\mathrm{a}_1) = \frac{1}{6}\sqrt[3]{648}$ and $\alpha(\mathrm{a}_2) = \frac{1}{18}\sqrt{648}$ since $648 = 2^3 \times 3^4$.

We implemented our approach for $n = 2$ in the SMT solver MiniSmt.[1] All tests have been performed on a single core of a server equipped with eight dual-core AMD Opteron® processors 885 running at a clock rate of 2.6 GHz and on 64 GB of main memory. We considered the non-linear benchmarks matrix 1 and matrix 2 stemming from termination analysis mentioned in [3], since we are not aware of any specific test-beds for non-linear real arithmetic. In Table 1 we compare the approximated encoding of $>_{\mathbb{R}}$ from [3] (lpar) against the exact encoding of Definition 10 (scss).[2] The numbers in parentheses indicate how many bits have been used to encode integers and intermediate results, respectively. In the table "yes" refers to the number of systems that could be shown satisfiable, "avg" gives the average time in seconds for finding a model and "to" indicates for how many systems the execution was aborted since no result was produced within 60 seconds.

It is not surprising that the lpar-approach is faster than the scss-approach since the constraints for the latter involve more multiplications, which are costly. On the contrary the scss-approach supports reason-

Table 1: Evaluation of two benchmark families from termination analysis (1391 constraints each)

	lpar(2,4)	scss(2,4)	lpar(3,4)	scss(3,4)	lpar(3,5)	scss(3,5)
	yes/ avg/ to	yes/ avg/ to	yes/ avg/ to	yes/ avg/ to	yes/ avg/ to	yes/ avg/ to
matrix 1	308/1.43/ 27	308/2.99/ 62	306/1.92/ 44	303/ 4.28/ 99	311/ 3.72/123	294/ 5.50/175
matrix 2	296/6.92/315	276/8.31/391	286/7.79/344	264/11.58/459	237/10.95/543	221/13.70/637

[1] `http://cl-informatik.uibk.ac.at/software/minismt`

[2] Full details available from `http://cl-informatik.uibk.ac.at/software/minismt/experiments/scss`.

ing about numbers that involve square roots different from $\sqrt{2}$. However, the test-beds we considered do not seem to require such numbers.

6 Conclusion

Recently in [3] an approach that reduces satisfiability of non-linear real arithmetic to non-linear rational/integer arithmetic has been presented. In [3] real variables may take values of the shape $c + \sqrt{2}d$ and $>_{\mathbb{R}}$ is only approximated. In this paper we generalized the approach to real algebraic numbers of the more general shape $\sum_{1 \leqslant i \leqslant n} c_i \sqrt[n]{m^{i-1}}$ and presented suitable encodings in non-linear integer/rational arithmetic. Furthermore for the setting of [3], where always $n = 2$, we formulated an exact encoding of $>_{\mathbb{R}}$.

Acknowledgments. We thank Friedrich Neurauter for helpful discussion.

References

[1] Collins, G.E.: Quantifier elimination for real closed fields by cylindrical algebraic decomposition. In: Proc. 2nd International Conference on Automata Theory and Formal Languages. LNCS, vol. 33, pp. 134–183 (1975)

[2] Tarski, A.: A Decision Method for Elementary Algebra and Geometry. 2nd edn. University of California Press, Berkeley (1957)

[3] Zankl, H., Middeldorp, A.: Satisfiability of non-linear (ir)rational arithmetic. In: LPAR-16. LNCS (LNAI). (2010). To appear. Available from `http://cl-informatik.uibk.ac.at/~hzankl/new/publications`.