REAL QUANTIFIER ELIMINATION FOR NON-PRENEX FORMULAS

ADAM STRZEBOŃSKI

ABSTRACT. In this paper we discuss quantifier elimination methods for non-prenex real polynomial formulas. The Cylindrical Algebraic Decomposition (CAD) algorithm requires input in prenex form. Transforming non-prenex formulas to prenex form increases the number of variables. To avoid this problem, we propose an alternative recursive algorithm using cylindrical algebraic formulas to represent intermediate results. We present an empirical comparison of the recursive algorithm with direct CAD computation.

1. INTRODUCTION

A real polynomial condition is a formula $f(x_1,\ldots,x_n)\rho 0$, where $f \in \mathbb{R}[x_1,\ldots,x_n]$ and ρ is one of $<,\leq,\geq,>,=$, or $\neq$. A real polynomial formula is a Boolean combination of real polynomial conditions. A quantified real polynomial formula is a formula constructed with real polynomial conditions using Boolean operators and quantifiers over real polynomial variables. By Tarski's theorem (see [35]), for any quantified real polynomial formula, there exists an equivalent quantifier-free real polynomial formula. The Cylindrical Algebraic Decomposition (CAD) algorithm [7, 6, 33] computes such a quantifier-free formula. The algorithm can also be used to compute an explicit representation of the solution set of a quantified real polynomial formula, that is the set of real values of free variables for which the formula is true. The CAD algorithm requires input in prenex form. Some applications, however, naturally lead to quantifier elimination problems with non-prenex real algebraic formulas. Every quantified real polynomial formula can be transformed to prenex form, but this transformation in general requires adding new variables. As shown in [3], the complexity of CAD computation grows at least exponentially in the number of added variables. A more efficient algorithm is proposed in [3]. The algorithm avoids adding new variables by constructing the CAD recursively. Intermediate results are represented as cylindrical algebraic decompositions with truth values attached to each cell. In [34] we proposed representation of semialgebraic sets as cylindrical algebraic formulas (CAF) and gave an algorithm for representing a prenex quantified Boolean combination of CAFs as a CAF in the free variables. A CAF consists of cylindrically arranged bounds on free variables, where each variable is bounded by algebraic functions of variables that precede it in the variable order (see Section 2 for a precise definition). In this paper we present a recursive algorithm computing a CAF representation of the solution set of an arbitrary quantified real polynomial formula. The algorithm uses CAFs to represent intermediate results. An advantage of using CAF representation over using cylindrical algebraic decompositions is that CAF representation provides explicit bounds on each variable and thus allows to reduce size of projection sets and terminate stack constructions early (see [34] for details).

Date: February 26, 2011.

Example 1. Let E_1, E_2, E_3 be ellipses given by

$$\begin{aligned} E_1 &= \{(x,y) : \frac{(x+5)^2}{4} + y^2 < 1\} \\ E_2 &= \{(x,y) : \frac{(x-3)^2}{4} + (y-3)^2 < 1\} \\ E_3 &= \{(x,y) : \frac{(x-2)^2}{4} + (y-1)^2 < 1\} \end{aligned}$$

Decide whether there is a circle with radius 5 containing E_1, properly intersecting E_2 and disjoint with E_3.

The problem can be expressed as the following non-prenex quantified real polynomial formula.

$$\begin{gathered} \exists_a \exists_b [(\forall_x \forall_y \tfrac{(x+5)^2}{4} + y^2 < 1 \Rightarrow (x-a)^2 + (y-b)^2 < 25) \wedge \\ (\exists_x \exists_y \tfrac{(x-3)^2}{4} + (y-3)^2 < 1 \wedge (x-a)^2 + (y-b)^2 = 25) \wedge \\ (\forall_x \forall_y \tfrac{(x-2)^2}{4} + y^2 < 1 \Rightarrow (x-a)^2 + (y-b)^2 > 25)] \end{gathered}$$

Transforming the formula to prenex form we obtain

$$\begin{gathered} \exists_a \exists_b \forall_{x_1} \forall_{y_1} \exists_{x_2} \exists_{y_2} \forall_{x_3} \forall_{y_3} (\tfrac{(x_1+5)^2}{4} + y_1^2 < 1 \Rightarrow (x_1-a)^2 + (y_1-b)^2 < 25) \wedge \\ (\tfrac{(x_2-3)^2}{4} + (y_2-3)^2 < 1 \wedge (x_2-a)^2 + (y_2-b)^2 = 25) \wedge \\ (\tfrac{(x_3-2)^2}{4} + y_3^2 < 1 \Rightarrow (x_3-a)^2 + (y_3-b)^2 > 25) \end{gathered}$$

Direct application of the CAD algorithm to the prenex formula gives $false$ after 158 seconds. On the other hand, a recursive algorithm computes a representation of

$$\forall_x \forall_y \frac{(x+5)^2}{4} + y^2 < 1 \Rightarrow (x-a)^2 + (y-b)^2 < 25$$

as a CAF $F_1(a,b)$ equal to

$$\begin{gathered} [a = -8 \wedge b = 0] \vee [-8 < a < -5 \wedge R_{1,2}(a) \leq b \leq R_{1,3}(a)] \vee [a = -5 \wedge -4 \leq b \leq 4] \vee \\ [-5 < a < -2 \wedge R_{1,2}(a) \leq b \leq R_{1,3}(a)] \vee [a = -2 \wedge b = 0] \end{gathered}$$

a representation of

$$\exists_x \exists_y \frac{(x-3)^2}{4} + (y-3)^2 < 1 \wedge (x-a)^2 + (y-b)^2 = 25$$

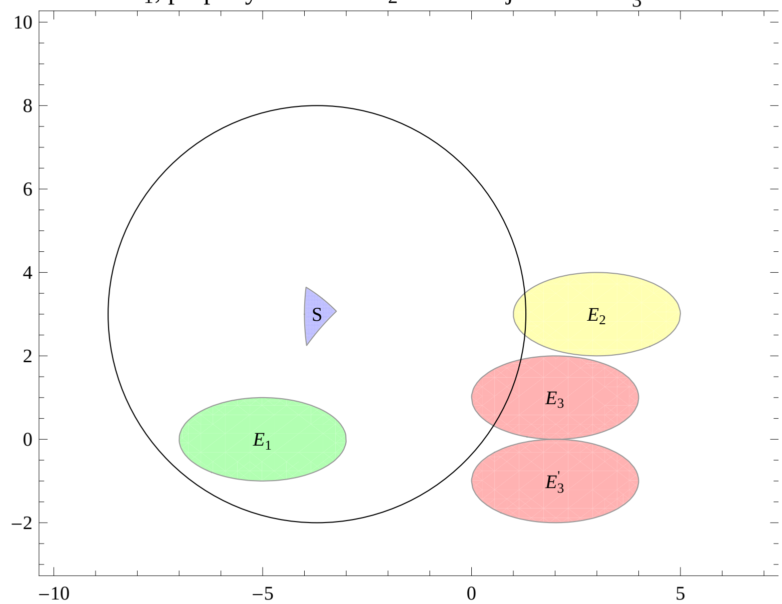
as a CAF $F_2(a,b)$ equal to

$$\begin{gathered} [-4 < a < 0 \wedge R_{2,1}(a) < b < R_{2,2}(a)] \vee \\ [0 \leq a < 3 \wedge (R_{2,1}(a) < b < R_{2,2}(a) \vee R_{2,3}(a) < b < R_{2,4}(a))] \vee \\ [a = 3 \wedge (-3 < b < -1 \vee 7 < b < 9)] \vee \\ [3 < a \leq 6 \wedge (R_{2,1}(a) < b < R_{2,2}(a) \vee R_{2,3}(a) < b < R_{2,4}(a))] \vee \\ [6 < a < 10 \wedge R_{2,1} p_2(a) < b < R_{2,2}(a)] \end{gathered}$$

and a representation of

$$\forall_x \forall_y \frac{(x-2)^2}{4} + y^2 < 1 \Rightarrow (x-a)^2 + (y-b)^2 > 25$$

FIGURE 1.1. There is no circle of radius 5 containing E_1, properly intersecting E_2 and disjoint with E_3. Each circle of radius 5 with center in S contains E_1, properly intersects E_2 and is disjoint with E_3'.



as a CAF $F_3(a,b)$ equal to

$$a \le -5 \vee [-5 < a < -1 \wedge (b \le R_{3,1}(a) \vee b \ge R_{3,2}(a))] \vee$$
$$[-1 \le a < 2 \wedge (b \le R_{3,1}(a) \vee b \ge R_{3,4}(a))] \vee [a = 2 \wedge (b \le -5 \vee b \ge 7)] \vee$$
$$[2 < a \le 5 \wedge (b \le R_{3,1}(a) \vee b \ge R_{3,4}(a))] \vee$$
$$[5 < a < 9 \wedge (b \le R_{3,1}(a) \vee b \ge R_{3,2}(a))] \vee a \ge 9$$

where $R_{i,j}(a)$ denotes the function of a equal to the j-th real root in t of $p_i(a,t)$, where p_1, p_2 and p_3 are bivariate polynomials in a and t of degree 8 in each variable. The computation times are, respectively, 1.41, 0.55 and 2.17 seconds. The computation of

$$\exists_a \exists_b F_1(a,b) \wedge F_2(a,b) \wedge F_3(a,b)$$

takes 0.4 seconds and returns $false$. Hence the total time for solving the problem using the recursive algorithm is 4.53 seconds.

If instead of ellipse E_3 we use ellipse E_3' given by

$$E_3' \quad = \quad \{(x,y) : \frac{(x-2)^2}{4} + (y+1)^2 < 1\}$$

the direct application of the CAD algorithm to the prenex version of the formula gives $true$ after 5.42 seconds and the recursive algorithm gives $true$ after 4.3 seconds. In this case there is little difference, since the CAD algorithm returns after finding the first cell on which the formula is true. If we want to find a full description of the set S of points (a,b) for which the circle centered at (a,b) with radius 5 contains E_1, properly intersects E_2 and is disjoint with E_3', the direct application of the CAD algorithm to the prenex version of the formula takes 207 seconds and the recursive algorithm takes 4.37 seconds.

The next section gives a precise definition of cylindrical algebraic formulas and specifications of main algorithms for computation with CAFs. Section 3 describes a recursive algorithm for computing CAF representations of solution sets of non-prenex quantified real polynomial formulas. Finally, the last section presents empirical comparison of the recursive algorithm with direct CAD computation.

2. Cylindrical Algebraic Formulas

Definition 2. A *real algebraic function* given by *defining polynomial* $f \in \mathbb{Z}[x_1,\ldots,x_n,y]$ and *root number* $p \in \mathbb{N}_+$ is the function

$$Root_{y,p} f : \mathbb{R}^n \ni (x_1,\ldots,x_n) \longrightarrow Root_{y,p} f(x_1,\ldots,x_n) \in \mathbb{R} \tag{2.1}$$

where $Root_{y,p} f(x_1,\ldots,x_n)$ is the p-th real root of f treated as a univariate polynomial in y. The function is defined for those values of $x_1,\ldots,x_n$ for which $f(x_1,\ldots,x_n,y)$ has at least p real roots. The real roots are ordered by the increasing value, counting multiplicities. A real algebraic number $Root_{y,p} f \in \mathbb{R}$ given by *defining polynomial* $f \in \mathbb{Z}[y]$ and *root number* p is the p-th real root of f. Let Alg be the set of real algebraic numbers and for $C \subseteq \mathbb{R}^n$ let Alg_C denote the set of all algebraic functions defined and continuous on C. (See [28, 31] for more details on how algebraic numbers and functions can be implemented in a computer algebra system.)

Definition 3. A set $P \subseteq \mathbb{R}[x_1,\ldots,x_n,y]$ is *delineable* over $C \subseteq \mathbb{R}^n$ iff

(1) $\forall f \in P\, \exists k_f \in \mathbb{N}\, \forall a \in C\, \sharp\{b \in \mathbb{R} : f(a,b) = 0\} = k_f$.
(2) For any $f \in P$ and $1 \le p \le k_f$, $Root_{y,p} f$ is a continuous function on C.
(3)
$$\begin{aligned}\forall f,g \in P \qquad &(\exists a \in C\, Root_{y,p} f(a) = Root_{y,q} g(a) \Leftrightarrow\\ &\forall a \in C\, Root_{y,p} f(a) = Root_{y,q} g(a))\end{aligned}$$

Definition 4. A *cylindrical system of algebraic constraints* in variables $x_1,\ldots,x_n$ is a sequence $A = (A_1,\ldots,A_n)$ satisfying the following conditions.

(1) For $1 \le k \le n$, A_k is a set of formulas
$$\begin{aligned}A_k = \{a_{i_1,\ldots,i_k}(x_1,\ldots,x_k) : 1 \le i_1 \le m \wedge\\ 1 \le i_2 \le m_{i_1} \wedge \ldots \wedge 1 \le i_k \le m_{i_1,\ldots,i_{k-1}}\}\end{aligned}$$
(2) For each $1 \le i_1 \le m$, $a_{i_1}(x_1)$ is *true* or
$$x_1 = r$$
where $r \in Alg$, or
$$r_1 \rho_1 x_1 \rho_2 r_2$$
where $r_1 \in Alg \cup \{-\infty\}$, $r_2 \in Alg \cup \{\infty\}$, $r_1 < r_2$ and $\rho_1, \rho_2 \in \{<, \le\}$. Moreover, if $s_1, s_2 \in Alg \cup \{-\infty, \infty\}$, s_1 appears in $a_u(x_1)$, s_2 appears in $a_v(x_1)$ and $u < v$ then $s_1 \le s_2$.
(3) Let $k < n$, $I = (i_1,\ldots,i_k)$ and let $C_I \subseteq \mathbb{R}^k$ be the solution set of
$$a_{i_1}(x_1) \wedge a_{i_1,i_2}(x_1,x_2) \wedge \ldots \wedge a_{i_1,\ldots,i_k}(x_1,\ldots,x_k) \tag{2.2}$$
(a) For each $1 \le i_{k+1} \le m_I$,
$$a_{i_1,\ldots,i_k,i_{k+1}}(x_1,\ldots,x_k,x_{k+1})$$
is *true* or
$$x_{k+1} = r(x_1,\ldots,x_k) \tag{2.3}$$

and $r \in Alg_{C_I}$, or

$$r_1(x_1,\ldots,x_k)\rho_1 x_{k+1}\rho_2 r_2(x_1,\ldots,x_k) \tag{2.4}$$

where $r_1 \in Alg_{C_I} \cup \{-\infty\}$, $r_2 \in Alg_{C_I} \cup \{\infty\}$, $r_1 < r_2$ on C_I and $\rho_1, \rho_2 \in \{<, \leq\}$.

(b) If $s_1, s_2 \in Alg_{C_I} \cup \{-\infty, \infty\}$, s_1 appears in

$$a_{i_1,\ldots,i_k,u}(x_1)$$

s_2 appears in

$$a_{i_1,\ldots,i_k,v}(x_1)$$

and $u < v$ then $s_1 \leq s_2$ on C_I.

(c) Let $P_I \subseteq \mathbb{Z}[x_1,\ldots,x_k,x_{k+1}]$ be the set of defining polynomials of all real algebraic functions that appear in formulas a_J for $J = (i_1,\ldots,i_k,i_{k+1})$, $1 \leq i_{k+1} \leq m_I$. Then P_I is delineable over C_I.

Definition 5. Let A be a cylindrical system of algebraic constraints in variables $x_1,\ldots,x_n$. Define

$$b_{i_1,\ldots,i_n}(x_1,\ldots,x_n) \quad := \quad true$$

For $2 \leq k \leq n$, *level k cylindrical algebraic subformulas* given by A are the formulas

$$b_{i_1,\ldots,i_{k-1}}(x_1,\ldots,x_n) := \\ \bigvee_{1\leq i_k \leq m_{i_1,\ldots,i_{k-1}}} a_{i_1,\ldots,i_k}(x_1,\ldots,x_k) \wedge b_{i_1,\ldots,i_k}(x_1,\ldots,x_n)$$

The cylindrical algebraic formula (CAF) given by A is the formula

$$F(x_1,\ldots,x_n) := \bigvee_{1\leq i_1 \leq m} a_{i_1}(x_1) \wedge b_{i_1}(x_1,\ldots,x_n)$$

Remark 6. Let $F(x_1,\ldots,x_n)$ be a CAF given by a cylindrical system of algebraic constraints A. Then

(1) For $1 \leq k \leq n$, $C_{i_1,\ldots,i_k}$ are nonempty connected subsets of $\mathbb{R}^k$.
(2) Sets

$$\{C_{i_1,\ldots,i_n} : 1 \leq i_1 \leq m \wedge 1 \leq i_2 \leq m_{i_1} \wedge \ldots \wedge \\ 1 \leq i_n \leq m_{i_1,\ldots,i_{n-1}}\}$$

form a decomposition of the solution set S_F of F, i.e. they are disjoint and their union is equal to S_F.

Proof. Both parts of the remark follow from the definitions of A and F. $\square$

Let us finish this section with specifications of the two main algorithms we will use for computation with CAFs. The first algorithm is the version of CAD described in [33]. It computes a CAF equivalent to a prenex quantified real polynomial formula.

Algorithm 7. *(CAD)*
Input: A prenex quantified real polynomial formula

$$P(x_1,\ldots,x_{n-k}) = Q_1 x_{n-k+1} \ldots Q_k x_n P_0(x_1,\ldots,x_n)$$

where $P_0(x_1,\ldots,x_n)$ is a quantifier-free real polynomial formula and $k \geq 0$.
Output: A cylindrical algebraic formula $F(x_1,\ldots,x_{n-k})$ equivalent to $P(x_1,\ldots,x_{n-k})$.

The second algorithm, introduced in [34], represents a prenex quantified Boolean combination of CAFs as a CAF in the free variables.

Algorithm 8. *(CAFCombine)*
Input: Cylindrical algebraic formulas

$$F_1(x_1,\ldots,x_n),\ldots,F_m(x_1,\ldots,x_n)$$

a Boolean formula $\Phi(p_1,\ldots,p_m)$ and a sequence of quantifiers $Q_1,\ldots,Q_k$, with $0\leq k\leq n$.
Output: A cylindrical algebraic formula $F(x_1,\ldots,x_{n-k})$ equivalent to

$$Q_1x_{n-k+1}\ldots Q_kx_n\Phi(F_1(x_1,\ldots,x_n),\ldots,F_m(x_1,\ldots,x_n))$$

3. The main algorithm

In this section we describe a recursive algorithm computing CAF representation of solution sets of arbitrary non-prenex quantified real polynomial formulas. Let us first give a formal definition of a quantified real polynomial formula.

Definition 9. The set $QPF(x_1,\ldots,x_n)$ of quantified real polynomial formulas in free variables $x_1,\ldots,x_n$, where $n\geq 0$, is defined recursively as follows.

(1) If $P=f(x_1,\ldots,x_n)\rho 0$, where $f\in\mathbb{R}[x_1,\ldots,x_n]$ and $\rho\in\{<,\leq,\geq,>,=,\neq\}$, then $P\in QPF(x_1,\ldots,x_n)$.
(2) If Φ is a Boolean operator with arity m and $P_1,\ldots,P_m\in QPF(x_1,\ldots,x_n)$, then $\Phi(P_1,\ldots,P_m)\in QPF(x_1,\ldots,x_n)$.
(3) If $Q\in\{\exists,\forall\}$ and $P\in QPF(x_1,\ldots,x_n,y)$, then $Qy\,P\in QPF(x_1,\ldots,x_n)$.

Let $P\in QPF(x_1,\ldots,x_n)$. P is quantifier-free if it does not contain quantifiers. P is prenex if

$$P=Q_1y_1\ldots Q_ky_kP_0$$

$m\geq 0$ and $P_0\in QPF(x_1,\ldots,x_n,y_1,\ldots,y_k)$ is quantifier-free.

The only unary Boolean operators are the negation and the identity. Since $\neg\exists_xP\Leftrightarrow\forall_x\neg P$ and $\neg\forall_xP\Leftrightarrow\exists_x\neg P$, we may assume without loss of generality that a quantified real polynomial formula does not contain the identity operator, double negation, or negation of a formula starting with a quantifier.

Algorithm 10. *(NPCAD)*
Input: A formula $P\in QPF(x_1,\ldots,x_n)$.
Output: A cylindrical algebraic formula $F(x_1,\ldots,x_n)$ equivalent to P.

(1) *If P is prenex return $CAD(P(x_1,\ldots,x_n))$.*
(2) *If*

$$P=Q_1y_1\ldots Q_ky_k\Phi(P_1,\ldots,P_m)$$

with $k\geq 0$ and $m\geq 2$, set $\Psi(p_1,\ldots,p_m):=\Phi(p_1,\ldots,p_m)$. Otherwise

$$P=Q_1y_1\ldots Q_ky_k\neg\Phi(P_1,\ldots,P_m)$$

with $k\geq 0$ and $m\geq 2$. Set $\Psi(p_1,\ldots,p_m):=\neg\Phi(p_1,\ldots,p_m)$.
(3) *For $1\leq i\leq m$, compute*

$$F_i(x_1,\ldots,x_n,y_1,\ldots,y_k):=NPCAD(P_i)$$

(4) *Use* CAFCombine *to compute a CAF $F(x_1,\ldots,x_n)$* equivalent to

$$Q_1y_1\ldots Q_ky_k\Psi(F_1,\ldots,F_m)$$

(5) *Return $F(x_1,\ldots,x_n)$.*

4. Empirical Results

In this section we compare the performance of *NPCAD* and of direct application of *CAD* to the formula transformed to the prenex form. Algorithms *CAD* and *NPCAD* have been implemented in C as a part of the kernel of *Mathematica*. The experiments have been conducted on a 3.0 GHz Intel Core i7 processor, with 6 GB of RAM available for the Linux virtual machine.

Example 11. Prove that for a monic cubic polynomial f, such that f does not have real roots outside $[-3,3]$ and f' has roots in $[-2,-1]$ and in $[1,2]$, the maximal possible value of f at a point in $[-1,1]$ is 26.

Let $f(x) = x^3 + ax^2 + bx + c$. The following formulas express the required conditions on f.

$$\begin{aligned} P_1 &:= \forall_x x^3 + ax^2 + bx + c \Rightarrow -3 \le x \le 3 \\ P_2 &:= \exists_x -2 \le x \le -1 \wedge 3x^2 + 2ax + b = 0 \\ P_3 &:= \exists_x 1 \le x \le 2 \wedge 3x^2 + 2ax + b = 0 \end{aligned}$$

The following formulas state that f attains a value, respectively, equal to and greater than 26 at a point in $[-1,1]$.

$$\begin{aligned} P_4 &:= \exists_x -1 \le x \le 1 \wedge x^3 + ax^2 + bx + c = 26 \\ P_5 &:= \exists_x -1 \le x \le 1 \wedge x^3 + ax^2 + bx + c > 26 \end{aligned}$$

We need to show that

$$\exists_a \exists_b \exists_c P_1 \wedge P_2 \wedge P_3 \wedge P_4$$

is true and

$$\exists_a \exists_b \exists_c P_1 \wedge P_2 \wedge P_3 \wedge P_5$$

is false. *NPCAD* does this in, respectively, 2.91 and 2.84 seconds, hence the total time for solving the problem with *NPCAD* is 5.75 seconds. Using *NPCAD* to find a CAF representation of the solution set of $P_1 \wedge P_2 \wedge P_3 \wedge P_4$ we can show (in 3.76 seconds) that the only polynomial satisfying the required conditions which attains the maximal value of 26 in $[-1,1]$ is

$$f(x) = x^3 - \frac{3}{2}x^2 - 6x + \frac{45}{2}$$

To solve the problem with *CAD* we need to prove that the prenex formula

$$\begin{gathered} \exists_a \exists_b \exists_c \forall_{x_1} \exists_{x_2} \exists_{x_3} \exists_{x_4} (x_1^3 + ax_1^2 + bx_1 + c \Rightarrow -3 \le x_1 \le 3) \wedge \\ -2 \le x_2 \le -1 \wedge 3x_2^2 + 2ax_2 + b = 0 \wedge 1 \le x_3 \le 2 \wedge 3x_3^2 + 2ax_3 + b = 0 \wedge \\ -1 \le x_4 \le 1 \wedge x_4^3 + ax_4^2 + bx_4 + c = 26 \end{gathered}$$

is equivalent to true and the prenex formula

$$\begin{gathered} \exists_a \exists_b \exists_c \forall_{x_1} \exists_{x_2} \exists_{x_3} \exists_{x_4} (x_1^3 + ax_1^2 + bx_1 + c \Rightarrow -3 \le x_1 \le 3) \wedge \\ -2 \le x_2 \le -1 \wedge 3x_2^2 + 2ax_2 + b = 0 \wedge 1 \le x_3 \le 2 \wedge 3x_3^2 + 2ax_3 + b = 0 \wedge \\ -1 \le x_4 \le 1 \wedge x_4^3 + ax_4^2 + bx_4 + c > 26 \end{gathered}$$

is equivalent to false. The computations take, respectively, 385 and 717 seconds, hence the total time for solving the problem with *CAD* is 1102 seconds.

The following experiment we uses a generalization of Example 1, in which we vary the number of quantified subformulas.

TABLE 1. Timings for randomly generated ellipses

n	*CAD*			*NPCAD*			*#true*
	T_a	T_{min}	T_{max}	T_a	T_{min}	T_{max}	
2	9.49	0.93	52.9	4.48	3.55	5.27	8
3	394	3.23	2214	6.65	5.58	7.10	4
4	1779	76.6	14387	10.6	8.18	12.7	5
5	> 3600			14.1	11.4	16.1	1
6	> 3600			15.9	13.6	19.9	0
7	> 3600			21.5	16.9	29.6	0
10	> 3600			28.2	22.9	44.6	0
20	> 3600			50.7	46.2	54.8	0

Example 12. For $n \geq 2$, let $E_1, \ldots, E_n$ be ellipses given by

$$E_i \quad = \quad \{(x,y) : \frac{(x - \frac{c_i}{1000})^2}{4} + (y - \frac{d_i}{1000})^2 < 1\}$$

where for $1 \leq i \leq n$, c_i and d_i are randomly generated integers in $[-10000, 10000]$. Let $s_1, \ldots, s_n$ be randomly generated elements of $\{-1, 0, 1\}$. Decide whether there is a circle with radius 5 containing all E_i for which $s_i = 1$, properly intersecting all E_i for which $s_i = 0$ and disjoint with all E_i for which $s_i = -1$. Solving the problem requires deciding whether the following fully quantified formula is true

$$\exists_a \exists_b \Phi_1 \wedge \ldots \wedge \Phi_n$$

where

$$\Phi_i = \begin{cases} \forall_x \forall_y \frac{(x - \frac{c_i}{1000})^2}{4} + (y - \frac{d_i}{1000})^2 < 1 \Rightarrow (x-a)^2 + (y-b)^2 < 25) & s_i = 1 \\ \exists_x \exists_y \frac{(x - \frac{c_i}{1000})^2}{4} + (y - \frac{d_i}{1000})^2 < 1 \wedge (x-a)^2 + (y-b)^2 = 25 & s_i = 0 \\ \forall_x \forall_y \frac{(x - \frac{c_i}{1000})^2}{4} + (y - \frac{d_i}{1000})^2 < 1 \Rightarrow (x-a)^2 + (y-b)^2 > 25) & s_i = -1 \end{cases}$$

For each value of n we ran the algorithms for 10 randomly generated examples. The results are given in Table 1. In the columns marked T_a, T_{min} and T_{max} we give, respectively, the average, the minimal and the maximal timing for the given n. The timings are given in seconds. The column marked *#true* gives the number of examples for which the answer was *true*. If for at least two of the ten examples the computation did not finish in 3600 seconds, the corresponding columns are marked > 3600.

4.1. **Conclusions.** The experiments show that for computation of CAF representation of the solution set of a non-prenex polynomial formula, it is usually faster to use *NPCAD* rather than convert the input to prenex form and use *CAD*. In fact, *NPCAD* was faster for all examples with more than three quantified subformulas, and was always faster on average.

REFERENCES

[1] H. Anai, K. Yokoyama, "CAD via Numerical Computation with Validated Symbolic Reconstruction, A3L 2005 Proceedings, 2005, 25-30.

[2] C. W. Brown, "Improved Projection for Cylindrical Algebraic Decomposition", J. Symbolic Comp., 32 (2001), 447-465.

[3] C. W. Brown, "Simple CAD Construction and its Applications", J. Symbolic Comp., 31 (2001), 521-547.

[4] C. W. Brown, “An Overview of QEPCAD B: a Tool for Real Quantifier Elimination and Formula Simplification”, J. JSSAC, Vol 10, No. 1 (2003), 13-22.
[5] C. W. Brown, “QEPCAD B - a program for computing with semi-algebraic sets using CADs”, ACM SIGSAM Bulletin, 37 (4), 2003, 97-108.
[6] B. Caviness, J. Johnson (eds.), “Quantifier Elimination and Cylindrical Algebraic Decomposition”, Springer-Verlag 1998.
[7] G. E. Collins, “Quantifier Elimination for the Elementary Theory of Real Closed Fields by Cylindrical Algebraic Decomposition”, Lect. Notes Comput. Sci., 33, 1975, 134-183.
[8] G. E. Collins, “Quantifier Elimination by Cylindrical Algebraic Decomposition - Twenty Years of Progress”, ", in B. Caviness, J. Johnson (eds.), Quantifier Elimination and Cylindrical Algebraic Decomposition, Springer-Verlag 1998, 8-23.
[9] G. E. Collins, “Application of Quantifier Elimination to Solotareff’s Approximation Problem”, Technical Report 95-31, RISC Report Series, University of Linz, Austria, 1995.
[10] G. E. Collins, H. Hong, "Partial Cylindrical Algebraic Decomposition for Quantifier Elimination", J. Symbolic Comp., 12 (1991), 299-328.
[11] G. E. Collins, J. R. Johnson, W. Krandick, “Interval Arithmetic in Cylindrical Algebraic Decomposition”, J. Symbolic Comp., 34 (2002), 145-157.
[12] C. Chen, M. Moreno Maza, B. Xia, L. Yang, “Computing Cylindrical Algebraic Decomposition via Triangular Decomposition”, Proceedings of ISSAC 2009, 95-102.
[13] P. Dorato, W. Yang, C. Abdallah, “Robust Multi-Objective Feedback Design by Quantifier Elimination”, J. Symbolic Comp., 24 (1997), 153-160.
[14] H. Hong, "An Improvement of the Projection Operator in Cylindrical Algebraic Decomposition", Proceedings of ISSAC 1990, 261-264.
[15] H. Hong, “Efficient Method for Analyzing Topology of Plane Real Algebraic Curves”, Proceedings of IMACS-SC 93, Lille, France, 1993.
[16] H. Hong, R. Liska, S. Steinberg, “Testing Stability by Quantifier Elimination”, J. Symbolic Comp., 24 (1997), 161-188.
[17] M. Jirstrand, “Nonlinear Control System Design by Quantifier Elimination”, J. Symbolic Comp., 24 (1997), 137-152.
[18] M. A. G. Jenkins, “Three-stage variable-shift iterations for the solution of polynomial equations with a posteriori error bounds for the zeros”, Ph.D. dissertation, Stanford University, 1969.
[19] J. B. Keiper, D. Withoff, "Numerical Computation in Mathematica", Course Notes, Mathematica Conference 1992.
[20] D. Lazard, “Solving Kaltofen’s Challenge on Zolotarev’s Approximation Problem, Proceedings of ISSAC 2006, 196-203.
[21] S. Łojasiewicz, “Ensembles semi-analytiques”, I.H.E.S. (1964).
[22] R. Loos, V. Weispfenning, "Applying Linear Quantifier Elimination", The Computer Journal, Vol. 36, No. 5, 1993, 450-461.
[23] S. McCallum, "An Improved Projection for Cylindrical Algebraic Decomposition of Three Dimensional Space", J. Symbolic Comp., 5 (1988), 141-161.
[24] S. McCallum, "An Improved Projection for Cylindrical Algebraic Decomposition", in B. Caviness, J. Johnson (eds.), Quantifier Elimination and Cylindrical Algebraic Decomposition, Springer-Verlag 1998, 242-268.
[25] S. McCallum, “On Projection in CAD-Based Quantifier Elimination with Equational Constraint”, In Proceedings of the 1999 International Symposium on Symbolic and Algebraic Computation, ACM Press 1999, 145-149.
[26] S. McCallum, “On Propagation of Equational Constraints in CAD-Based Quantifier Elimination”, In Proceedings of the 2001 International Symposium on Symbolic and Algebraic Computation, ACM Press 2001, 223-230.
[27] A. Strzebonski, “An Algorithm for Systems of Strong Polynomial Inequalities”, The Mathematica Journal, vol. 4, iss. 4 (1994), 74-77.
[28] A. Strzebonski, “Computing in the Field of Complex Algebraic Numbers”, J. Symbolic Comp., 24 (1997), 647-656.
[29] A. Strzebonski, “Algebraic Numbers in Mathematica 3.0”, The Mathematica Journal, vol. 6, iss. 4 (1996), 74-80.
[30] A. Strzebonski, “A Real Polynomial Decision Algorithm Using Arbitrary-Precision Floating Point Arithmetic”, Reliable Computing, Vol. 5, Iss. 3 (1999), 337-346.

[31] A. Strzebonski, “Solving Systems of Strict Polynomial Inequalities”, J. Symbolic Comp., 29 (2000), 471-480.
[32] A. Strzebonski, "Solving Algebraic Inequalities", The Mathematica Journal, Vol. 7, Iss. 4 (2000), 525-541.
[33] A. Strzebonski, “Cylindrical Algebraic Decomposition using validated numerics”, J. of Symbolic Comp. 41 (2006), 1021-1038.
[34] A. Strzebonski, “Computation with Semialgebraic Sets Represented by Cylindrical Algebraic Formulas”, In Proceedings of the 2010 International Symposium on Symbolic and Algebraic Computation, ACM Press 2010, 61-68.
[35] A. Tarski, “A decision method for elementary algebra and geometry”, University of California Press, Berkeley 1951.
[36] S. Wolfram, "The Mathematica Book", 4th. Ed., Wolfram Media/Cambridge University Press, 1999.

WOLFRAM RESEARCH INC., 100 TRADE CENTRE DRIVE, CHAMPAIGN, IL 61820, U.S.A.
E-mail address: adams@wolfram.com