Quantifier Elimination in Term Algebras
The Case of Finite Languages

Thomas Sturm[1] and Volker Weispfenning[2]

[1] University of Passau, Germany
sturm@uni-passau.de
http://www.fmi.uni-passau.de/~sturm/
[2] University of Passau, Germany
weispfen@uni-passau.de
http://www.fmi.uni-passau.de/~weispfen/

Abstract. We give a quantifier elimination procedure for term algebras over suitably expanded finite first-order languages. Our expansion is purely functional. Our method works by substituting finitely many parametric test terms. This allows us to obtain in addition sample solutions for an outermost existential quantifier block. The existence of our method implies that the considered quantifier elimination problem and as well the corresponding decision problem are in the fourth Grzegorcyk complexity class. For prenex input formulas with a bounded number of quantifiers our quantifier elimination procedure is elementary recursive. The same applies to arbitrary input formulas in case the language has only constants and unary function symbols. As a corollary we get corresponding upper bounds for the decision problem for term algebras.

1 Introduction

In 1962 Malcev published his famous fundamental result [Mal71] on the decidability of theories of free algebras and of algebras that are free relative to specified commutativity relations for each operation. For the special case of absolutely free algebras he had obtained a corresponding result already one year earlier by an apparently more complicated proof. His proof in [Mal71] introduces certain—fairly complicated—definable relations into the given language $\mathcal{L}$. His main theorem then provides an algorithmic reduction of arbitrary formulas to an and-or-combination of these relations. As a corollary he obtains a decision method for closed formulas. If one construes these definable relations as new atomic relations in an extended language $\mathcal{L}'$, then Malcev's main theorem establishes an algorithmic quantifier elimination procedure in $\mathcal{L}'$. A simplified and extended version of Malcev's approach appeared in [Mah88].

An alternative much more perspicuous quantifier elimination procedure for absolutely free algebras in a different extension language $\mathcal{L}''$ was found by Belegradek [Bel88]. It appears in slightly modified form in [Hod93]. Here the language $\mathcal{L}$ is extended by unary relation symbols R_f and unary function symbols $\mathrm{inv}_{f,i}$ for every n-ary function symbol f in $\mathcal{L}$ and all $i \in \{1, \dots, n\}$. These new symbols are interpreted in an absolutely free $\mathcal{L}$-algebra as follows: $R_f(a)$ holds for the elements a of the algebra of the form $f(b_1, \dots, b_n)$ with b_i arbitrary. If a is of this form, then $\mathrm{inv}_{f,i}(a) = b_i$; otherwise $\mathrm{inv}_{f,i}(a) = a$. Both this quantifier elimination procedure and the resulting decision procedure are rather obviously primitive recursive. No tighter upper complexity bound was known so far.

On the other hand the fundamental study of Compton and Henson [CH90] shows that decision procedures for absolutely free $\mathcal{L}$-algebras are not elementary recursive, provided $\mathcal{L}$ contains at least one function symbol of arity greater than one. This means that no decision algorithm for one of these algebras can run in time bounded by a finite tower of exponential functions. The class of elementary recursive functions is also known as the third class of the Grzegorcyk hierarchy [Gri99,Sch74]. This result implies that also no quantifier elimination procedure for any such algebra can be elementary recursive. This applies in particular to the procedures of Malcev, of Belegradek, and of Hodges.

More information on related efficient algorithms and complexity issues for related partial and extended problems is contained in the papers [RV01,DV00,VV98].

In this paper we revisit the quantifier elimination problem and the decision problem for absolutely free algebras from a new viewpoint. We assume that the given functional language $\mathcal{L}$ is finite and has at least one constant and at least one function symbol of positive arity. We study the $\mathcal{L}$-algebra $\mathbf{T}$ of variable-free $\mathcal{L}$-terms. In a first step we show that except for some very simple languages $\mathcal{L}$ the $\mathcal{L}$-structure $\mathbf{T}$ never admits quantifier elimination. So we consider instead a purely functional extension language $\mathcal{L}^*$ of $\mathcal{L}$ that is a sublanguage of the extension $\mathcal{L}''$ considered by Belegradek and Hodges. The $\mathcal{L}$-algebra $\mathbf{T}$ expands in a natural way to an $\mathcal{L}^*$-algebra $\mathbf{T}^*$ with universe T.

Our approach will show that in $\mathcal{L}^*$ the algebra $\mathbf{T}^*$ admits a very special type of quantifier elimination, viz. quantifier elimination by finitely many test terms. This quantifier elimination method is technically a quantifier elimination by substitution of Skolem terms in the sense introduced and studied in [Wei88,LW93]. By virtue of this fact we obtain new and detailed upper complexity bounds for the quantifier elimination problem and the decision problem in $\mathbf{T}^*$ wrt. the number of alternating quantifiers in the input formula.

Roughly speaking quantifier elimination for a prenex input formula φ with a alternating blocks of quantifiers each of at most k quantifiers can be performed in time and space bounded by an $a \cdot (k+2)$-fold iteration of the exponential function. As a consequence we get similar upper bounds for the decision problem of closed formulas φ. In particular the decision problem is in the fourth class of the Grzegorcyk hierarchy; so this upper bound matches closely the lower bound provided by Compton and Henson [CH90]. For input formulas with a bounded number of quantifiers both problems are elementary recursive, i.e. in the third class of the Grzegorcyk hierarchy.

For the special case where the language $\mathcal{L}$ has only constants and unary function symbols we obtain a doubly exponential and hence elementary recursive upper bound for both the quantifier elimination problem and the decision problem in $\mathbf{T}^*$.

An additional practical benefit of quantifier elimination by test terms is the fact that it yields *extended quantifier elimination* with answers for an outermost existential quantifier block, in particular for existential formulas. This means that we get a finite selection of sample solutions for all the variables in the outermost existential quantifier block of the input formula. In particular we get a quadruply exponential upper complexity bound for extended quantifier elimination in positive existential formulas.

For the special case where the language $\mathcal{L}$ contains k constants and n unary function symbols for $n > 1$ the term structure $\mathbf{T}$ can be viewed as the complete unlabelled tree of outdegree n with k roots, and all our results can be interpreted as results about this tree. For $k = 1$ and $n = 2$ the theory of $\mathbf{T}$ is thus the first-order fragment of the monadic second order theory, whose decidability was proved in a famous result of Rabin [Rab77]. In contrast to this theory that is known to be non-elementary recursive, our results show that the quantifier elimination problem and the decision problem for the first-order part is indeed elementary recursive, in fact doubly exponential.

From a practical point of view, our new test term approach has a crucial advantage: Quantifier elimination by test terms has been generically implemented within the comprehensive REDUCE-based computer logic system REDLOG [DS97]. It can thus be expected that a stable and efficient implementation will be available soon.

Recently, there is considerable research on integrating quantifier elimination techniques with constraint logic programming, where quantifier elimination takes the part of the constraint solver. One such system is CLP(RL) [Stu02], which is based on REDLOG mentioned above. Having quantifier elimination in the term algebra available will allow to replace also unification in the logic programming part by quantifier elimination. This quantifier elimination can in turn be considered another particular constraint solver, which leads to a new completely uniform view on constraint logic programming. In addition, we obtain a theoretical foundation for a notion of parametric constraint logic programming.

The authors gratefully acknowledge the detailed expert advice of Andrei Voronkov on published results concerning free term algebras that were obtained after Malcev's fundamental papers.

2 A Negative and a Positive Result

We start with some notational conventions. Let $\mathcal{L}$ be a finite first order language without relation symbols and with at least one constant symbol c_0. We denote the set of variable-free $\mathcal{L}$-terms by T and the canonical $\mathcal{L}$-structure with universe T by $\mathbf{T}$. We refer to $\mathbf{T}$ as the *term algebra* of $\mathcal{L}$.

Given an $\mathcal{L}$-term t for a language $\mathcal{L}$, we denote the length of the word t by $|t|$, and we denote the depth of t considered as a tree by $\Delta(t)$. We agree that $\Delta(c) = 0$ for constants c. These definitions also apply to elements of T. For $\mathcal{L}$-formulas φ, we denote by $|\varphi|$, the length of the word φ, and by $\Delta(\varphi)$ the maximal depth among all terms occurring in φ. For sets S of terms or formulas, $\Delta(S)$ denotes the maximal depth among all terms or formulas in S, respectively.

For any set M, we denote by $\#M$ the cardinality of M. In particular, for finite sets S of formulas or terms $\#S$ is the number of contained formulas or terms. Similarly, we denote by $\#\mathcal{L}$ the cardinality of a language $\mathcal{L}$. This is the cardinality of the set of constants and function symbols in $\mathcal{L}$.

In this section we show that quantifier elimination is impossible for $\mathbf{T}$ regarded as $\mathcal{L}$-structure, unless $\mathcal{L}$ contains exactly one unary function symbol. The clue to this impossibility result is the following structural analysis of subsets of T that are definable by a quantifier-free $\mathcal{L}$-formula. We call a subset M of T *definable* in $\mathbf{T}$ if there is an $\mathcal{L}$-formula $\varphi(x)$ with at most one free variable x such that $M = \{\, a \in T \mid \mathbf{T} \models \varphi(a) \,\}$.

Theorem 1. *Let $S \subseteq T$. Then S is definable in $\mathbf{T}$ by a quantifier-free $\mathcal{L}$-formula iff S is finite or cofinite.*

Proof. If $S = \{a_1, \dots, a_n\}$ with $a_i \in T$, then S is defined by the quantifier-free $\mathcal{L}$-formula

$$\bigvee_{i=1}^{n} x = a_i,$$

and $T \setminus S$ is defined by the negation of this formula.

For the converse we show that any subset S of T defined by a single $\mathcal{L}$-equation is empty or a one-element set or all of T. This fact implies that any set defined by a Boolean combination of $\mathcal{L}$-equations is finite or cofinite. We show our claim for $\mathcal{L}$-equations $s = t$ with at most one variable x by induction on the sum $|s| + |t|$ of the lengths of the $\mathcal{L}$-terms s and t. If $|s| + |t| = 2$, then the equation is of one of the forms $c = d$ or $x = c$ or $c = x$ or $x = x$ for $\mathcal{L}$-constants c, d, and the claim is obvious. Otherwise we may assume that $|s| > 1$, and hence that s is of the form $f(s_1, \dots, s_n)$ for some $n > 0$ and some n-ary function symbol f in $\mathcal{L}$. If t is not of the form $f(t_1, \dots, t_n)$, then the equation defines the empty set. If t is of the form $f(t_1, \dots, t_n)$, then the equation is equivalent in $\mathbf{T}$ to the conjunction

$$\bigwedge_{i=1}^{n} s_i = t_i.$$

The claim follows now from the induction hypothesis applied to the equations in this conjunction. □

Corollary 2. *Suppose $\mathcal{L}$ contains at least one function symbol of arity at least 2 or at least two different unary function symbols. Then quantifier elimination is impossible in $\mathbf{T}$.*

Proof. If $\mathcal{L}$ contains at least one function symbol f of arity at least 2, then pick an $\mathcal{L}$-constant c, and consider the formula

$$\exists y\big(x = f(y, c, \dots, c)\big).$$

The set defined by this formula in $\mathbf{T}$ is infinite and coinfinite in T. Hence this formula cannot be equivalent to a quantifier-free $\mathcal{L}$-formula in $\mathbf{T}$.

If $\mathcal{L}$ contains at least two different unary function symbols f, g, then we consider the formula

$$\exists y\big(x = f(y)\big)$$

and arrive at the same conclusion. □

Corollary 2 leaves open the case that $\mathcal{L}$ contains exactly one unary function symbol. If there is only one constant, then $\mathbf{T}$ is isomorphic to the structure $\mathbb{N}$ of natural numbers with constant 0 and the successor function s. If there is more than one constant, then $\mathbf{T}$ can be considered as finitely many independent copies of $(\mathbb{N}, 0, s)$. The following theorem establishes a quantifier elimination procedure in $\mathcal{L}$ for these cases:

Theorem 3. *Let $\mathcal{L} = (c_1, \dots, c_n, s)$ be a language with constant symbols c_1, …, c_n and one unary function symbol s. Then the corresponding term structure $\mathbf{T}$ admits quantifier elimination in $\mathcal{L}$.*

Proof. To begin with, denote for $k \in \mathbb{N}$ by s^k the k-fold iteration of the function symbol s. Then we have for k, $m \in \mathbb{N}$ with $k \neq m$ that

$$\mathbf{T} \models s^k(x) = s^k(x) \longleftrightarrow \text{true} \quad \text{and} \quad \mathbf{T} \models s^m(x) = s^k(x) \longleftrightarrow \text{false}\,.$$

So it suffices to eliminate the quantifier in formulas $\exists x \varphi(x, y_1, \dots, y_m)$ where φ is of the form

$$\bigwedge_{i \in I} s^i(x) = t_i(y_1, \dots, y_m) \wedge \bigwedge_{j \in J} \neg s^j(x) = t'_j(y_1, \dots, y_m) \quad \text{with} \quad I, J \subseteq \mathbb{N}_0.$$

Let $\mu = \max I \cup J$. Then our formula φ is equivalent to the following formula φ':

$$\bigwedge_{i \in I} s^{\mu}(x) = s^{\mu-i}(t_i) \wedge \bigwedge_{j \in J} \neg s^{\mu}(x) = s^{\mu-j}(t'_j).$$

Defining

$$\psi_{\mu}(z) = \bigwedge_{k=1}^{n} \bigwedge_{i=0}^{\mu-1} \neg z = s^i(c_k),$$

we have $\mathbf{T} \models \exists x\big(s^{\mu}(x) = z\big) \longleftrightarrow \psi_{\mu}(z)$. It follows that in case $I \neq \varnothing$, say $i_0 \in I$, the original formula $\exists x \varphi$ is equivalent in $\mathbf{T}$ to

$$\varphi\big(s^{\mu-i_0}(t_{i_0}), y_1, \dots, y_m\big) \wedge \psi_{\mu}\big(s^{\mu-i_0}(t_{i_0})\big).$$

In case $I = \varnothing$, we find that $\exists x \varphi$ is simply equivalent to "true." In fact, for arbitrary values of $y_1, \dots, y_m$ in T the value $x = s^k(c_1) \in T$ will satisfy $\varphi(x, y_1, \dots, y_m)$ for some sufficiently high value $k \in \mathbb{N}$ depending on the particular values of $y_1, \dots, y_m$. □

Note that the above theorem does not establish an extended quantifier elimination procedure. Such a procedure will be provided later in some extended language $\mathcal{L}^*$.

3 Inverse Functions and Normal Forms

Our aim is to find a suitable extension language of $\mathcal{L}$ and a corresponding expansion of the term algebra $\mathbf{T}$ by definable functions such that quantifier elimination becomes generally possible.

For this purpose we extend the given language by certain new function symbols. These new symbols represent componentwise inverses to the given functions on the term algebra $\mathbf{T}$. Recall that in $\mathbf{T}$ an element $f^{\mathbf{T}}(a_1, \dots, a_n) \in T$ with $a_1, \dots, a_n \in T$ determines its immediate subterms $a_1, \dots, a_n$ uniquely. Accordingly, we define for every n-ary function symbol f in $\mathcal{L}$ and every integer $i \in \{1, \dots, n\}$ a function $\mathrm{inv}_{f,i} : T \to T$ as follows:

$$\mathrm{inv}_{f,i}(a) = \begin{cases} a_i & \text{if } a = f(a_1, \dots, a_n), \\ a & \text{else} \end{cases}$$

The function $\mathrm{inv}_{f,i}$ thus delivers the i-th argument term for terms beginning with f, and for all others returns, as sort of a default value, the complete term itself.

We let $\mathcal{L}^*$ be the extension language of $\mathcal{L}$ obtained by adding unary function symbols $\mathrm{inv}_{f,i}$ corresponding to these *inverse* functions for all $n \in \mathbb{N}$ with $n > 0$, all n-ary function symbols f in $\mathcal{L}$, and every $i \in \{1, \dots, n\}$. It is important to understand that this is an extension of our first-order language $\mathcal{L}$ to $\mathcal{L}^*$ together with a corresponding $\mathcal{L}^*$-expansion of the term algebra $\mathbf{T}$ to $\mathbf{T}^*$. The universe of $\mathbf{T}^*$, however, is still T. Due to the non-trivial interpretation of the inverse function symbols, $\mathbf{T}^*$ is strictly speaking not longer a term algebra. We shall speak of an *expanded term algebra*.

Equivalence of $\mathcal{L}^*$-terms is defined wrt. the structure $\mathbf{T}^*$: Two $\mathcal{L}^*$-terms s and t are $\mathbf{T}^*$*-equivalent* if $\mathbf{T}^* \models s = t$. Note here that s and t possibly contain variables, and recall that by definition the equality in $\mathbf{T}^*$ must hold for all interpretations of these variables. In other words, s and t are $\mathbf{T}^*$-equivalent if and only if they describe the same functions in $\mathbf{T}^*$.

We call any nested superposition of inverse function symbols in front of a variable an *inverse term* of $\mathcal{L}^*$. This notion includes empty superpositions, i.e. variables themselves. Based on this notion, we can exhibit a normal form for $\mathcal{L}^*$-terms wrt. to $\mathbf{T}^*$-equivalence: An $\mathcal{L}^*$-term u is in *normal form*, if it is of the form $l[v_1/x_1, \dots, v_n/x_n]$ where l is an $\mathcal{L}$-term, and $v_1, \dots, v_n$ are inverse terms. In other words, u is the result of substituting certain inverse terms $v_1, \dots, v_n$ for certain variables $x_1, \dots, x_n$ in an $\mathcal{L}$-term l.

Lemma 4 (Normal form of $\mathcal{L}^*$-terms). *Every $\mathcal{L}^*$-term t is $\mathbf{T}^*$-equivalent to an $\mathcal{L}^*$-term u in normal form. This normal form u of t can be obtained from t algorithmically in polynomial time and linear space including the size of the output, and we have $\Delta(u) \leqslant \Delta(t)$.*

Proof. We prove the assertion by induction on the length of t. If $|t| = 1$, then t is either a variable or an $\mathcal{L}$-constant. In either case, we put $u = t$ with no substitution performed.

Consider now $|t| > 1$. If t is of the form $f(s_1, \dots, s_n)$, where f is an n-ary $\mathcal{L}$-function symbol, then we can by the induction hypothesis compute normal forms $s'_1, \dots, s'_n$ for the $s_1, \dots, s_n$, and obtain $u = f(s'_1, \dots, s'_n)$ as a normal form for t. If t is of the form $\mathrm{inv}_{f,i}(s)$ for an n-ary $\mathcal{L}$-function symbol f, an integer $i \in \{1, \dots, n\}$, and an $\mathcal{L}^*$-term s, then we distinguish three subcases: First, if s is a variable or an inverse term, then we can simply put $u = t$. Second, if s is an $\mathcal{L}$-constant or of the form $g(s_1, \dots, s_m)$ for some m-ary $\mathcal{L}$-function symbol g different from f and $\mathcal{L}^*$-terms $s_1, \dots, s_m$, then we put $u = s$. In the remaining third case, s is of the form $f(s_1, \dots, s_n)$ with $\mathcal{L}^*$-terms $s_1, \dots, s_n$. Then t is $\mathbf{T}^*$-equivalent to s_i, and so we put $u = s'_i$, a normal form of s_i, which can be computed by the induction assumption. $\square$

Next we apply our above normal form lemma to find normal forms for equations between $\mathcal{L}^*$-terms. For this purpose, equivalence between $\mathcal{L}^*$-formulas is defined in terms of logical equivalence in the structure $\mathbf{T}^*$: Two $\mathcal{L}^*$-formulas φ and ψ are $\mathbf{T}^*$*-equivalent* if and only if $\mathbf{T}^* \models \varphi \longleftrightarrow \psi$. The normal forms provided by the following lemma are actually just an intermediate step towards refined normal forms provided by the lemma after.

Lemma 5 (Normal form of $\mathcal{L}^*$-equations). *Every equation φ between $\mathcal{L}^*$-terms is equivalent to* true *or* false *or to a conjunction $\tilde{\varphi}$ of equations $w = u$, where w is either constant or an inverse term, and u is a term in normal form. We generally have $\Delta(\tilde{\varphi}) \leqslant \Delta(\varphi)$. If there are only unary function symbols in $\mathcal{L}$, then the conjunction degenerates to a single equation, and, consequently, the number of atomic formulas does not increase. If there are function symbols of arity greater than one, then the number of atomic formulas in $\tilde{\varphi}$ is bounded by $2^{O(\Delta(\varphi))}$.*

Proof. By Lemma 4 we may assume that the given equation φ is of the form $u = u'$ with $\mathcal{L}^*$-terms u and u' in normal form. We proceed by induction on $\min\{|u|, |u'|\}$. If w.l.o.g $|u| = 1$, then u is either a constant or a variable. We thus put $\tilde{\varphi}$ to be φ.

Suppose now that $|u| > 1$ and $|u'| > 1$. If u is an inverse term, we again put $\tilde{\varphi}$ to be φ; similarly if u' is an inverse term, we put $\tilde{\varphi}$ to be $u' = u$. If both u and u' are not inverse terms, then both begin with an $\mathcal{L}$-function symbol. If the corresponding function symbols are not identical, then we put $\tilde{\varphi}$ to be "false." Otherwise, we have $u = f(s_1, \dots, s_n)$ and $u' = f(t_1, \dots, t_n)$ where f is some n-ary $\mathcal{L}$-function symbol, and $s_1, \dots, s_n, t_1, \dots, t_n$ are $\mathcal{L}^*$-terms. In this case φ is equivalent to the conjunction

$$\bigwedge_{i=1}^{n} s_i = t_i,$$

and we can apply our induction hypothesis to each of these equations. $\square$

Lemma 6 (Refined normal form of $\mathcal{L}^*$-equations). *Every equation φ between $\mathcal{L}^*$-terms is equivalent to* true *or* false *or to a conjunction $\tilde{\varphi}$ of equations and negated equations. The equations in $\tilde{\varphi}$ are of the form $w = v$, where w is either constant or an inverse term, and v is an inverse term. The negated equations in $\tilde{\varphi}$ are of the more restricted form $\neg \operatorname{inv}_{f,1}(v) = v$, where v is an inverse term. We generally have $\Delta(\tilde{\varphi}) \leqslant \Delta(\varphi)$. If there are only unary function symbols in $\mathcal{L}$, then the number of atomic formulas in the conjunction is bounded by $O(\Delta(\varphi))$. If there are function symbols of arity greater than one, then the number of atomic formulas in $\tilde{\varphi}$ is bounded by $2^{O(\Delta(\varphi))}$.*

Proof. By Lemma 5, it suffices to consider equations of the form $w = u$, where w is either constant or an inverse term, and u is a term in normal form. Let w be a constant, and let u be not an inverse term. Then we can by inspection decide $w = u$ to be either "true" or "false." If, in contrast, u is an inverse term, then we are already done.

It remains to treat the case that w is an inverse term. We transform the equation $w = u$ into a conjunction of equations of the required form. This is done by induction on the length of u. If $|u| = 1$, then u is either an $\mathcal{L}$-constant or a variable. In the former case, the equation $u = w$ is of the required form. In the latter case, u is an inverse term, and we are already done. Let now $|u| > 1$. If u is an inverse term, then we are done. Else u is of the form $f(t_1, \dots, t_n)$ where f is an n-ary $\mathcal{L}$-function symbol, and $t_1, \dots, t_n$ are $\mathcal{L}^*$-terms. Then the equation $w = f(t_1, \dots, t_n)$, which we are considering, is equivalent to

$$\neg \operatorname{inv}_{f,1}(w) = w \wedge \bigwedge_{i=1}^{n} \operatorname{inv}_{f,i}(w) = t_i.$$

We can now apply the induction hypothesis to each of these equations. Notice that the first equation $\operatorname{inv}_{f,1}(w) = w$ is already in the desired form, such that we will not possibly obtain a conjunction within the scope of the negation. $\square$

We give an intuitive idea of the content of Lemma 6. If we regard the elements a of T in a natural way as labelled trees, then the composite function of $\mathbf{T}^*$ given by some inverse term assigns to each $a \in T$ a certain subtree of a specified by the inverse term. In this view Lemma 6 reduces equations $s = t$ of $\mathcal{L}^*$-terms to a set of equations, each of which restricts the possible values of the variables in s and t. Each inverse term describes a subtree of the one contained variable. There are three possible types of restrictions:

(a) A negated equation $\neg\,\mathrm{inv}_{f,1}(v) = v$ states that the subtree described by the inverse term v does not start with the function symbol f.
(b) An equation $w = v$ where w is a constant states that this constant w occurs at the position described by the inverse term v thus making up a trivial subtree.
(c) An equation $w = v$, where w is an inverse term states equality of the subtrees described by w and v. Here w and v can describe subtrees of either the same variable or of different ones.

4 The Elimination Procedure

Recall that we consider only languages with at least one constant symbol, because otherwise we would obtain an empty universe T. We are going to distinguish three different *types* of finite languages:

Type U-1 A language is of type U-1 if it contains a finite nonzero number of constant symbols, exactly one unary function symbol, and no function symbols of arity greater than 1.
Type U-N A language is of type U-N if it contains a finite nonzero number of constant symbols, at least two but finitely many unary function symbols, and no function symbols of arity greater than 1.
Type N-N A language is of type N-N if it contains a finite nonzero number of constant symbols and an arbitrary finite number of function symbols including at least one function symbols of arity greater than 1.

Depending on the type of a language $\mathcal{L}$ there are alternative ways to obtain large sets of different terms of restricted depth:

Lemma 7. *(i) Let $\mathcal{L}$ be of type U-1, let c be an $\mathcal{L}$-constant, and let f be a unary $\mathcal{L}$-function symbol. For each $n \in \mathbb{N}$, we define the following set of $\mathcal{L}$-terms:*

$$G_n = \{c, f(c), f(f(c)), \ldots, f^n(c)\} \quad \textit{for} \quad n \in \mathbb{N}.$$

For all $n \in \mathbb{N}$, we have $\#G_n = n + 1$ and $\Delta(G_n) = n$.
(ii) Let $\mathcal{L}$ be of type U-N, let c be an $\mathcal{L}$-constant, and let f and g be a unary $\mathcal{L}$-function symbols. We recursively define the following sequence of sets of $\mathcal{L}$-terms:

$$G_0 = \{c\}, \qquad G_n = \{\, f(s) \mid s \in G_{n-1} \,\} \cup \{\, g(s) \mid s \in G_{n-1} \,\} \quad \textit{for} \quad n > 0.$$

Then for all $n \in \mathbb{N}$, we have that $\#G_n = 2^n$ and $\Delta(G_n) = n$.
(iii) Let $\mathcal{L}$ be of type N-N, let c be an $\mathcal{L}$-constant, and let g be an $\mathcal{L}$-function symbol of arity $k \geqslant 2$. We recursively define the following sequence of sets of $\mathcal{L}$-terms:

$$G_0 = \{c\}, \quad G_n = G_{n-1} \cup \{\, g(s, s', t, \ldots, t) \mid s, s' \in G_{n-1} \,\} \quad \textit{for} \quad n > 0.$$

Then for all $n \in \mathbb{N}$, we have that $\#G_n \geqslant 2^n$ and $\Delta(G_n) = n$.

Proof. (i) This is obvious. Notice that by convention $f^0(c) = t$.
(ii) By induction on n. Clearly, $\#G_0 = 1 = 2^0$ and $\Delta(G_0) = \Delta(c) = 0$. Consider now $n > 0$. Then by the induction hypothesis, $\#G_{n-1} = 2^{n-1}$ and $\Delta(s) = n - 1$ for all $s \in G_{n-1}$. Since the union in the recursive definition of G_n is obviously a disjoint one, it follows that

$$\#G_n = 2\#G_{n-1} = 2 \cdot 2^{n-1} = 2^n,$$

and for all $s \in G_n$, we have $\Delta(s) = n - 1 + 1 = n$.
(iii) By induction on n. The case $n = 0$ is as in (ii). Consider now $n > 0$. Then by the induction hypothesis, $\#G_{n-1} \geqslant 2^{n-1}$ and there is $s \in G_{n-1}$ of maximal depth $n - 1$. From this we obtain

$$\#G_n = \#G_{n-1} + (\#G_{n-1})^2 \geqslant 2^{n-1} + 2^{n-1} \cdot 2^{n-1} = 2^{n-1}(1 + 2^{n-1}) \geqslant 2^n.$$

Furthermore, there is $g(s, s, t, \ldots, t) \in G_n$ of maximal depth n. □

We now focus our attention to the situation, when one wants to eliminate a quantifier $\exists x(\varphi)$, where φ is a quantifier-free formula containing besides x possibly other free variables $y_1, \ldots, y_n$.

We may assume that φ is a conjunction of equations and negated equations in normal form according to Lemma 6. These equations may be each of one of the forms

(a) $v(x) = c$,
(b) $v(x) = v'(x)$,
(c) $v(x) = v'(y_i)$,

where all terms v, v' are inverse terms. All these conditions of the form (a)–(c) specify, or restrict the subterms of the unknown element x at certain specified positions. Other equations in φ like $v(y_i) = v'(y_i)$ and $v(y_i) = c$ with inverse terms v, v' do not involve the quantified variable x.

We may w.l.o.g. suppose that φ is an and-or-combination of equations and negated equations of the form (a)–(c). After a preparing technical Lemma, we will give out elimination theorem, which is the clue for efficient quantifier elimination in $\mathbf{T}^*$. It simulates the truth values for an element x satisfying φ of all equations in φ. For this, we can neglect the Boolean structure of φ, and from now on consider instead the set Φ of equations occurring in φ.

Notice that all function symbols in Φ are either inverse function symbols or $\mathcal{L}$-constants. Thus the maximal depth $\Delta(\Phi)$ of terms in Φ is the maximal nesting of inverse function symbols. We will also consider more precisely $\Delta_x(\Phi)$ and $\Delta_y(\Phi)$, which are the maximal nestings in Φ of inverse function symbols involving the variable x and not involving the variable x, respectively. Obviously, we have that $\Delta(\Phi) = \max\{\Delta_x(\Phi), \Delta_y(\Phi)\}$.

Our basic idea is that equations and inequalities between inverse terms impose identities and non-identities between the subterms that they access in their arguments. Intuitively, a single inverse term cannot look deeper into its argument than its depth, and even taking transitivity into account there will still be a bound in terms of $\Delta(\Phi)$ and $\#\Phi$. Thus details of satisfying terms that lie deeper than this bound are not relevant, and it suffices to try candidate terms in the variables $y_1, \ldots, y_m$ of restricted depth for simulating consistent truth value distributions on the atomic formulas in Φ. We call such terms Φ-restricted terms.

The exact definition of a *Φ-restricted term* depends on the type of the language $\mathcal{L}$: For languages of type U-1, we recall the notion of normal forms of $\mathcal{L}^*$-terms as defined for Lemma 4, and define the set of Φ-restricted terms as

$$R_\Phi = \{\, \mathcal{L}^*\text{-term } t(y_1, \ldots, y_m) \mid t \text{ in normal form and } \Delta(t) \leqslant 2\,\Delta(\Phi) + \#\Phi \,\}.$$

For languages of type U-N, the set of Φ-restricted terms is given by

$$R_\Phi = \{\, \mathcal{L}^*\text{-term } t(y_1, \ldots, y_m) \mid \Delta(t) \leqslant 2\,\Delta(\Phi) + \log(\#\Phi + 1) \,\}.$$

For languages of type N-N let A be the maximal arity of all function symbols in $\mathcal{L}$. Then the set of Φ-restricted terms is given by

$$R_\Phi = \{\, \mathcal{L}^*\text{-term } t(y_1, \ldots, y_m) \mid \Delta(t) \leqslant \Delta(\Phi) \cdot (\#\Phi + 1) + \log(\#\Phi + 1) \,\}.$$

Asymptotically, we obtain in this case $\Delta(R_\Phi) = O(\Delta(\Phi) \cdot \#\Phi)$.

Combining our information on upper bounds for $\Delta(R_\Phi)$ with the maximal arity of function symbols in $\mathcal{L}$, we obtain bounds for the maximal length $|R_\Phi|$ of terms in R_Φ: For type U-1:

$$|R_\Phi| \leqslant 2\,\Delta(\Phi) + \#\Phi = O(\Delta(\Phi) + \#\Phi),$$

for type U-N:

$$|R_\Phi| \leqslant 2\,\Delta(\Phi) + \log(\#\Phi + 1) = O(\Delta(\Phi) + \log \#\Phi),$$

and for type N-N:

$$|R_\Phi| \leqslant A^{\Delta(\Phi)\cdot(\#\Phi+1)+\log(\#\Phi+1)+1} = 2^{O(\Delta(\Phi)\cdot\#\Phi)}.$$

In either case, it is obvious that R_Φ has the crucial property to be finite. In the case of U-1-languages, we have normal forms $t = f^{\Delta\, t-k}\,\mathrm{inv}_{f,1}^k\, z$, where $k \in \{0, \ldots, \Delta(t)\}$, f is the unique unary $\mathcal{L}$-function symbol, and z is either an $\mathcal{L}$-constant symbol or one of the variables $y_1, \ldots, y_m$. Thus

$$\#R_\Phi \leqslant (m + \#L) \cdot (2\Delta(\Phi) + \#\Phi) \cdot (2\Delta(\Phi) + \#\Phi) = O(m \cdot (\Delta(\Phi) + \#\Phi)^2).$$

For U-N-languages:

$$\#R_\Phi \leqslant (2\#\mathcal{L}+m)^{2\,\Delta(\Phi)+\log(\#\Phi+1)} = 2^{O(\log m\cdot(\Delta(\Phi)+\log\#\Phi))},$$

and for N-N-languages:

$$\#R_\Phi \leqslant ((A+1)\#\mathcal{L}+m)^{A^{\Delta(\Phi)\cdot(\#\Phi+1)+\log(\#\Phi+1)+1}} = 2^{\log m\cdot 2^{O(\Delta(\Phi)\cdot\#\Phi)}}.$$

The latter two bounds follow from the fact that the respective exponents are bounds for the number of nodes in the tree associated with a Φ-restricted term and $\#\mathcal{L}+m$ is a bound on the number of symbols that may occur at each node. These symbols include the variables $y_1, \dots, y_m$, the constant and function symbols from $\mathcal{L}$, and all corresponding inverse function symbols from $\mathcal{L}^*$.

For establishing the elimination theorem, we need some further definitions. Each term $t \in T$ can be considered a tree. If t is a variable or a constant then this tree degenerates to one single node. Otherwise, t is of the form $f(t'_1,\dots,t'_n)$ for some n-ary function symbol f and terms $t'_1, \dots, t'_n \in T$. Then the root of the tree is f, and the successors are the trees recursively obtained from $t'_1, \dots, t'_n$. From this point of view we can identify *positions* of symbols in the term t by describing the path to the corresponding symbol by a finite sequence $\omega \in \mathbb{N}^*$ of natural numbers:

(i) The position of the first symbol of the term, i.e. the root of the tree, is the empty sequence $()$.
(ii) Let α be a symbol in t, which is not the first one. Then α is the first symbol of a proper subterm of t. There is an n-ary function symbol f such that t is the i-th argument of f for some $i \in \{1,\dots,n\}$. Let ω be the position of this f in t. Then we define the position of α to be the concatenation $w \circ i$.

We denote by $\Omega(t) \subseteq \mathbb{N}^*$ the finite set of all existing positions in t. For $\omega \in \Omega(t)$ we denote by $t\omega \in T$ the subterm of t starting at position ω. The length of a sequence $\omega \in \Omega(t)$ is denoted by $|\omega|$. It follows that $\Delta(t\omega) = \Delta(t) - |\omega|$.

There is a natural partial order on the set of positions: If for two given positions ω, $\omega' \in \mathbb{N}^*$ we have that ω is an initial segment of ω', then we write $\omega \preccurlyeq \omega'$. If $\omega \preccurlyeq \omega'$ or, vice versa, $\omega' \preccurlyeq \omega$, then we say that ω and ω' are *comparable*; else they are *uncomparable*.

Let v be an inverse term, and let $t \in T$. Then v uniquely induces a position $\mathrm{pos}(v,t) \in \Omega(t)$ as follows:

(i) If v is a variable, then $v^{\mathbf{T}^*}(t) = t$, and we define $\mathrm{pos}(v,t) = ()$.
(ii) Let v be of the form $\mathrm{inv}_{f,i}(v')$, where f is an n-ary function symbol, $i \in \{1,\dots,n\}$, and v' is an inverse term, and let $\mathrm{pos}(v',t) = \omega$. If $v'^{\mathbf{T}^*}(t)$ is of the form $f(t'_1,\dots,t'_n)$, then $\mathrm{pos}(v,t) = \omega \circ i$, else $\mathrm{pos}(v,t) = \omega$.

It follows that $t\,\mathrm{pos}(v,t) = v^{\mathbf{T}^*}(t)$. On the other hand, t might contain further copies of $v^{\mathbf{T}^*}(t)$ at other positions.

Our last definition enables us to transform a term $t \in T$ of our term structure into a $\mathcal{L}^*$-term by replacing certain occurrences of some subterm by some $\mathcal{L}^*$-term: Let $t \in T$, let t' be a subterm of t, let s be an $\mathcal{L}^*$-term, and let $M \subseteq \mathbb{N}^*$. Then the $\mathcal{L}^*$-term $t\langle s,t',M\rangle$ is obtained by considering all $\omega \in M$ for which $t\omega = t'$ and replacing at each such position ω the subterm t' by s. The following is a formal recursive definition of this:

(i) If t is an $\mathcal{L}$-constant, then there are two possible cases: if $t = t'$ and $() \in M$ then $t\langle s,t',M\rangle = s$; else $t\langle s,t',M\rangle = t$.
(ii) Consider t of the form $f(t_1,\dots,t_n)$ for an n-ary $\mathcal{L}$-function symbol f and $t_1, \dots, t_n \in T$. If $t = t'$ then we proceed as in the case of a constant: If $() \in M$ then $t\langle s,t',M\rangle = s$; else $t\langle s,t',M\rangle = t$. If, in contrast, $t \neq t'$, then

$$t\langle s,t',M\rangle = f\big(t_1\langle s,t',M_1\rangle,\dots,t_n\langle s,t',M_n\rangle\big),$$

where $M_i = \{\,\omega \in \mathbb{N}^* \mid i \circ \omega \in M\,\}$ for $i \in \{1,\dots,n\}$.

Let x be a variable, let Φ be a finite set of equations of the form $v(x) = v'(x)$ or $u(x) = u'(y'_i)$, where v, v', u are inverse terms, and u' is either an inverse term or a constant. Let $d_1, \dots, d_m$, $t \in T$ such that $\mathbf{T}^* \models \Phi(t,d_1,\dots,d_m)$. With the definition of pos above we have learned that the inverse terms in the equations in Φ access particular positions within t. The following lemma specifies the exact way how to replace an occurrence ω of a subterm in t, where ω is of sufficient length, by an arbitrary $\mathcal{L}$-term e without endangering the validity of the equations in Φ.

Lemma 8. *Let x, y_1, …, y_m be variables. Let Φ_x be a set of equations of the form $v(x) = v'(x)$, where v, v' are inverse terms. Let Φ_y be a set of equations of the form $u(x) = u'(y_i)$, where u is an inverse term, u' is an inverse term or a constant, and $i \in \{1, \dots, m\}$. Let t, d_1, …, $d_m \in T$ such that*

$$\mathbf{T}^* \models v(t) = v'(t) \quad \text{for all} \quad v(x) = v'(x) \in \Phi_x, \qquad \mathbf{T}^* \models u(t) = u'(d_i) \quad \text{for all} \quad u(x) = u'(y_i) \in \Phi_y.$$

Then for any position $\mu \in \Omega(t)$ with $|\mu| > \Delta_x(\Phi_x \cup \Phi_y) \cdot (\#\Phi_x + 1)$ there are sets P_μ, $D_\mu \subseteq \Omega(t)$ with the following properties:

1. *For all ω, $\omega' \in P_\mu$ we have $t\omega = t\omega'$.*
2. *For all $\omega \in D_\mu$ we have that $t\omega$ is a subterm of one of the d_1, …, d_m.*
3. *Any two positions ω, $\omega' \in P_\mu \cup D_\mu$ are uncomparable, i.e., neither $\omega \preccurlyeq \omega'$ nor $\omega' \preccurlyeq \omega$.*
4. *$P_\mu \cap D_\mu = \varnothing$.*
5. *Suppose that $\mu' \notin D_\mu$ for all $\mu' \preccurlyeq \mu$. Let $e \in T$.*
   If there is $\mu' \in P_\mu$ for some $\mu' \preccurlyeq \mu$, then this μ' is unique by property (3), and we set
   $$t' = \big\langle t\mu'\langle e, t\mu, \{t\mu\}\rangle, t\mu', P_\mu\big\rangle.$$
   If there is no such μ', then we set $t' = t\langle e, t\mu, \{t\mu\}\rangle$.
   In either case it follows that the obtained t' can take the role of the original t in all our equations:
   $$\mathbf{T}^* \models v(t') = v'(t') \quad \text{for all} \quad v(x) = v'(x) \in \Phi_x,$$
   $$\mathbf{T}^* \models u(t') = u'(d_i) \quad \text{for all} \quad u(x) = u'(y_i) \in \Phi_y.$$
6. *For all $\omega \in P_\mu \cup D_\mu$ we have $|\omega| \leqslant \Delta_x(\Phi_y) \cdot (\#\Phi_x + 1)$.*

Proof. Let $\mu \in \Omega(t)$ with $|\mu| > \Delta_x(\Phi_x \cup \Phi_y) \cdot (\#\Phi_x + 1)$. We prove the existence of suitable P_μ and D_μ by induction on $\#\Phi_x$. If $\#\Phi_x = 0$, i.e. $\Phi_x = \varnothing$, then we define some auxiliary

$$D^*_\mu = \big\{ \operatorname{pos}(u, t) \in \Omega(t) \ \big|\ u(x) = u'(y_i) \in \Phi_y \big\},$$

set $P_\mu = \varnothing$ and let D_μ be the set of all $\preccurlyeq$-minimal elements of D^*_μ. For this choice we verify properties 1–6:

1. This trivially follows from $P_\mu = \varnothing$.
2. Let $\omega \in D_\mu$. Then there is $u(x) = u'(y_i) \in \Phi_y$ such that $\omega = \operatorname{pos}(u, t)$, and it follows that
   $$t\omega = u^{\mathbf{T}^*}(t) = u'^{\mathbf{T}^*}(d_i),$$
   which is obviously a subterm of d_i.
3. Let ω, $\omega' \in D_\mu$. Then by the minimality of ω and ω' in D^*_μ these positions are uncomparable.
4. This trivially follows from $P_\mu = \varnothing$.
5. Suppose $\mu' \notin D_\mu$ for all $\mu' \preccurlyeq \mu$, and let $e \in T$. Since $P_\mu = \varnothing$, we certainly obtain $t' = t\langle e, t\mu, \{t\mu\}\rangle$. Recall that we are in the case that $\Phi_x = \varnothing$. For the equations in Φ_y assume for a contradiction that there is $u(x) = u'(y_i) \in \Phi_y$ with $u^{\mathbf{T}^*}(t) = u'^{\mathbf{T}^*}(d_i)$ but $u^{\mathbf{T}^*}(t') \neq u'^{\mathbf{T}^*}(d_i)$. Note that by definition of D^*_μ we have $\operatorname{pos}(u, t) \in D^*_\mu$. So there exists an equation $w(x) = w'(y_j) \in \Phi_y$ with $w^{\mathbf{T}^*}(t) = w'^{\mathbf{T}^*}(d_j)$ and $\operatorname{pos}(w, t) \preccurlyeq \operatorname{pos}(u, t)$ and $\operatorname{pos}(w, t) \in D_\mu$. Hence $\operatorname{pos}(w, t) \not\preccurlyeq \mu$, and so $\operatorname{pos}(u, t) \not\preccurlyeq \mu$. This implies that $u^{\mathbf{T}^*}(t) \neq u^{\mathbf{T}^*}(t')$, and thus we must have $\operatorname{pos}(u, t) \preccurlyeq \mu$, a contradiction.
6. Let $\omega \in D_\mu$. Then $\omega = \operatorname{pos}(u, t)$ for some $u(x) = u'(y_i) \in \Phi_y$. Thus
   $$|\omega| = |\operatorname{pos}(u, t)| \leqslant \Delta_x(u) \leqslant \Delta_x(\Phi_y) = \Delta_x(\Phi_y) \cdot (\#\Phi_x + 1).$$

For the induction step let now $\#\Phi_x > 0$, say $\Phi_x = \Phi'_x \cup \{u_0(x) = u'_0(x)\}$ with $\#\Phi'_x < \#\Phi_x$. By the induction hypothesis there exist suitable sets P'_μ and D'_μ for Φ'_x and Φ_y. Denote $\pi = \operatorname{pos}(u_0, t)$ and $\pi' = \operatorname{pos}(u'_0, t)$. If $\pi = \pi'$, then we set

$$P_\mu = \varnothing \quad \text{and} \quad D_\mu = D'_\mu.$$

Else π and π' are uncomparable. We distinguish cases on the comparability of π and π' on one hand with μ and positions in P'_μ and D'_μ on the other hand. We do so w.l.o.g only for π. If $\pi \not\preccurlyeq \mu$, then we again set

$$P_\mu = \varnothing \quad \text{and} \quad D_\mu = D'_\mu.$$

From now on we assume that $\pi \preccurlyeq \mu$.

Case 1 There exists $\mu' \preccurlyeq \mu$ with $\mu' \in D'_\mu$: Then we set

$$P_\mu = \varnothing \quad \text{and} \quad D_\mu = D'_\mu.$$

Case 2 There exists $\omega \in P'_\mu$ such that $\omega \preccurlyeq \pi$: Note that by Property 3 of the induction hypothesis, there is no $\mu' \in P'_\mu \cup D'_\mu$ such that $\mu' \preccurlyeq \mu$ and $\pi \prec \mu' \preccurlyeq \mu$. There is a unique $\nu \in \mathbb{N}^*$ such that $\omega \circ \nu = \pi$ and furthermore a unique ϱ such that $\omega \circ \nu \circ \varrho = \mu$. We distinguish four subcases:

2.1 There is $\omega' \in P'_\mu$ such that $\omega' \preccurlyeq \pi'$:
Let $\nu' \in \mathbb{N}^*$ be the unique choice such that $\pi' = \omega' \circ \nu'$, and set

$$P_\mu = \{\, \omega'' \circ \nu, \omega'' \circ \nu' \mid \omega'' \in P'_\mu \,\} \quad \text{and} \quad D_\mu = D'_\mu.$$

2.2 There exists ν' with $\pi' \preccurlyeq \pi' \circ \nu' \preccurlyeq \pi' \circ \varrho$ and $\pi' \circ \nu' \in P'_\mu$: Then we set

$$P_\mu = \bigl(P'_\mu \setminus \{\omega\}\bigr) \cup \{\, \omega'' \circ \nu \circ \nu' \mid \omega'' \in P'_\mu \,\} \quad \text{and} \quad D_\mu = D'_\mu.$$

2.3 There exists ν' with $\pi' \preccurlyeq \pi' \circ \nu' \preccurlyeq \pi' \circ \varrho$ and $\pi' \circ \nu' \in D'_\mu$: Then we set

$$P_\mu = \varnothing \quad \text{and} \quad D_\mu = D'_\mu \cup \{\pi \circ \nu'\}.$$

2.4 Else there is no position in $D'_\mu \cup P'_\mu$ comparable with π': We set

$$P_\mu = \bigl(P'_\mu \setminus \{\omega\}\bigr) \cup \{\pi, \pi'\} \quad \text{and} \quad D_\mu = D'_\mu.$$

Case 3 There exists $\omega \in P'_\mu$ with $\pi \preccurlyeq \omega \preccurlyeq \mu$: say $\omega = \pi \circ \nu$ and $\mu = \pi \circ \nu \circ \varrho$. Put $\mu' = \pi' \circ \nu \circ \varrho$. We again distinguish four subcases:

3.1 There exists $\omega' \in D'_\mu$ with $\omega' \preccurlyeq \pi'$: Then set

$$P_\mu = \varnothing \quad \text{and} \quad D_\mu = \bigl(D'_\mu \setminus \{\omega'\}\bigr) \cup \{\pi, \pi'\}.$$

3.2 There exists ν' with $\pi' \circ \nu' \in D'_\mu$ and $\nu' \preccurlyeq \nu \circ \varrho$: Then set

$$P_\mu = \varnothing \quad \text{and} \quad D_\mu = D'_\mu \cup \{\pi \circ \nu'\}.$$

3.3 There is no element of P'_μ comparable to π': Then

$$P_\mu = P'_\mu \cup \{\, \pi' \circ \nu' \mid \pi \circ \nu' \in P'_\mu \,\} \quad \text{and} \quad D_\mu = D'_\mu.$$

3.4 There is $\omega' \in P'_\mu$ with $\pi' \preccurlyeq \omega'$: Then we set

$$P^*_\mu = P'_\mu \cup \{\, \pi' \circ \nu' \mid \pi \circ \nu' \in P'_\mu \,\}, \quad D_\mu = D'_\mu,$$

and we set P_μ to be the set of all $\preccurlyeq$-maximal Elements of P^*_μ.

It is now not too hard to verify properties 1–6 for the sets P_μ and D_μ obtained this way. □

Theorem 9 (Elimination Theorem). *Let Φ be a nonempty set of $\mathcal{L}^*$-equations, let x, $y_1, \ldots, y_m$ be the variables occurring in the equations in Φ, and let b, $d_1, \ldots, d_m \in T$. Then there exists a Φ-restricted term $t(y_1, \ldots, y_m)$ such that for all equations $\psi(x, y_1, \ldots, y_m) \in \Phi$ we have*

$$\mathbf{T}^* \models \psi(b, d_1, \ldots, d_m) \quad \textit{iff} \quad \mathbf{T}^* \models \psi\bigl(t^{\mathbf{T}^*}(d_1, \ldots, d_m), d_1, \ldots, d_m\bigr).$$

Proof. For the proof, we discuss our three types of languages, which are U-1, U-N, and N-N, separately.

To begin with, let $\mathcal{L}$ be a language of type U-1. For an arbitrary set Ψ of equations we define $\overline{\Psi} = \{\, \neg\psi \mid \psi \in \Psi \,\}$. We partition $\Phi = \Phi_x \dot\cup \Phi_y$ where Φ_x is the set of equations in Φ of the form $v(x) = v'(x)$ and Φ_y is the set of equations in Φ of the form $u(x) = u'(y_i)$, and furthermore $\Phi_x = \Phi_x^+ \dot\cup \Phi_x^-$ and $\Phi_y = \Phi_y^+ \dot\cup \Phi_y^-$ such that

$$\mathbf{T}^* \models (\Phi_x^+ \cup \Phi_y^+)(b, d_1, \ldots, d_m) \quad \text{and} \quad \mathbf{T}^* \models \bigl(\overline{\Phi_x^-} \cup \overline{\Phi_y^-}\bigr)(b, d_1, \ldots, d_m).$$

We have to show that there exists a Φ-restricted term $t(y_1, \ldots, y_m)$ such that

$$\mathbf{T}^* \models (\Phi_x^+ \cup \Phi_y^+)(t(d_1, \ldots, d_m), d_1, \ldots, d_m),$$
$$\mathbf{T}^* \models \bigl(\overline{\Phi_x^-} \cup \overline{\Phi_y^-}\bigr)(t(d_1, \ldots, d_m), d_1, \ldots, d_m).$$

In fact, we are going to show that there is an $\mathcal{L}^*$-term in normal form of the more restricted depth $\Delta(t) \leqslant \Delta_x(\Phi) + \Delta_y(\Phi) + \#\Phi_x^- + \#\Phi_y^-$.

We make the important observation that since there are not function symbols in $\mathcal{L}$ of arity greater than 1, all positions in $\Omega(b)$ are of the form $()$, (1), $(1, 1)$, … and thus comparable. We distinguish cases on the existence of positive mixed equations:

Case 1: $\Phi_y^+ \neq \varnothing$, say $u(x) = u'(y_i) \in \Phi_y^+$. Define

$$t(y_1, \dots, y_m) = b\langle u'(y_i), u^{\mathbf{T}^*}(b), \{\mathrm{pos}(u, b)\}\rangle.$$

Then obviously $b = t^{\mathbf{T}^*}(d_1, \dots, d_m)$ and so

$$\mathbf{T}^* \models (\Phi_x^+ \cup \Phi_y^+)(t(d_1, \dots, d_m), d_1, \dots, d_m),$$
$$\mathbf{T}^* \models \big(\overline{\Phi_x^-} \cup \overline{\Phi_y^-}\big)(t(d_1, \dots, d_m), d_1, \dots, d_m).$$

From $\Delta(u'(y_i)) \leqslant \Delta_y(\Phi)$, it follows that

$$\Delta(t) \leqslant |\,\mathrm{pos}(u, b)| + \Delta_y(\Phi) \leqslant \Delta_x(\Phi) + \Delta_y(\Phi).$$

Since $b \in T$ is an $\mathcal{L}$-term and $u'(y_i)$ is an inverse term, it follows that t is in normal form.

Case 2: $\Phi_y^+ = \varnothing$. If $\Delta(b) \leqslant \Delta_x(\Phi) + \Delta_y(\Phi) + \#\Phi_x + \#\Phi_y$, then b satisfies our depth bound, and, as an $\mathcal{L}$-term, b is in normal form. So we simply set

$$t(y_1, \dots, y_m) = b.$$

It remains to treat the case that $\Delta(b) > \Delta_x(\Phi) + \Delta_y(\Phi) + \#\Phi_x + \#\Phi_y$. Let

$$k = \#\Phi_x^- + \#\Phi_y^- \leqslant \#\Phi_x + \#\Phi_y,$$

and consider the set G_k of $\mathcal{L}$-terms of Lemma 7(i) with $|G_k| = k + 1$ and $\Delta(G_k) = k$, and name its $k + 1$ elements like $G_k = \{g_0, \dots, g_k\}$. For the unique position $\omega = (1, \dots, 1) \in \Omega(b)$ with $|\omega| = \Delta_x(\Phi) + \Delta_y(\Phi) + 1 > \Delta_x(\Phi)$ put

$$b_i = b\langle g_i, b\omega, \{\omega\}\rangle \quad \text{for} \quad i \in \{0, \dots, k\}.$$

Notice that for any equation $v(x) = v'(x) \in \Phi_x^+$ there exists a single position $\omega^* \in \Omega(b)$ with $\omega^* = \mathrm{pos}(v, b) = \mathrm{pos}(v', b)$ and $|\omega^*| \leqslant \Delta_x(\Phi)$, and we have $v^{\mathbf{T}^*}(b) = v'^{\mathbf{T}^*}(b) = b\omega^*$. It is easy to see that all these equations hold also for our replacements $b_i = b\langle g_i, b\omega, \{\omega\}\rangle$ because $|\omega| > \Delta_x(\Phi)$. Hence

$$\mathbf{T}^* \models \Phi_x^+(b_i, d_1, \dots, d_m) \quad \text{for} \quad i \in \{0, \dots, k\}.$$

For any equation $v(x) = v'(x) \in \Phi_x^-$ we have

$$b(1, \dots, 1) = b\,\mathrm{pos}(v, b) = v^{\mathbf{T}^*}(b) \neq v'^{\mathbf{T}^*}(b) = b\,\mathrm{pos}(v', b) = b(1, \dots, 1),$$

where $\mathrm{pos}(v, b)$, $\mathrm{pos}(v', b) \in \Omega(b)$ with $|\,\mathrm{pos}(v, b)|$, $|\,\mathrm{pos}(v', b)| \leqslant \Delta_x(\Phi)$. It follows immediately that even $|\,\mathrm{pos}(v, b)| \neq |\,\mathrm{pos}(v', b)|$, which is invariant under our replacements $b_i = b\langle g_i, b\omega, \{\omega\}\rangle$ because $|\omega| > \Delta_x(\Phi)$. Hence also

$$\mathbf{T}^* \models \Phi_x^-(b_i, d_1, \dots, d_m) \quad \text{for} \quad i \in \{0, \dots, k\}.$$

We finally turn to the equations Φ_y^-. Let $u(x) = u'(y_i) \in \Phi_y^-$, and let i, $j \in \{0, \dots, k\}$ with $i \neq j$. Then obviously $b_i \neq b_j$. Assume for a contradiction that $u^{\mathbf{T}^*}(b_i) = u^{\mathbf{T}^*}(b_j)$. It is easy to see that $u^{\mathbf{T}^*}$ removes at most $\Delta_y(\Phi)$ many outermost function symbols from b_i and b_j and that on these initial segments both b_i and b_j are equal to b. This implies $b_i = b_j$ a contradiction. Thus

$$|\{u^{\mathbf{T}^*}(b_0), \dots, u^{\mathbf{T}^*}(b_k)\}| = k + 1 > \#\Phi_y \geqslant \#\Phi_y^-,$$

and according to the pigeon hole principle, there is at least one choice $b_{i_0} \in \{b_0, \dots, b_k\}$ such that

$$\mathbf{T}^* \models \overline{\Phi_y^-}(b_{i_0}, d_1, \dots, d_m).$$

So we may put $t(y_1, \dots, y_m) = b_{i_0}$. Then t is an $\mathcal{L}$-term and thus in normal form, and we have that $\Delta(t) \leqslant \omega + k \leqslant \Delta_x(\Phi) + \Delta_y(\Phi) + \#\Phi_x + \#\Phi_y$.

For languages of type U-N, we proceed exactly the same way as in the case of type U-1 above, but we let $k = \log(\#\Phi_x^- + \#\Phi_y^- + 1)$ and take the corresponding set G_k from Lemma 7(ii). Then

$$|G_k| = 2^k = 2^{\log(\#\Phi_x^- + \#\Phi_y^- + 1)} = \#\Phi_x^- + \#\Phi_y^- + 1,$$
$$\Delta(G_k) = k = \log(\#\Phi_x^- + \#\Phi_y^- + 1).$$

This leads to an improved depth bound

$$\Delta(t) \leqslant \omega + k \leqslant \Delta_x(\Phi) + \Delta_y(\Phi) + \log(\#\Phi_x^- + \#\Phi_y^- + 1).$$

We finally turn to the case of languages of type N-N. We consider the set

$$B' = \{\, b' \in T \mid \mathbf{T}^* \models \psi(b', d_1, \dots, d_m) = \psi(b, d_1, \dots, d_m) \,\}.$$

Obviously, $b \in B'$ and thus $B' \neq \varnothing$. For each term $b' \in B'$ we consider the set $\Omega(b')$ of all corresponding tree positions. From this set $\Omega(b')$ we isolate two subsets:

1. The set $D_{b'}$ of all positions that firstly contribute to a subterm of b' that is equal to some subterm of d_i for some $i \in \{1, \dots, m\}$ starting in d_i not deeper than $\Delta_y(\Phi)$, and secondly occur in b' not deeper than $\Delta(\Phi) \cdot (\#\Phi + 1)$:
$$S(d_1, \dots, d_m) = \bigcup_{i=1}^{m} \{\, d_i\omega \mid \omega \in \Omega(d_i) \text{ and } |\omega| \leqslant \Delta_y(\Phi) \,\},$$
$$D_{b'} = \bigl\{\, \omega \in \Omega(b') \bigm| b'\omega \in S(d_1, \dots, d_m) \text{ and } |\omega| \leqslant \Delta(\Phi) \cdot (\#\Phi + 1) \,\bigr\}.$$
2. The set $\tilde{\Omega}(b')$ of all positions not located within a subtree starting at a position in $D_{b'}$:
$$\tilde{\Omega}(b') = \bigl\{\, \omega \in \Omega(b') \bigm| \omega' \notin D_{b'} \text{ for all } \omega' \in \Omega(b') \text{ with } \omega' \preccurlyeq \omega \,\bigr\}.$$

Notice that $\tilde{\Omega}(b')$ establishes one connected tree. We define

$$\tilde{\Delta}(b') = \max_{w \in \tilde{\Omega}(b')} |w|$$

to be the maximal length of positions in $\tilde{\Omega}(b')$. Consider all $b' \in B'$ such that $\tilde{\Delta}(b')$ is minimal. Among these choose one particular b' such that the number

$$\#\bigl\{\, \omega \in \tilde{\Omega}(b') \bigm| |\omega| = \tilde{\Delta}(b') \,\bigr\}$$

of positions of length $\tilde{\Delta}(b')$ is in turn minimal.

We simplify notation, and write Ω, D, $\tilde{\Omega}$, and $\tilde{\Delta}$ omitting our fixed b'. We claim that

$$\tilde{\Delta} \leqslant \Delta(\Phi) \cdot (\#\Phi + 1) + \log \#\Phi + 1.$$

Assume for a contradiction that $\tilde{\Delta} > \Delta(\Phi) \cdot (\#\Phi + 1) + \log(\#\Phi + 1)$. Pick $\mu \in \tilde{\Omega}$ with $|\mu| = \Delta(\Phi) \cdot (\#\Phi + 1) + 1$. We are going to apply Lemma 8 to this position μ and choose t as b', Φ_x as Φ^+, Φ_y as Φ_y^+, and e corresponding to one of the elements of G_k chosen for $k = \log(\#\Phi_x^- + \#\Phi_y^- + 1)$ according to Lemma 7(iii). As in the case of type U-N, we have

$$|G_k| \geqslant 2^{\log(\#\Phi_x^- + \#\Phi_y^- + 1)} = \#\Phi_x^- + \#\Phi_y^- + 1 \quad \text{and} \quad \Delta(G_k) = \log(\#\Phi_x^- + \#\Phi_y^- + 1).$$

By virtue of the fact that $\mu \in \tilde{\Omega}$ it follows that $\mu' \notin D_\mu$, where D_μ is defined in Lemma 8, for all $\mu' \preccurlyeq \mu$. So the substitution of the terms from G_k as described in property 5 of Lemma 8 results in new $\mathcal{L}$-terms $b_0, \dots, b_k$ such that

$$\mathbf{T}^* \models (\Phi_x^+ \cup \Phi_y^+)(b_i, d_1, \dots, d_m) \quad \text{for} \quad i \in \{0, \dots, k\}.$$

Moreover, we obtain in analogy to the case of type U-1 that by the pigeon hole principle at least one $b_{i_0} \in \{b_0, \dots, b_k\}$ will satisfy

$$\mathbf{T}^* \models \bigl(\overline{\Phi_x^-} \cup \overline{\Phi_y^-}\bigr)(b_{i_0}, d_1, \dots, d_m).$$

By construction, we have $\tilde{\Delta}(b_{i_0}) \leqslant \Delta(\Phi) \cdot (\#\Phi + 1) + \log(\#\Phi + 1)$ contradicting the minimal choice of b'. This proves our claim.

Next, we replace in b' each occurrence $d_i' \in D$ of some subterm of d_i in b' by a corresponding inverse term $v_i(y_i)$, such that $v_i^{\mathbf{T}^*}(d_i) = d_i'$. Notice that by definition of D the depth of this $v_i(y_i)$ is bounded by $\Delta_y(\Phi)$. As a consequence the resulting $\mathcal{L}^*$-term b'' has a depth bounded by $\Delta(\Phi) \cdot (\#\Phi + 1) + \log(\#\Phi + 1)$.

Thus we finally have a Φ-restricted $\mathcal{L}^*$-term $b''(y_1, \ldots, y_m)$ containing at most the variables $y_1, \ldots, y_m$ such that $b''^{\mathbf{T}^*}(d_1, \ldots, d_m) = b'$. So b'' satisfies the theorem. ◻

Corollary 10 (Quantifier elimination). *Let φ be a quantifier-free $\mathcal{L}^*$-formula with variables x, y_1, …, y_m. Let $\tilde{\varphi}$ be an equivalent and-or-combination of equations and negated equations in refined normal form according to Lemma 6. Let Φ be the set of all atomic subformulas of $\tilde{\varphi}$, and denote by R_Φ the set of Φ-restricted terms for the considered type of language. Then*

$$\mathbf{T}^* \models \exists x \varphi \longleftrightarrow \bigvee_{t \in R_\Phi} \varphi[t/x].$$

Proof. The implication from the left to the right is an immediate consequence of Theorem 9. The converse follows from the fact that for fixed parameters $y_1, \ldots, y_m$, the terms $t \in R_\Phi$ evaluate to elements of T. ◻

Using double negation $\mathbf{T}^* \models \forall x \varphi \longleftrightarrow \neg \exists x \neg \varphi$ we can by means of Corollary 10 as well eliminate universal quantifiers. Furthermore, every first-order formula can easily be transformed into an equivalent one where all quantifiers are prenex. Then all quantifiers can be eliminated successively beginning with the innermost one. Notice that for the elimination of a block of like quantifiers, each disjunction obtained by the elimination of a quantifier can be interchanged with the next quantifier before continuing elimination.

Corollary 11 (Extended quantifier elimination). *Let φ be a quantifier-free $\mathcal{L}^*$-formula with variables x, y_1, …, y_m. Let $\tilde{\varphi}$ be an equivalent and-or-combination of equations and negated equations in refined normal form according to Lemma 6. Let Φ be the set of all atomic subformulas of $\tilde{\varphi}$, and denote by R_Φ the set of Φ-restricted terms for the considered type of language. Let $R_\Phi = \{t_1, \ldots, t_n\}$, and consider the scheme*

$$\left[\begin{array}{c|c} \varphi[t_1/x] & x = t_1 \\ \vdots & \vdots \\ \varphi[t_n/x] & x = t_n \end{array}\right].$$

Then for fixed values of the parameters y_1, …, y_m we have that $\mathbf{T}^* \models \exists x \varphi$ if and only if $\mathbf{T}^* \models \varphi[t_i/x]$ for at least on $i \in \{1, \ldots, n\}$, and in the positive case the corresponding $x = t_i$ is one possible choice for x. ◻

Considering again prenex formulas, one can collect the sample solutions for an entire block of existential quantifiers. If there are besides an outermost block of existential quantifiers further quantifier blocks inside, these can be eliminated by regular quantifier elimination beforehand.

5 Complexity

In this section we compute upper complexity bounds on the space and time required by our quantifier elimination procedure when applied iteratively to an arbitrary prenex formula.

Consider an input formula $\exists x \varphi(x, y_1, \ldots, y_m)$ with quantifier-free φ, depth $\Delta(\varphi)$, and $\operatorname{at}(\varphi)$ many atomic subformulas. Recall using the results of Section 3, we will first compute a refined normal form $\tilde{\varphi}$ of φ. Then, as discussed in Section 4, we will switch from $\tilde{\varphi}$ to the set Φ of equations in $\tilde{\varphi}$. We have in Section 4 given quite detailed complexity bounds on the sets R_Φ of Φ-restricted terms. For bounding the complexity of the entire elimination procedure, these bounds have to be in turn bounded in terms of the original input formula $\exists x \varphi$. For this we using the complexity information given in the various normal form lemmas in Section 3: For the depth of Φ, we obtain

$$\Delta(\Phi) = \Delta(\tilde{\varphi}) \leqslant \Delta(\varphi).$$

Our bound on $\#\Phi$ depends on the type of the considered language. For languages of type U-1 and U-N we can according to the remarks on languages with only unary function symbols in the normal form lemmas 5 and 6 bound $\#\Phi$ as follows:

$$\#\Phi \leqslant \operatorname{at}(\tilde{\varphi}) \leqslant \Delta(\varphi) \cdot \operatorname{at}(\varphi).$$

Here $\mathrm{at}(\tilde\varphi)$ and $\mathrm{at}(\varphi)$ denote the number of atomic formulas and $\tilde\varphi$ and φ, respectively. In the case of type N-N we have, in contrast, only the weaker bound

$$\#\Phi \leqslant \mathrm{at}(\tilde\varphi) \leqslant 2^{O(\Delta(\varphi))} \cdot \mathrm{at}(\varphi).$$

Our input formula $\exists x\varphi$ is equivalent in $\mathbf{T}^*$ to the quantifier-free disjunction $\varphi_1 = \bigvee_{t\in R_\Phi} \varphi[t/x]$. We compute some characteristic quantities for this result φ_1 for our three types of languages, viz. the maximal depth $\Delta(\varphi_1)$ of the contained terms, the number of disjuncts obtained, which obviously equals $\#R_\Phi$, and the maximal word length of a single disjunct, which equals $|\varphi[t/x]|$ for some $t \in R_\Phi$. In addition to the following results, notice the invariant $\mathrm{at}(\varphi[t/x]) = \mathrm{at}(\varphi)$:

Type U-1

$$\begin{aligned}
\Delta(\varphi_1) &\leqslant \Delta(\varphi) + \Delta(R_\Phi) = \Delta(\varphi) + 2\,\Delta(\Phi) + \#\Phi \leqslant \Delta(\varphi)\cdot\bigl(3+\mathrm{at}(\varphi)\bigr) = O\bigl(\Delta(\varphi)\cdot\mathrm{at}(\varphi)\bigr)\\
\#R_\Phi &= O\bigl(m\cdot(\Delta(\Phi)+\#\Phi)^2\bigr) = O\bigl(m\cdot\Delta(\varphi)^2\cdot\mathrm{at}(\varphi)^2\bigr)\\
\bigl|\varphi[t/x]\bigr| &= |\varphi|\cdot|R_\Phi| = |\varphi|\cdot O\bigl(\Delta(\Phi)+\#\Phi\bigr) = O\bigl(|\varphi|\cdot\Delta(\varphi)\cdot\mathrm{at}(\varphi)\bigr) \qquad (t\in R_\Phi)
\end{aligned}$$

Type U-N

$$\begin{aligned}
\Delta(\varphi_1) &\leqslant \Delta(\varphi) + \Delta(R_\Phi) = \Delta(\varphi) + 2\,\Delta(\Phi) + \log\#\Phi \leqslant 4\,\Delta(\varphi) + \log\mathrm{at}(\varphi) = O\bigl(\Delta(\varphi)+\log\mathrm{at}(\varphi)\bigr)\\
\#R_\Phi &= 2^{O(\log m\cdot(\Delta(\Phi)+\log\#\Phi))} = 2^{O(\log m\cdot(\Delta(\varphi)+\log\mathrm{at}(\varphi)))}\\
\bigl|\varphi[t/x]\bigr| &= |\varphi|\cdot|R_\Phi| = |\varphi|\cdot O\bigl(\Delta(\Phi)+\log\#\Phi\bigr) = O\bigl(|\varphi|\cdot(\Delta(\varphi)+\log\mathrm{at}(\varphi))\bigr) \qquad (t\in R_\Phi)
\end{aligned}$$

Type N-N

$$\begin{aligned}
\Delta(\varphi_1) &\leqslant \Delta(\varphi) + \Delta(R_\Phi) = \Delta(\varphi) + \Delta(\Phi)\cdot(\#\Phi+1) + \log\#\Phi = 2^{O(\Delta(\varphi))}\cdot\mathrm{at}(\varphi)\\
\#R_\Phi &= 2^{\log m\cdot 2^{O(\Delta(\Phi)\cdot\#\Phi)}} = 2^{\log m\cdot 2^{2^{O(\Delta(\varphi))}\cdot\mathrm{at}(\varphi)}}\\
\bigl|\varphi[t/x]\bigr| &= |\varphi|\cdot|R_\Phi| = |\varphi|\cdot 2^{O(\Delta(\Phi)\cdot\#\Phi)} = |\varphi|\cdot 2^{2^{O(\Delta(\Phi))}\cdot\mathrm{at}(\varphi)} \qquad (t\in R_\Phi)
\end{aligned}$$

On this basis we obtain the following results for the elimination of a single quantifier block. For $n\in\mathbb{N}$ we denote by $\exp^n(x)$ an n-fold iteration of the exponential function, like

$$2^{2^{\cdot^{\cdot^{2^x}}}}.$$

Theorem 12 (Elimination of one block). *Let φ' be the result of eliminating the prenex block of k like quantifiers from $\exists x_1\ldots\exists x_k\varphi$.*

(i) For languages of type U-1, we have

$$|\varphi'| = |\varphi|^{O(k^2)}.$$

This is singly exponential in the input word length. More precisely, it is singly exponential in the number of quantifiers, and polynomial in $|\varphi|$.

(ii) For languages of type U-N, we have

$$|\varphi'| = |\varphi|\cdot 2^{2^{O(k)}\cdot\log m\cdot(\Delta(\varphi)+\log\mathrm{at}(\varphi))}.$$

This is doubly exponential in the input word length. More precisely, it is doubly exponential in the number of quantifiers, singly exponential in $|\varphi|$, and polynomial in $\mathrm{at}(\varphi)$.

(iii) For languages of type N-N, we have

$$|\varphi'| = |\varphi|\cdot 2^{\log m\cdot\exp^k(2^{O(\Delta(\varphi))}\cdot\mathrm{at}(\varphi))} = \exp^{k+2}\bigl(O(|\varphi|)\bigr)$$

For input word length l, this is l-fold exponential in l. More precisely, it is $k+2$-fold exponential in $\Delta(\varphi)$, $k+1$-fold exponential in $\mathrm{at}(\varphi)$, and polynomial in all other possible complexity parameters. □

For the elimination of several blocks of quantifiers we obtain the following complexity bounds:

Theorem 13 (Elimination of several blocks). *Let φ' be the result of eliminating the a prenex blocks of at most k like quantifiers each from*

$$\exists x_{11} \ldots \exists x_{1k} \forall x_{21} \ldots \forall x_{2k} \ldots \exists x_{a1} \ldots \exists x_{ak} \varphi.$$

(i) For languages of type U-1, we have

$$|\varphi'| = |\varphi|^{k^{O(a)}}.$$

This is doubly exponential in the input word length. More precisely, it is doubly exponential in the number of quantifier blocks, singly exponential in the number of quantifiers, and polynomial in $|\varphi|$.

(ii) For languages of type U-N, we have

$$|\varphi'| \leqslant 2^{2^{O(ak)} \cdot |\varphi|^2}.$$

This is doubly exponential in the input word length. More precisely, it is doubly exponential in the number of quantifiers and singly exponential in $|\varphi|$.

(iii) For languages of type N-N, we have

$$|\varphi'| = \exp^{a \cdot (k+2)}\big(O(|\varphi|)\big).$$

For input word length l, this is l^2-fold exponential in l, and reduces to l-fold exponential for $k \geqslant 2$. More precisely, it is $a \cdot (k+2)$-fold exponential in $|\varphi|$. □

In part (iii) of the above theorem, notice that in spite of the immense complexity, we can still make the well-known observation that changing quantifiers are more expensive than the same amount of like quantifiers. For instance, we obtain for inputs $\exists x_1 \exists x_2 \varphi$ on one hand and $\forall x_1 \exists x_2 \varphi$ on the other hand the respective complexity bounds

$$\exp^{1 \cdot (2+2)}\big(O(|\varphi|)\big) = 2^{2^{2^{2^{O(|\varphi|)}}}} \quad \text{and} \quad \exp^{2 \cdot (1+2)}\big(O(|\varphi|)\big) = 2^{2^{2^{2^{2^{2^{O(|\varphi|)}}}}}}.$$

Corollary 14 (Complexity of quantifier elimination).

(i) For languages of type U-1 and U-N, the quantifier elimination problem is in at most doubly exponential time-space. This is elementary recursive, and thus in the third class of the Grzegorcyk hierarchy.

(ii) For languages of type N-N, the quantifier elimination problem is not elementary recursive. It is, however, in the fourth class of the Grzegorcyk hierarchy. For a bounded number of quantifiers, the quantifier elimination problem is elementary recursive, i.e., in the third class of the Grzegorcyk hierarchy. □

Since the evaluation of Boolean combinations of variable-free $\mathcal{L}^*$-equations in $\mathbf{T}^*$ can obviously be performed in polynomial time, we get corresponding bounds for the decision problem:

Corollary 15 (Complexity of the decision problem).

(i) For languages of type U-1 and U-N, the decision problem is in at most doubly exponential time-space. This is elementary recursive, and thus in the third class of the Grzegorcyk hierarchy.

(ii) For languages of type N-N, the decision problem is not elementary recursive. It is, however, in the fourth class of the Grzegorcyk hierarchy. For a bounded number of quantifiers, the decision problem is elementary recursive, i.e., in the third class of the Grzegorcyk hierarchy. □

Corollaries 14 and 15 leave open the following problem: *Is the quantifier elimination problem or the decision problem elementary recursive for input formulas with a bounded number of quantifier blocks but an unbounded number of quantifiers?* In particular these questions are open for the class of existential formulas as inputs.

6 Conclusions

We have revisited the quantifier elimination and decision problem for suitably expanded absolutely free term algebras $\mathbf{T}^*$ for a given first-order functional language $\mathcal{L}$. Positive solutions for these problems were given by Malcev, Maher, Belegradek, Hodges without any upper complexity bounds. A non-elementary recursive lower bound was proved by Compton and Henson for languages $\mathcal{L}$ having at least one function symbol of arity greater than 1.

We have presented a new type of quantifier elimination for $\mathbf{T}^*$ that works by substitution of finitely many parametric test terms for the quantified variable. As a consequence it yields sample solutions for all variables in the outermost existential quantifier block, a feature that is important in logic programming.

Using our new quantifier elimination procedure we have obtained upper complexity bounds for the quantifier elimination and decision problem for expanded absolutely free term algebras $\mathbf{T}^*$ that match closely the known lower bounds. In particular they show that both problems are in the fourth Grzegorcyk class. Our bounds are quite detailed and refer to the quantifier structure of the input formulas. For languages $\mathcal{L}$ having only constants and unary function symbols we have obtained elementary recursive upper complexity bounds. The same applies to arbitrary languages $\mathcal{L}$ but input formulas of a bounded number of quantifiers.

The test term approach allow for a fast and efficient implementation, which is planned within the computer logic system REDLOG based on REDUCE. Within this framework there are most interesting connections to constraint logic programming, which will be certainly subject to further research.

References

[Bel88] O. Belegradek. Some model theory of locally free algebras (in Russian). In B. Dahn and H. Wolter, editors, *6th Easter Conference on Model Theory*, pages 28–32, Berlin, 1988. Humbold University.

[CH90] K. J. Compton and C. W. Henson. A uniform method for proving lower bounds on the computational complexity of logical theories. *Annals of Pure and Applied Logic*, 48:1–79, 1990.

[DS97] A. Dolzmann and T. Sturm. Redlog: Computer algebra meets computer logic. *ACM SIGSAM Bulletin*, 31(2):2–9, June 1997.

[DV00] E. Dantsin and A. Voronkov. Expressive power and data complexity of nonrecursive query languages for lists and trees. In *PODS 2000, Principles of Database Systems*, pages 157–165, 2000.

[Gri99] E. R. Griffor. *Handbook of Computability Theory*. Elsevier, Amsterdam, 1999.

[Hod93] W. Hodges. *Model Theory*. Cambridge University Press, 1993.

[LW93] R. Loos and V. Weispfenning. Applying linear quantifier elimination. *The Computer Journal*, 36(5):450–462, 1993. Special issue on computational quantifier elimination.

[Mah88] M. Maher. Complete axiomatizations of the algebras of finite, rational and infinite trees. In *Proc. IEEE Conference on Logic in Computer Science (LICS)*, pages 348–357, 1988.

[Mal71] A. I. Malcev. Axiomatizable classes of locally free algebras of various types. In Wells. B. F. III, editor, *The Metamathematics of Algebraic Systems, Collected papers*, chapter 23. North Holland Publishing Company, 1971.

[Rab77] M. O. Rabin. Decidable theories. In J. Barweise, editor, *Handbook of Mathematical Logic*, volume 90 of *Studies in Logic and the Foundations of Mathematics*, pages 595–629. North Holland Publishing Company, 1977.

[RV01] T. Rybina and A. Voronkov. A decision procedure for term algebras with queues. *ACM Transactions on Computational Logic*, 2(2):155–181, 2001.

[Sch74] C. P. Schnorr. *Rekursive Funktionen und ihre Komplexität*. Teubner, 1974.

[Stu02] T. Sturm. Quantifier elimination-based constraint logic programming. Technical Report MIP-0202, FMI, Universität Passau, D-94030 Passau, Germany, January 2002.

[VV98] S. Vorobyov and A. Voronkov. Complexity of nonrecursive logic programs with complex values. In *PODS'98, Principles of Database Systems*, pages 244–253, 1998.

[Wei88] V. Weispfenning. The complexity of linear problems in fields. *Journal of Symbolic Computation*, 5(1–2):3–27, February–April 1988.