SCHOOL OF OPERATIONS RESEARCH
AND INDUSTRIAL ENGINEERING
COLLEGE OF ENGINEERING
CORNELL UNIVERSITY
ITHACA, NEW YORK 14853

TECHNICAL REPORT NO. 858

August 1989
(Revised July 1991)

ON THE COMPUTATIONAL COMPLEXITY OF APPROXIMATING SOLUTIONS FOR REAL ALGEBRAIC FORMULAE

By

James Renegar

This research was supported by NSF Grant No. DMS-8800835.

1. INTRODUCTION[1]

This paper is concerned with the computational complexity of constructing solutions to a very general class of algebraic problems defined over the real numbers. The class includes many non-linear problems from numerical analysis and mathematical programming. The class is naturally defined in terms of the classical decision problem for the first order theory of the reals.

The decision problem for the first order theory of the reals is the problem of determining if expressions of a certain form are true or false. Although a more general form is allowed, all allowable expressions can be reduced to the form

$$(Q_1 x^{[1]} \in \mathbb{R}^{n_1})(Q_2 x^{[2]} \in \mathbb{R}^{n_2}) \ldots (Q_\omega x^{[\omega]} \in \mathbb{R}^{n_\omega})\ P(x^{[1]},\ldots,x^{[\omega]}) \tag{1.1}$$

where

(i) each Q_k is one of the quantifiers $\exists$ or $\forall$;

(ii) $P(x^{[1]},\ldots,x^{[\omega]})$ is a quantifier free Boolean formula with *atomic predicates* of the form

$$g_i(x^{[1]},\ldots,x^{[\omega]})\ \Delta_i\ 0$$

each $g_i\colon \underset{k=1}{\overset{\omega}{\times}} \mathbb{R}^{n_k} \to \mathbb{R}$ being a real polynomial, and Δ_i being any one of the "standard relations"

$$>,\ \geq,\ =,\ \neq,\ \leq,\ <\ . \tag{1.2}$$

Such an expression is referred to as a *sentence*. Catenating blocks of variables if necessary, it may be

---

[1]Through theorem 1.1, the introduction has much overlap with the introduction of Renegar [19].

assumed that for each k, Q_k and Q_{k+1} are not the same quantifier. Hence, ω-1 is the number of *quantifier alternations.*

As a simple example of a sentence, consider

$$(\exists\ y \in \mathbb{R}^n)[(g_1(y) \geq 0) \wedge .... \wedge (g_m(y)) \geq 0)] \tag{1.3}$$

where $g_1,...,g_m: \mathbb{R}^n \rightarrow \mathbb{R}$ are polynomials. This sentence asserts that the "feasible set" $\{x: g_i(x) \geq 0 \ \ \forall\ i\}$ is non-empty. Depending on the specific coefficients of the polynomials $g_1,...,g_m$, this sentence is either true or false.

A more interesting example is provided by the following sentence in which $f, g_1,...,g_m: \mathbb{R}^n \rightarrow \mathbb{R}$ are assumed to be polynomials:

$$\begin{aligned}(\exists\ y \in \mathbb{R}^n)(\forall\ x \in \mathbb{R}^n)[&(g_1(y) \geq 0) \wedge ... \wedge (g_m(y) \geq 0) \\ &\wedge \\ &[(g_1(x) < 0) \vee ... \vee (g_m(x) < 0) \vee (f(y) - f(x) \leq 0)]].\end{aligned} \tag{1.4}$$

This sentence asserts something about the algebraic non-linear programming problem (NLP)

$$\begin{aligned}&\min \quad f(x) \\ &\text{s.t.} \quad g_i(x) \geq 0 \qquad i = 1,...,m.\end{aligned} \tag{1.5}$$

The sentence asserts that there exists $y \in \mathbb{R}^n$ such that both y is feasible for the NLP and for all $x \in \mathbb{R}^n$, either x is infeasible or the objective function value at y is at least as good as at x. In other words, the sentence asserts that the NLP has an optimal solution y. Depending on the specific coefficients of the polynomials $f, g_1,...,g_m$, this sentence is either true or false.

The collection of all true sentences constitutes the first order theory of the reals, denoted by $\mathrm{Th}(\mathbb{R})$. A decision method for $\mathrm{Th}(\mathbb{R})$ is an algorithm which, given a sentence, determines if the sentence is in $\mathrm{Th}(\mathbb{R})$. Tarski [23] was the first to present a decision method for $\mathrm{Th}(\mathbb{R})$. As regards computational complexity, much better algorithms than his are now known.

A brief survey of results on the complexity of the decision problem can be found in Renegar [19], [22]. Important results have been established by Collins [8], Grigor'ev [12] and Heintz, Roy and Solernó [13] among others.

The sentence (1.1) is said to be in prenex form; all quantifiers occur in front. As already mentioned, sentences are allowed to be of a more general form, but all sentences can be reduced to equivalent sentences in prenex form. The reduction can be accomplished efficiently, as is discussed in the introduction of [19]. In the present paper we focus on sentences in prenex form.

Traditionally, attention has been restricted to sentences for which the coefficients of the polynomials g_i are rational numbers. Consequently, a decision method for $\mathrm{Th}(\mathbb{R})$ is an algorithm in the usual Turing machine sense. However, there is no ambiguity regarding what is meant for a sentence of the form (1.1) to be true or false if we allow the coefficients of the polynomials g_i to be real numbers. Borrowing a phrase from Blum and Smale [4] we will refer to the resulting collection of true sentences as "the extended first order theory of the reals" and denote it by $\mathrm{ETh}(\mathbb{R})$. Thus, we view $\mathrm{Th}(\mathbb{R})$ as the subset of $\mathrm{ETh}(\mathbb{R})$ consisting of those sentences for which all of the polynomials occurring in the atomic predicates have rational coefficients.

An appropriate model of computation for defining what is meant by a "decision method for $\mathrm{ETh}(\mathbb{R})$" is the model developed by Blum, Shub and Smale [3]. This model formalizes and extends what researchers often refer to as "arithmetic complexity." Computations are restricted to the arithmetic operations $+, -, \cdot, \div$, all assumed to be performed exactly on real numbers with no rounding errors (i.e. infinite precision) and branching decisions are made using the comparison

operations $>$ and $=$. (A complete formalization of the model requires developing an appropriate notion of 'uniform algorithm', etc.; these issues are dealt with in [3].)

When speaking of a decision method for $\mathrm{Th}(\mathbb{R})$ in the usual Turing machine sense we will, for brevity, speak of the "bit model" of computation. When speaking of a decision method for $\mathrm{ETh}(\mathbb{R})$ as an algorithm in the arithmetic complexity sense, we will speak of the "real number model" of computation. Some discussion of the significance of the real number model of computation as regards the decision problem is presented in [19].

In defining the sentence (1.1) we merely required that P be a "quantifier free Boolean formula." Now we state more precisely the form we will be assuming P to have.

Given an arbitrary Boolean function $\mathbb{P}: \{0,1\}^m \to \{0,1\}$ and given m atomic predicates $g_i(x)\Delta_i 0$ there is an obvious and natural way to define a 0-1 valued function $P(x)$, namely

$$P(x) := \mathbb{P}(B_1(x),...,B_m(x))$$

where

$$B_i(x) := \begin{cases} 1 \text{ if } g_i(x)\Delta_i 0 \\ 0 \text{ otherwise.} \end{cases}$$

The perspective we take in this paper is that $\mathbb{P}$ is given, and the function P appearing in (1.1) is then defined as above.

In some way a measure of the cost of evaluating the Boolean function $\mathbb{P}$ must enter into the cost of a decision method. Traditionally, $\mathbb{P}$ has been assumed to be of restricted forms. Rather than requiring $\mathbb{P}$ to be of a restricted form, we assume that a procedure (i.e. oracle) is available for evaluating $\mathbb{P}$ when arbitrary values 0 or 1 are substituted for the variables B_i. A component of the bounds we state will be the number of "calls to $\mathbb{P}$", meaning the number of times the procedure for evaluating $\mathbb{P}$ is used. Of course we could restrict $\mathbb{P}$ to be of a specific form, but doing so would reduce the versatility of our results.

When stating time bounds for parallel computation, we will use $\mathrm{Time}(\mathbb{P}, N)$ to denote the worst-case time (over all 0-1 vectors) required to compute $\mathbb{P}$ using N processors.

When we refer to "operations" it will be in the context of $\mathrm{ETh}(\mathbb{R})$. Formally, for the sequential operation bounds that follow, "operations" can be taken to refer to those allowed in the real number model of computation developed by Blum, Shub and Smale [3]. For readers unfamiliar with that paper, "operations" can simply be taken to refer to the ordered field operations $+, -, \cdot, \div, >$ and $=$ (and operations for storing and retrieving data). Although a model for parallel computation over the reals is not formalized in [3], the uniform and elementary nature of the algorithms designed for proving the "real number model parallel bounds" that follow guarantee that the bounds will hold for any reasonable real number model of parallel computation.

When we refer to "bit operations" it will be in the context of $\mathrm{Th}(\mathbb{R})$ and will refer to Turing machine operations. As with the real number model algorithms, the uniform and elementary nature of the algorithms designed for proving the "bit model parallel bounds" that follow guarantee that the bounds will hold for any reasonable bit model of parallel computation, of which there are several (e.g. the circuit model commonly used in defining NC).

In what follows we assume that $P(x^{[1]},\ldots,x^{[\omega]})$ has m atomic predicates and we assume that $d \geq 2$ is an upper bound on the degrees of the polynomials occurring in the atomic predicates. Also recall that n_k is the number of variables occurring in $x^{[k]}$.

When referring to a decision method for $\mathrm{Th}(\mathbb{R})$ we may assume that the coefficients of the polynomials are integers; we then let L denote the maximum, over all of the coefficients, of the number of bits required to specify the coefficient.

The data specifying a sentence is ω, $n_1,\ldots,n_\omega$, $Q_1,\ldots,Q_\omega$, m, $\Delta_1,\ldots,\Delta_m$, d, the coefficients of the polynomials $g_1,\ldots,g_m$, and the Boolean function $\mathbb{P}$.

Theorem 1.1 (Renegar [19], [20]). There is an algorithm for the decision problem for $\mathrm{ETh}(\mathbb{R})$ that requires only

$$(\mathrm{md})^{2^{O(\omega)}\prod_k n_k} \text{ operations and } (\mathrm{md})^{O(\sum_k n_k)} \text{ calls to } \mathbb{P}.$$

The algorithm requires no divisions. The algorithm can be implemented in parallel, requiring time

$$[2^{\omega}\,(\prod_k n_k)\log(\mathrm{md})]^{O(1)} + \mathrm{Time}(\mathbb{P},N)$$

if $(\mathrm{md})^{2^{O(\omega)}\prod_k n_k}$ processors are used for the operations and $N(\mathrm{md})^{O(\sum_k n_k)}$ processors are used for the calls (for any $N \geq 1$).

When restricted to sentences involving only polynomials with integer coefficients, the algorithm becomes a decision method for $\mathrm{Th}(\mathbb{R})$ requiring only

$$L(\log L)(\log\log L)(\mathrm{md})^{2^{O(\omega)}\prod_k n_k}$$

sequential bit operations and $(\mathrm{md})^{O(\sum_k n_k)}$ calls to $\mathbb{P}$. When implemented in parallel the algorithm requires time

$$\log(L)[2^{\omega}(\prod_k n_k)\log(\mathrm{md})]^{O(1)} + \mathrm{Time}(\mathbb{P},N)$$

if $L^2(\mathrm{md})^{2^{O(\omega)}\prod_k n_k}$ processors are used for bit operations and $N(\mathrm{md})^{O(\sum_k n_k)}$ processors are used for the calls (for any $N \geq 1$). □

A similar theorem was established by Heintz, Roy and Solernó [13].

As an application of the theorem, recall the two examples of sentences that we gave. Applying the algorithm of the theorem to the first sentence shows that in the real number model one can

determine if the feasible region of the NLP(1.5) is non-empty with $(md)^{O(n)}$ operations performed in time $[n \log(md)]^{O(1)}$ using $(md)^{O(n)}$ parallel processors, assuming that the degrees of the polynomials are at most d. Applying the algorithm to the second sentence shows that one can determine if the NLP has an optimal solution with $(md)^{O(n^2)}$ operations performed in time $[n \log(md)]^{O(1)}$ using $(md)^{O(n^2)}$ parallel processors. The theorem provides analogous bounds for the bit model assuming that the coefficients of f, $g_1,...,g_m$ are integers.

There are many, many interesting decision problems which can be reduced to the problem of deciding if a particular sentence is true or false. Indeed, the importance of the decision problem for the first order theory of the reals is largely a consequence of its generality and the fact that decision methods for it exist. We list a few more problems, motivated by non-linear mathematical programming, that the reader may find of interest. With a little practice, the reader can undoubtedly construct many others.

In the examples we use "$A \Rightarrow B$" as shorthand for "$(\sim A) \vee (A \wedge B)$," and we use "x is feasible" as shorthand for "$(g_1(x) \geq 0) \wedge ... \wedge (g_m(x) \geq 0)$," i.e., x is feasible for the NLP(1.5).

The following sentence is true if and only if (1.5) has a local optimum:

$$(\exists\, y \in \mathbb{R}^n)(\exists\, \delta \in \mathbb{R})(\forall\, x \in \mathbb{R}^n)[(\delta > 0) \wedge (y \text{ is feasible}) \wedge ([(x \text{ is feasible}) \wedge (\|x-y\|^2 \leq \delta^2)] \Rightarrow (f(x) - f(y) \geq 0))].$$

Here, $\| \|$ denotes the Eucliean norm.

For the next example assume that the objective polynomial f is dependent on an additional parameter ρ. Assume that for the situation of interest this parameter is only known to lie within the range $[0,\bar{\rho}]$. The following question is then natural: given $\delta > 0$, does there exist $y \in \mathbb{R}^n$ such that if ρ is fixed at any value in the interval $[0,\bar{\rho}]$, then the resulting NLP(1.5) has an optimal solution within distance δ of y? The answer is yes if and only if the following sentence is true:

$$(\exists\ y\ \in\mathbb{R}^n)(\forall \rho\ \in\mathbb{R})(\exists\ x\ \in\mathbb{R}^n)(\forall\ z\ \in\mathbb{R}^n)[(\rho > 0) \wedge (\rho \leq \bar{\rho})]$$

$$\Rightarrow$$

$$[(x \text{ is feasible}) \wedge (\|y\text{-}x\|^2 - \delta^2 \leq 0)$$

$$\wedge$$

$$[(z \text{ is feasible}) \Rightarrow (f(\rho,z) - f(\rho,x) \geq 0)]].$$

Similarly, one can construct interesting sentences when the constraint polynomials depend on additional parameters. One can construct a sentence to determine if there is a non-degenerate optimal solution. One can construct sentences to determine which constraints are redundant. One can construct sentences to determine if the feasible set is convex and if the objective and constraint polynomials are convex. One can also do all of these things if the functions involved are only piecewise polynomial, assuming that the underlying decomposition of the domain space is defined via polynomial equalities and inequalities. With only a little thought, one can also do the same things for rational functions (i.e., quotients of polynomials). By introducing additional variables and atomic predicates, one can also introduce radicals into the sentences, e.g., a new variable y and an atomic predicate requiring that $y^2 - x = 0$ allows y to be used as the square root of x. (Of course adding new variables can be disastrous in terms of the complexity bounds provided by theorem 1.1.) Finally, we remark that deciding if an algebraic "min-max" problem has a solution can often be easily recast into deciding if a particular sentence is true or false. Etc.

In the above examples we have been primarily concerned with determining if a solution of an algebraic problem exists. (Is there a solution to the feasibility constraints of the non-linear programming problem? Does the NLP have an optimal solution?) The present paper is concerned with the computational complexity of approximating such solutions when they do exist.

To be precise, we first introduce the definition of a *formula.* A formula is defined exactly as a sentence is, except that in a formula not all variables are required to be quantified. The variables that

are not quantified are referred to as the *free* variables; when specific values are substituted for the free variables, the formula becomes a sentence.

Consider a formula

$$(Q_1 x^{[1]} \in \mathbb{R}^{n_1}) \ldots (Q_\omega x^{[\omega]} \in \mathbb{R}^{n_\omega})\ P(y, x^{[1]}, \ldots, x^{[\omega]}) \tag{1.6}$$

with free variables $y = (y_1, \ldots, y_\ell)$. We say that $\bar{y} \in \mathbb{R}^\ell$ is a *solution* for the formula if the sentence obtained by substituting $\bar{y}$ into the formula is true. We say that $\hat{y} \in \mathbb{R}^\ell$ is an *ϵ-approximate solution* for the formula if there exists a solution $\bar{y}$ for the formula such that $||\hat{y}-\bar{y}|| \leq \epsilon$, where the norm is the Euclidean norm on $\mathbb{R}^\ell$. This paper is concerned with the computational complexity of constructing ϵ-approximate solutions.

As an example, a point $\bar{y}$ is an ϵ-approximate solution for the quantifier free formula

$$[(g_1(y) \leq 0) \wedge \ldots \wedge (g_m(y) \leq 0)]$$

if and only if it is within distance ϵ of a feasible point for the algebraic NLP(1.5). Similarly, a point $\bar{y}$ is an ϵ-approximate solution for the formula

$$(\forall\ x \in \mathbb{R}^n)[(g_1(y) \geq 0) \wedge \ldots \wedge (g_m(y) \geq 0) \wedge [(g_1(x) < 0) \vee \ldots \vee (g_m(x) < 0) \vee (f(y) - f(x) < 0)]]$$

if and only if it is within distance ϵ of an optimal solution of the NLP.

Given a formula (1.6) and $r \geq 0$, define SOLUTIONS(r) to be the set of all solutions $\bar{y}$ satisfying $||\bar{y}|| \leq r$. The following theorem is our main result.

Theorem 1.2. There are algorithms which, given $0 < \epsilon < r$ and a formula (1.6), construct a set $\{y^{(i)}\}_i$ of $(md)^{2^{O(\omega)}\ell\prod_k n_k}$ distinct ϵ-approximate solutions with the property that for each connected component of SOLUTIONS(r), at least one of the points $y^{(i)}$ is within distance ϵ of the component.

One such real number model algorithm requires

$$(md)^{2^{O(\omega)}\ell\prod_k n_k} \log\log(3 + \tfrac{r}{\epsilon})$$

sequential operations and $(md)^{2^{O(\omega)}\ell\prod_k n_k}$ calls to $\mathbb{P}$.

Another such real number model algorithm requires

$$(md)^{2^{O(\omega)}\ell\prod_k n_k} \log(1 + \tfrac{r}{\epsilon})$$

operations (no divisions) and $(md)^{2^{O(\omega)}\ell\prod_k n_k}$ calls to $\mathbb{P}$; this algorithm is significant because it can be implemented in parallel, requiring time

$$[2^{\omega}\ell(\prod_k n_k)\log(md)]^{O(1)}\log(1 + \tfrac{r}{\epsilon}) + \mathrm{Time}(\mathbb{P}, N)$$

if $(md)^{2^{O(\omega)}\ell\prod_k n_k}$ processors are used for operations and $N(md)^{2^{O(\omega)}\ell\prod_k n_k}$ processors are used for calls (for any $N \geq 1$).

Assuming $0 < \epsilon < r$ are integral powers of 2, there is such a bit model algorithm which, when implemented in parallel, requires time

$$[2^{\omega}\ell(\prod_k n_k)\log(md\ L + |\log(\epsilon)| + |\log(r)|)]^{O(1)} + \mathrm{Time}(\mathbb{P}, N)$$

if $(L + |\log(\epsilon)| + |\log(r)|)^{O(1)}(md)^{2^{O(\omega)}\ell\prod_k n_k}$ processors are used for operations and

$N(md)^{2^{O(\omega)}\ell\prod_k n_k}$ processors are used for calls (for any $N \geq 1$). The ϵ-approximate solutions constructed will then have rational coordinates with numerators and denominators bounded in bit length by $O(\log(\ell) + |\log(\epsilon)| + |\log(r)|)$. □

Of course the theorem provides upper bounds on the computational complexity of approximating solutions to the various problems already discussed. We leave determination of the bounds implied to the reader.

The bit model algorithm of the theorem relies heavily on the recent algorithm of Neff [16] for approximating all roots of univariate polynomials. Neff resolved positively the longstanding open problem of whether approximating all roots can be done quickly in parallel.

Neff [16] dealt specifically with bit complexity. However, it seems that slight modifications of his ideas lead to an efficient parallel real number model algorithm for approximating roots of univariate polynomials; namely, given $0 < \epsilon < r$, it seems that ϵ-approximations to all roots within distance r of the origin can be obtained in time $\log(d + \log(1 + \frac{r}{\epsilon}))^{O(1)}$ using $(d + \log(1 + \frac{r}{\epsilon}))^{O(1)}$ parallel processors, where d is the degree of the polynomial. If this is indeed true, then theorem 3.2 implies a corresponding parallel time bound for approximating solutions of general formulae; namely, time

$$[2^{\omega}\ell(\prod_k n_k)\log(md + \log(1 + \tfrac{r}{\epsilon}))]^{O(1)} + \text{Time}(\mathbb{P}, N)$$

if $(md)^{2^{O(\omega)}\ell \prod_k n_k (\log(2 + \frac{r}{\epsilon}))^{O(1)}}$ processors are used for operations and $N(md)^{2^{O(\omega)}\prod_k n_k}$ processors are used for calls (for any $N \geq 1$). Since a rigorous extension of Neff's algorithm to the real number model computation has not been written, we cannot claim this bound for general formulae to be proven. Moreover, even if true, contrasting this parallel time bound with the sequential time bound for the first algorithm in the theorem leads to the open question of whether

there exists a real number algorithm achieving the same parallel time bound using only $(md)^{2^{O(\omega)}\prod_k n_k}$ processors for operations and $(md)^{2^{O(\omega)}\prod_k n_k}\,N$ processors for calls.

In terms of ϵ and r alone, the bound provided by the first algorithm of the theorem is optimal for a very general model of computation. More precisely, the following lower bound is known. Let $\mathcal{A}$ denote an algorithm which, given any $s \in [0,r^2]$, constructs a value within distance ϵ of $\sqrt{s}$. In section 2 of Renegar [17] it is proven that if $\mathcal{A}$ is an algorithm in terms of a very general model of computation which allows the operations $+, -, \cdot, \div, >$ and $=$, then the following is true: there exists $s \in [0,r^2]$ such that when algorithm $\mathcal{A}$ is applied to s, it will require at least $C \log\log(3 + \frac{r}{\epsilon})$ operations, where $C > 0$ is independent of $\mathcal{A}$, r and ϵ. The optimality claim follows.

Theorem 1.2 is somewhat unsatisfactory in that r is given apriori. It would be nice to also have an upper bound on the computational complexity of obtaining a single ϵ-approximate solution when only ϵ and the formula (1.6) are input, and not r. The previously mentioned lower bound of [17] implies that an additional parameter must occur in any such upper bound. More specifically, defining

$$r(\epsilon) = \inf\{r;\ r > \epsilon \text{ and } \mathrm{SOLUTIONS}(r) \neq \emptyset\},$$

the lower bound result implies that in terms of ϵ and $r(\epsilon)$ alone, the best upper bound possible (for a very general model of computation) would grow like $\log\log(3 + \frac{r(\epsilon)}{\epsilon})$.

If we could design an algorithm which, given $\epsilon > 0$ and a formula (1.6) for which the solution set is non-empty (a condition that can be efficiently verified using the algorithm of theorem 1.1), efficiently constructs a good upper bound $\bar{r}(\epsilon)$ to $r(\epsilon)$, then combined with the algorithms of theorem 1.2, we would have methods for efficiently constructing an ϵ-approximate solution where the only input to the methods would be ϵ and the formula. However, it is easy to design an algorithm for determining a good upper bound $\bar{r}(\epsilon)$ to $r(\epsilon)$ using theorem 1.1.

First check if SOLUTIONS(ϵ) is non-empty. If it is non-empty, let $r(\epsilon) := \epsilon$. Otherwise, replace ϵ with 2ϵ and try again. Assuming that on the i^{th} iteration it is determined that SOLUTIONS($s_i\epsilon$) = $\emptyset$ for a specific number s_i, replace s_i with $s_{i+1} = (s_i)^2$ and try again. Terminate with the first value of s_i thus obtained for which SOLUTIONS($s_i\epsilon$) $\neq \emptyset$ and define $\bar{r}(\epsilon) := s_i\epsilon$.

Combining this procedure with the first algorithm of theorem 1.2 yields a method for constructing an ϵ-approximate solution with operation count, in terms of ϵ and $r(\epsilon)$ alone, growing only like $\log\log(3 + \frac{r(\epsilon)}{\epsilon})$. By the previous remarks, this is optimal in terms of ϵ and $r(\epsilon)$ alone (for a very general model of computation), among all algorithms depending only on input ϵ and the formula (1.6).

I wish I knew how to design an efficient real number model algorithm for determining a relatively sharp upper bound on the infimum of those values r for which <u>every</u> connected component of the solution set intersects $\{y; ||y|| \leq r\}$. For the bit model of computation we do have the following proposition, which is established in section 3.

Given a formula (1.6), let SOLUTIONS denote the set of its solutions.

<u>Proposition 1.3</u>. If formula (1.6) has only integer coefficients, each of bit length at most L, then every connected component of SOLUTIONS intersects $\{y; ||y|| \leq r\}$ where r satisfies $\log(r) = L(md)^{2^{O(\omega)}\ell\prod_k n_k}$. □

A similar, but weaker, bound can be found in Vorobjov [24].

Our proofs are not lengthy because the bulk of the mathematics needed to establish them has already been developed in Renegar ([19], [20] and [21]). In section 2 we collect the propositions from those papers that we will rely on. In section 3 we reduce the problem of designing algorithms to establish theorem 1.2 to the problem of designing efficient algorithms for approximating zeros of real

univariate polynomials; the results in section 3 are phrased to be applicable to any univariate polynomial zero approximation algorithm. In section 4 we recall some known facts regarding the computational complexity of approximating the real zeros of real univariate polynomials. The results of sections 3 and 4 together give the theorems.

Several researchers have considered the problem of obtaining worst-case computational complexity bounds for approximating solutions of systems of polynomial equations, including Lazard [15], Chistov and Grigor'ev [7], Renegar [18] and Canny [6]. Grigor'ev and Vorobjov [11] considered the problem of approximating solutions of real polynomial inequalities. (Except for [18], the analyses and algorithms in these papers rely on structure provided by the bit model of computation that is not available in the real number model.) When specialized to systems of polynomials, Theorem 1.2 provides at least as good of bounds as those obtained by all of these researchers, except for the fact that the constants in the exponent are unspecified.

An understanding of the decision methods of Collins [8], Grigor'ev [12], and Heintz, Roy and Solernó [13] lead to algorithms for approximating solutions of formulae, similar to the way in which the algorithms in the present paper are developed from [19], [20] and [21]. (Both Collins and Grigor'ev deal only with bit complexity, and do not present efficient parallel decision methods). However, in the same ways that the complexity bounds in [19] are superior to those found in these other works (see [19] or [22] for a comparison), the resulting bounds for approximating solutions are also superior.

Finally, as will become obvious to anyone who proceeds, this is strictly a theoretical work. Although the ideas underlying the algorithms may someday lead to "practical" algorithms, the algorithms herein are constructed solely as means to proving the theorems.

2. PRELIMINARIES

In this section we introduce definitions and record several previously established propositions.

The notation used in this and subsequent sections may strike the reader as odd; it has been chosen to conform with the notation of [19], [20] and [21] where the reader is referred for many of the proofs.

Whenever we speak of "constructing" something, we mean that there is a real number model algorithm for doing so. Each of the algorithms in the following propositions and lemmas yield bit model algorithms when restricted to integer inputs, assuming that the underlying operations are carried out "bit by bit".

Let $h_1,...,h_{\mathcal{M}}: \mathbb{R}^\ell \to \mathbb{R}$ be arbitrary polynomials of degree at most $\mathcal{D}$. We use $\{h_i\}_i$ to denote the set of these polynomials. A vector $\sigma \in \{-1, 0, 1\}^{\mathcal{M}}$ is said to be a "consistent sign vector" for $\{h_i\}_i$ if there exists $\bar{y} \in \mathbb{R}^\ell$ such that the sign of $h_i(\bar{y})$ is σ_i for all i. The "sign vector of $\{h_i\}_i$ at $\bar{y}$" is the vector in $\{-1, 0, 1\}^{\mathcal{M}}$ whose i^{th} coordinate has the same sign as $h_i(\bar{y})$.

The following proposition is a restatement of proposition 4.1 from [19].

Proposition 2.1. Any set $\{h_i\}_i$ of $\mathcal{M}$ polynomials $h_i: \mathbb{R}^\ell \to \mathbb{R}$, of degree at most $\mathcal{D} \geq 2$, has at most $(\mathcal{M}\mathcal{D})^{O(\ell)}$ consistent sign vectors. The entire set of consistent sign vectors can be constructed from the coefficients of $\{h_i\}_i$ with $(\mathcal{M}\mathcal{D})^{O(\ell)}$ operations (no divisions) performed in time $[\ell \log(\mathcal{M}\mathcal{D})]^{O(1)}$ using $(\mathcal{M}\mathcal{D})^{O(\ell)}$ parallel processors. If the coefficients of $\{h_i\}_i$ are integers of bit length at most L, the construction can be accomplished with $L(\log L)(\log\log L)(\mathcal{M}\mathcal{D})^{O(\ell)}$ sequential bit operations, or in time $(\log L)[\ell \log(\mathcal{M}\mathcal{D})]^{O(1)}$ using $L^2(\mathcal{M}\mathcal{D})^{O(\ell)}$ parallel processors. □

The "connected sign partition" $CSP\{h_i\}_i$ generated by a finite set $\{h_i\}_i$ of polynomials $h_i: \mathbb{R}^\ell \to \mathbb{R}$ is the partition of $\mathbb{R}^\ell$ whose elements are the maximal connected subsets with the

following property: if $\bar{y}$ and $\hat{y}$ are in the same element then the sign of $h_i(\bar{y})$ is the same as the sign $h_i(\hat{y})$ for all i.

The following proposition is a restatement of proposition 6.2.2 from [20]. The polynomials $\{g_i\}_i$ occurring in the proposition are assumed to be those occurring in the formula (1.6).

Proposition 2.2. Given a formula (1.6), there exists a set $\{h_i\}_i$ of $(md)^{2^{O(\omega)}\prod_k n_k}$ polynomials $h_i\colon \mathbb{R}^\ell \to \mathbb{R}$, of degree at most $(md)^{2^{O(\omega)}\prod_k n_k}$, with the property that if $\bar{y}$ and $\hat{y}$ are in the same element of $CSP\{h_i\}_i$ then $\bar{y} \in$ SOLUTIONS if and only if $\hat{y} \in$ SOLUTIONS. The set $\{h_i\}_i$ can be constructed from the coefficients of $\{g_i\}_i$ with $(md)^{2^{O(\omega)}\ell\prod_k n_k}$ operations (no divisions) in time $[2^\omega \ell(\prod_k n_k)\log(md)]^{O(1)}$ using $(md)^{2^{O(\omega)}\ell\prod_k n_k}$ parallel processors. If the coefficients of $\{g_i\}_i$ are integers of bit length at most L, then the construction can be accomplished with $L(\log L)(\log\log L)(md)^{2^{O(\omega)}\ell\prod_k n_k}$ sequential bit operations, or in time $(\log L)[2^\omega \ell(\prod_k n_k)\log(md)]^{O(1)}$ if $L^2(md)^{2^{O(\omega)}\ell\prod_k n_k}$ parallel processors are used; moreover, the coefficients of $\{h_i\}_i$ will then be integers of bit length at most $(L+\ell)(md)^{2^{O(\omega)}\prod_k n_k}$. □

For $\xi, U \in \mathbb{C}^{\ell+1}$ define $\xi \cdot U = \sum_j \xi_j U_j$. If ξ satisfies $\xi_{\ell+1} \neq 0$, define

$$\mathrm{Aff}(\xi) := \frac{1}{\xi_{\ell+1}}(\xi_1,\ldots,\xi_\ell) \in \mathbb{C}^\ell,$$

the "affine image" of ξ. Let

$$e_{\ell+1} := (0,\ldots,0,1) \in \mathbb{R}^{\ell+1}.$$

For a polynomial $R\colon \mathbb{R}^{\ell+1} \to \mathbb{R}$ in the variables $U_1,\ldots,U_{\ell+1}$, define

$$\nabla R := \left(\frac{\partial R}{\partial U_1},\ldots,\frac{\partial R}{\partial U_{\ell+1}}\right).$$

The following proposition is a partial restatement of proposition 3.8.1 from [19].

Proposition 2.3. Assume that $h_1,...,h_{\mathcal{M}}: \mathbb{R}^{\ell} \to \mathbb{R}$ are polynomials of degree at most $\mathcal{D} \geq 2$. There exists a set $\mathcal{R}\{h_i\}_i$ of $(\mathcal{M}\mathcal{D})^{O(\ell)}$ polynomials $R: \mathbb{R}^{\ell+1} \to \mathbb{R}$ of degree at most $D = (\mathcal{M}\mathcal{D})^{O(\ell)}$ with the following properties:

(i) for each element of $CSP\{h_i\}_i$ there exists $R \in \mathcal{R}\{h_i\}_i$ such that R is not identically zero and factors linearly over the complex numbers $R(U) = \prod_i \xi^{(i)} \cdot U$ where for some i, $Aff(\xi^{(i)})$ is well-defined and is in the element;

(ii) for each $\beta \in \mathbb{R}^{\ell+1}$ the entire set of univariate polynomials

$$t \mapsto R(\beta + te_{\ell+1})$$

$$t \mapsto \frac{d^j}{dt^j} \nabla R(\beta + te_{\ell+1}) \qquad j = 0,...,D$$

obtained from all $R \in \mathcal{R}\{h_i\}_i$ can be constructed from β and the coefficients of $\{h_i\}_i$ with $(\mathcal{M}\mathcal{D})^{O(\ell)}$ operations (no divisions) in time $[\ell \log(\mathcal{M}\mathcal{D})]^{O(1)}$ using $(\mathcal{M}\mathcal{D})^{O(\ell)}$ parallel processors; if the coefficients of $\{h_i\}_i$ and β are integers of bit length at most L, then all numbers occurring during the construction will be integers of bit length at most $L(\mathcal{M}\mathcal{D})^{O(\ell)}$. □

The significance of the bound on the bit length of the integers occurring during the construction is that bit operation bounds are easily deduced from it and the real number model operation bounds of the proposition; this is made especially easy because the construction avoids divisions. For example, because two integers of bit length at most L can be multiplied in sequential time $O(L(\log L)(\log\log L))$, the proposition gives an overall sequential bit operation bound of $L(\log L)(\log\log L)(\mathcal{M}\mathcal{D})^{O(\ell)}$. Similarly, a parallel time bound for the bit model is easily deduced from the fact that two integers of bit length at most L can be multiplied in time $O(\log L)$ using $O(L^2)$ parallel processors.

Define

$$\mathfrak{B}(\ell+1,\, D) := \{(i^{\ell-1},\, i^{\ell-2},\ldots,\, i,\, 0);\ i \in \mathbb{Z} \text{ and } 0 \le i \le \ell D^2\}.$$

Thus, $\mathfrak{B}(\ell+1,\, D) \subset \mathbb{R}^{\ell+1}$.

The next proposition is a partial restatement of proposition 2.3.1 from [19].

Proposition 2.4. Given any real polynomial $R\colon \mathbb{R}^{\ell+1} \to \mathbb{R}$ of degree at most D that is not identically zero and factors linearly over the complex numbers $R(U) = \prod_i \xi^{(i)} \cdot U$, the following is true: for each $\xi^{(i)}$ for which $\mathrm{Aff}(\xi^{(i)})$ is well-defined and real there exist $\beta \in \mathfrak{B}(\ell+1,\, D)$ and $0 \le k \le D$ such that the univariate polynomial $t \mapsto R(\beta + te_{\ell+1})$ is not identically zero and for some real zero $\bar{t}$ of $t \mapsto R(\beta + te_{\ell+1})$, the vector

$$\bar{\xi} := \frac{d^k}{dt^k} \nabla R(\beta + \bar{t}e_{\ell+1})$$

satisfies $\mathrm{Aff}(\bar{\xi}) = \mathrm{Aff}(\xi^{(i)})$. □

The following easily proven proposition is a restatement of proposition 4.1.1 of [20]. The importance of this proposition has been recognized by others (e.g. see Coste and Roy [10]).

Proposition 2.5 (Thom's lemma). Assume that $p \not\equiv 0$ is a real univariate polynomial of degree d. If $t', t'' \in \mathbb{R}$ are such that $t' < t''$ and for some $0 \le i < d$ there is a real zero of the ith derivative $p^{(i)}$ contained in the interval $[t', t'']$, then for some $i \le j < d$ the sign of $p^{(j)}(t')$ differs from the sign of $p^{(j)}(t'')$. □

As a simple consequence of the proposition, note that if $p \not\equiv 0$, $t' \ne t''$, $p(t') = p(t'') = 0$, then the sign vector of $\{p^{(j)}\}_{j=0}^{d-1}$ at t' differs from that at t''. Hence, the sign vectors of $\{p^{(j)}\}_{j=0}^{d-1}$ at the real zeros of p serve as representatives of the zeros; the sign vectors distinguish the zeros from one another.

Let $p(t) = \sum_{i=0}^{d} a_i t^i$, $q(t) = \sum_{i=0}^{e} b_i t^i$ be univariate polynomials of degrees at most d and e, respectively. The "Sylvester resultant" of p and q is the determinant of the $(d+e)\times(d+e)$ "Sylvester matrix" $[m_{ij}]$ defined by

$$m_{ij} := \begin{cases} a_{d+j-i} & \text{if } j \leq e\text{-}k \\ b_{j-i} & \text{if } j > e\text{-}k. \end{cases}$$

An extremely well-known and classical result states that if the degrees of p and q are exactly d and e, then the Sylvester resultant of p and q is zero if and only if p and q have a common zero (among the complex numbers). A proof of this is provided by lemma 3.1 of [20].

Another well known fact that we will rely on is that interpolation of a univariate polynomial $p(t) = \sum_{i=0}^{d} a_i t^i$ can be accomplished quickly in parallel. A proof of the following easy lemma can be found in appendix B of [19].

Lemma 2.6. Assume that $p: \mathbb{C} \to \mathbb{C}$ is a polynomial of degree at most $d \geq 2$. A positive multiple of p can be computed solely from the values $p(\bar{t})$, $\bar{t} \in \{0,1,...,d\}$, using $d^{O(1)}$ operations (no divisions). The computations can be implemented in parallel, requiring time $[\log(d)]^{O(1)}$ if $d^{O(1)}$ processors are used. If the values $p(\bar{t})$, $\bar{t} \in \{0,1,...,d\}$, are all integers of bit length at most L, all numbers occurring during the computations will be integers of bit length at most $L+d^{O(1)}$. □

Yet another well-known fact that we will use is that the determinant of a matrix can be computed quickly in parallel. The algorithm underlying the following proposition is constructed by slightly extending ideas of Csanky [9] to avoid divisions. A proof of the proposition can be found in appendix A of [19].

Proposition 2.7 (Csanky [9], Berkowitz [1]). There exists an algorithm which, given any $n \geq 1$ and any complex $n\times n$ matrix A, computes $n!\det(A)$ without divisions in time $O(\log^2(n))$ using $n^{O(1)}$ parallel processors. If the coefficients of A are integers of bit length at most L, all numbers occurring during the computation will be integers of bit length at most $Ln^{O(1)}$. □

Propositions similar to the following proposition are well-known (e.g. Borodin, von zur Gathen and Hopcroft [2]).

Proposition 2.8. Suppose that p_1, p_2 and p_3 are real univariate polynomials of degree at most d. Let p denote the greatest common divisor of $\{p_1, p_2, p_3\}$. (Of course p is unique up to a constant multiple.) Then we can efficiently construct real polynomials $\bar{p}_1$, $\bar{p}_2$ and $\bar{p}_3$ for which there exists a common constant $c \neq 0$ satisfying $cp_i = p\bar{p}_i$ for all i. By "efficiently construct" we mean that the polynomials $\bar{p}_i$ can be constructed with $d^{O(1)}$ operations (no divisions) in time $[\log(d)]^{O(1)}$ using $d^{O(1)}$ parallel processors. If the coefficients of p_1, p_2 and p_3 are integers of bit length at most L, then all numbers occurring during the construction will be integers of bit length at most $Ld^{O(1)}$.

Proof. Begin by computing p. It is well-known that this can be accomplished in parallel time $[\log(d)]^{O(1)}$ using $d^{O(1)}$ processors (e.g. by relying on Brown and Traub [5] and Csanky [9]). A complete proof of this is provided just following proposition 8.2 in [21]. The proof there shows that this can be accomplished without divisions by relying on proposition 2.7 above. Moreover, the proof shows that if p_1, p_2 and p_3 have integer coefficients of bit length at most L, then the constructed polynomial p will have integer coefficients of bit length at most $Ld^{O(1)}$.

To construct $\bar{p}_i$, consider the linear equations corresponding to the identity $p_i = p\hat{p}_i$, viewing the coefficients of $\hat{p}_i$ as variables. Use the algorithm of proposition 2.7 to efficiently compute (multiples of) the numerator and denominator determinants arising from Cramer's rule. Multiply the quotients by the product of the three denominator determinants (for $i = 1$, 2 and 3) to obtain $\bar{p}_i$. □

We close this section with a well-known and easily proven lemma that will be relied upon in establishing proposition 1.3.

Lemma 2.9. Suppose that $p(t) = \sum_{i=0}^{d} a_i t^i$ is a univariate polynomial, where $a_d \neq 0$. If $p(\bar{t}) = 0$, then $|\bar{t}| \leq 1 + \max_{i<d} \left| \frac{a_i}{a_d} \right|$.

Proof. We may assume that $|\bar{t}| > 1$. Clearly,

$$|\bar{t}|^d \leq \sum_{i=0}^{d-1} \left| \frac{a_i}{a_d} \right| |\bar{t}|^i$$

$$\leq \max_{i<d} \left| \frac{a_i}{a_d} \right| \sum_{i=0}^{d-1} |\bar{t}|^i$$

$$< \max_{i<d} \left| \frac{a_i}{a_d} \right| \frac{|\bar{t}|^d}{|\bar{t}| - 1} .$$

The lemma follows. □

3. REDUCTION TO UNIVARIATE POLYNOMIAL ZERO APPROXIMATION

In this section we show how to reduce the problem of constructing algorithms for establishing theorem 1.2 to the problem of constructing efficient algorithms for approximating zeros of univariate polynomials. The highlight of this section is a theorem which allows one to deduce operation and time bounds on the cost of obtaining ϵ-approximate solutions from operation and time bounds for univariate polynomial zero approximation algorithms.

Before stating and proving the theorem of this section we present a proposition that will be used in the proof. This proposition regards the computational complexity of approximating the real factors of multi-variate polynomials that are known to factor linearly over the complex numbers.

We assume that a real number model procedure for obtaining approximations to the real zeros of real univariate polynomials is available. We treat the procedure as an oracle. Letting P_d denote the set of non-constant real univariate polynomials of degree at most d, we let $Cost(d,r,\epsilon)$ denote the worst-case (over P_d) number of sequential operations required by the procedure to construct a set of $d^{O(1)}$ points that contains ϵ-approximations to all of those real zeros $\bar{x}$ satisfying $|\bar{x}| \leq r$. (We don't require that each of the $d^{O(1)}$ points be an ϵ-approximation to a zero.) We let $Time(d,r,\epsilon,N)$ denote the worst-case time required by the procedure if it is implemented using N parallel processors. We also assume that we have an analogous bit model procedure. Letting $P_{L,d}$ denote the set of non-constant real univariate polynomials, of degree at most d, whose coefficients are integers of bit length at most L, we let $Cost(L,d,r,\epsilon)$ denote the worst case number of sequential bit operations required by the procedure, and we let $Time(L,d,r,\epsilon,N)$ denote the worst case time required by the procedure if it is implemented using N parallel processors. We assume that this procedure constructs rational points $y^{(i)}$ with numerator and denominator bounded in bit length by $O(|\log(r)| + |\log(\epsilon)|)$.

Recall that $\mathfrak{B}(\ell+1, D) := \{(i^{\ell-1}, i^{\ell-2},..., i, 0);\ i \in \mathbb{Z},\ 0 \leq i \leq \ell D^2\}$.

Proposition 3.1. Assume that $R: \mathbb{R}^{\ell+1} \to \mathbb{R}$ is a (not identically zero) polynomial, of degree at most $D \geq 2$, that factors linearly $R(U) = \prod_i \xi^{(i)} \cdot U$ over the complex numbers. Assume that the coefficients of all of the following pairs of univariate polynomials are available;

$$t \mapsto R(\beta + te_{\ell+1}) \tag{3.1}$$

$$t \mapsto \nabla R(\beta + te_{\ell+1}) \tag{3.2}$$

where β ranges over $\mathfrak{B}(\ell+1, D)$. (Note that we do not assume the coefficients of R are known; we do assume that for each β we know which pair (3.1), (3.2) corresponds to β.)

Given $0 < \epsilon < r$, a set $\{y^{(i)}\}_i \subset \mathbb{R}^\ell$ of $D^{O(1)}$ points satisfying the following property can be efficiently constructed; for each $\xi^{(i)}$ satisfying $\xi^{(i)}_{\ell+1} \neq 0$, $\mathrm{Aff}(\xi^{(i)}) \in \mathbb{R}^\ell$ and $||\mathrm{Aff}(\xi^{(i)})|| \leq r$, there exists $y^{(j)} \in \{y^{(i)}\}_i$ satisfying $||y^{(j)} - \mathrm{Aff}(\xi^{(i)})|| \leq \epsilon$.

By "efficiently constructed" we mean that the set $\{y^{(i)}\}_i$ can be constructed with $(\ell D)^{O(1)}\mathrm{Cost}(D, r, \frac{\epsilon}{\ell})$ sequential operations, or in time $[\log(\ell D)]^{O(1)} + \mathrm{Time}(D, r, \frac{\epsilon}{\ell}, N)$ using $N(\ell D)^{O(1)}$ parallel processors (for any $N \geq 1$). If the coefficients of the polynomials (3.1) and (3.2) are all integers of bit length at most L, it can then be constructed with

$$[\hat{L}(\log \hat{L})\ (\log\log \hat{L}) + \mathrm{Cost}(\bar{L}, D, r, \tfrac{\epsilon}{\ell})](\ell D)^{O(1)}$$

sequential bit operations, where $\hat{L} = L + \log(\ell) + |\log(r)| + |\log(\epsilon)|$ and $\bar{L} = LD^{O(1)}$. It can then be constructed in parallel, requiring time

$$(\log \hat{L})[\log(\ell D)]^{O(1)} + \mathrm{Time}(\bar{L}, D, r, \tfrac{\epsilon}{\ell}, N)$$

if $(\hat{L}^2+N)(\ell D)^{O(1)}$ processors are used (for any $N \geq 1$). The coordinates of the points $y^{(i)}$ will then be rational with numerator and denominator bounded in bit length by $O(\log(\ell) + |\log(r)| + |\log(\epsilon)|)$.

Proof. By proposition 2.4 there exist $0 \leq k \leq D$ and $\beta \in \mathfrak{B}(\ell+1, D)$ which satisfy

$$t \mapsto R(\beta + te_{\ell+1}) \not\equiv 0, \tag{3.3}$$

$$t \mapsto \frac{d^k}{dt^k} \frac{\partial}{\partial U_{\ell+1}} R(\beta + te_{\ell+1}) \not\equiv 0. \tag{3.4}$$

Moreover, such k and β can be efficiently determined from the polynomials (3.1), (3.2). Fix such a pair k, β.

Consider the system of $D + 1$ univariate polynomials

$$t \mapsto \frac{d^w}{dt^w} R(\beta + te_{\ell+1}) \qquad w = 0,...,D\text{-}1$$

$$t \mapsto \frac{d^k}{dt^k} \frac{\partial}{\partial U_{\ell+1}} R(\beta + te_{\ell+1}).$$

Assuming that the coordinates of the consistent sign vectors for this system are indexed from 1 to D+1, define $\mathcal{T}$ as the set consisting of those consistent sign vectors τ for the system that satisfy both of the properties $\tau_1 = 0$, $\tau_{D+1} \neq 0$. Because the coefficients of the polynomials (3.1) and (3.2) are assumed available, proposition 2.1 shows that $\mathcal{T}$ can be constructed efficiently.

Relying on proposition 2.5, there is a natural one-to-one correspondence between $\mathcal{T}$ and the set consisting of those real zeros $\bar{t}$ of $t \mapsto R(\beta + te_{\ell+1})$ for which $\mathrm{Aff}[\frac{d^k}{dt^k} \nabla R(\beta + \bar{t}e_{\ell+1})]$ is well-defined. For $\tau \in \mathcal{T}$, define $t(\tau)$ to be the zero corresponding to τ, and define

$$y(\tau) := \mathrm{Aff}[\frac{d^k}{dt^k} \nabla R(\beta + t(\tau)e_{\ell+1})].$$

We will show how to efficiently construct a finite subset of $\mathbb{R}^\ell$ that contains an ϵ-approximation to every point $y(\tau)$ satisfying $||y(\tau)|| \leq r$. Letting $\{y^{(i)}\}_i$ denote the union of the

subsets thus obtained from all $0 \leq k \leq D$ and $\beta \in \mathfrak{B}(\ell+1, D)$ satisfying (3.3) and (3.4), proposition 2.4 then implies that $\{y^{(i)}\}_i$ satisfies the requirements of the proposition.

We continue to assume that k and β are fixed and that they satisfy (3.3) and (3.4).

For each $i = 1,...,\ell$, consider the Sylvester resultant of the two univariate polynomials

$$t \mapsto R(\beta + te_{\ell+1}) \tag{3.5}$$

$$t \mapsto \frac{d^k}{dt^k} \frac{\partial}{\partial U_i} R(\beta + te_{\ell+1}) - s \frac{d^k}{dt^k} \frac{\partial}{\partial U_{\ell+1}} R(\beta + te_{\ell+1}), \tag{3.6}$$

treating the second polynomial as a polynomial of formal degree e, where e is the maximal degree achieved by the second polynomial (as a polynomial in t) as s ranges over $\mathbb{C}$. We can thus view the Sylvester resultant as a real univariate polynomial in the variable s; let q_i denote this polynomial.

From the coefficients of the polynomials (3.1) and (3.2), a real non-zero constant multiple of q_i can be computed quickly in parallel by using the algorithm of proposition 2.7 to evaluate the determinant of the Sylvester matrix for particular values of s, and then interpolating using the algorithm of lemma 2.6.

Note that $y_i(\tau)$ is a zero of q_i for all i. This property of q_i will be especially important for us. However, the above construction can produce q_i that are identically zero; this we must avoid. We now modify the construction to produce $q_i \not\equiv 0$ which still has $y_i(\tau)$ as a zero for all τ.

Rewrite the univariate polynomials (3.5) and (3.6) as $t \mapsto p_1(t)$ and $t \mapsto p_2(t) - sp_3(t)$, respectively. Using the algorithm of proposition 2.8, replace p_1, p_2 and p_3 with the polynomials $\bar{p}_1$, $\bar{p}_2$ and $\bar{p}_3$ occurring in the statement of that proposition. Let $q_i(s)$ denote the Sylvester resultant arising from the polynomials $t \mapsto \bar{p}_1(t)$, $t \mapsto \bar{p}_2(t) - s\bar{p}_3(t)$. Since $\bar{p}_1$, $\bar{p}_2$ and $\bar{p}_3$ share

no common zero, $q_i \not\equiv 0$. Since $p_3(t(\tau)) \neq 0 = p_1(t(\tau))$ for all $\tau \in \mathcal{T}$, we have that $\bar{p}_1(t(\tau)) = 0$ for all $t \in \mathcal{T}$. Consequently, since $p_2/p_3 = \bar{p}_2/\bar{p}_3$, we have that $y_i(\tau)$ is a zero of q_i for all $\tau \in \mathcal{T}$.

Henceforth, we may assume that $q_i \not\equiv 0$ and $q_i(y_i(\tau)) = 0$ for all $\tau \in \mathcal{T}$.

For each $i = 1,...,\ell$, apply the algorithm for approximating zeros of univariate polynomials to q_i to obtain a set of $D^{O(1)}$ points $\{s_{ij}\}_j \subset \mathbb{R}$ with the property that for each real zero $\bar{s}$ of q_i that satisfies $|\bar{s}| \leq r$, there exists j such that $|s_{ij} - \bar{s}| \leq \frac{\epsilon}{\ell}$. In particular, for each pair (i,τ) where $||y(\tau)|| \leq r$, there exists $j' = j(i,\tau)$ satisfying $|y_i(\tau) - s_{ij'}| \leq \frac{\epsilon}{\ell}$.

We will discuss a procedure that, given i,j and $\tau \in \mathcal{T}$, efficiently determines if $|y_i(\tau) - s_{ij}| \leq \frac{\epsilon}{\ell}$. Applying this procedure to all triples (i, j, τ), one can then efficiently determine the set $\mathcal{T}^*$ consisting of those τ with the property that for each i there exists $j' = j(i,\tau)$ satisfying $|y_i(\tau) - s_{ij'}| \leq \frac{\epsilon}{\ell}$; of course one determines indices $j(i,\tau)$ as well. (If there are several such indices for some pair i and τ, discard all but one of them and denote it by $j(i,\tau)$.) Then the set of points $\{(s_{1,j(1,\tau)},...,s_{\ell,j(\ell,\tau)});\ \tau \in \mathcal{T}^*\}$ will contain an ϵ-approximation to each point $y(\tau)$ $(\tau \in \mathcal{T})$ satisfying $||y(\tau)|| \leq r$.

Finally, to complete the proof, here is the procedure that, given i, j and $\tau \in \mathcal{T}$, efficiently determines if $|y_i(\tau) - s_{ij}| \leq \frac{\epsilon}{\ell}$. Consider the following system of $D+3$ bivariate polynomials:

$$t \mapsto \frac{d^w}{dt^w} \nabla R(\beta + te_{\ell+1}) \qquad w = 0,...,D\text{-}1$$

$$t \mapsto \frac{d^k}{dt^k} \frac{\partial}{\partial U_{\ell+1}} R(\beta + te_{\ell+1})$$

$$(s,t) \mapsto \frac{d^k}{dt^k} \frac{\partial}{\partial U_i} R(\beta + te_{\ell+1}) - s \frac{d^k}{dt^k} \frac{\partial}{\partial U_{\ell+1}} R(\beta + te_{\ell+1})$$

$$s \mapsto \frac{\epsilon^2}{\ell^2} - (s - s_{ij})^2. \tag{3.7}$$

Relying on the algorithm of proposition 2.1, the set of consistent sign vectors for this system can be efficiently constructed. Using proposition 2.5, the point $t = t(\tau)$, $s = y_i(\tau)$ is the unique point at which the sign vector of the sytem has $(\tau,0)$ as its first $D+2$ coordinates. Searching through the consistent sign vectors we can thus find the sign vector corresponding to $t = t(\tau)$, $s = y_i(\tau)$; the last coordinate of that sign vector is 1 if and only if $|y_i(\tau) - s_{ij}| \leq \frac{\epsilon}{\ell}$.

The operation and time bounds of the proposition follow easily from the propositions and lemma referred to in the construction. □

Before continuing, we note the following. If the coefficients of the polynomials (3.1) and (3.2) are integers of bit length at most L, then the coefficients of the polynomial q_i occurring in the proof of the proposition will be integers of bit length at most $LD^{O(1)}$. In particular, since $y_i(\tau)$ (as in the proof) is a zero of q_i for all $\tau \in \mathcal{T}$, it follows from lemma 2.9 that $\log(||\mathrm{Aff}(\xi^{(i)})|| \leq \log(\ell) + LD^{O(1)}$ for all i such that $\mathrm{Aff}(\xi^{(i)})$ is well-defined. This fact will be important in establishing proposition 1.3.

Let $\mathrm{Cost}(d,r,\epsilon)$, $\mathrm{Time}(d,r,\epsilon,N)$, $\mathrm{Cost}(L,d,r,\epsilon)$ and $\mathrm{Time}(L,d,r,\epsilon,N)$ be as defined just prior to proposition 3.1. We can now easily establish the following theorem.

Theorem 3.2. Given $0 < \epsilon < r$ and a formula (1.6), a set $\{y^{(i)}\}_i$ of $(md)^{2^{O(\omega)}\ell\prod_k n_k}$ distinct ϵ-approximate solutions satisfying the following property can be efficiently constructed: for each connected component of SOLUTIONS(r) there exists $y^{(i)}$ within distance ϵ of the component.

By "efficiently constructed," we mean that the set $\{y^{(i)}\}_i$ can be constructed with $(md)^{2^{O(\omega)}\ell\prod_k n_k}\ \mathrm{Cost}(D, r, \frac{\epsilon}{\ell})$ operations and $(md)^{2^{O(\omega)}\ell\prod_k n_k}$ calls to $\mathbb{P}$, where $D = (md)^{2^{O(\omega)}\ell\prod_k n_k}$. It can be constructed in time

$$[2^{\omega}\ell(\prod_k n_k)\log(md)]^{O(1)} + \mathrm{Time}(D, r, \tfrac{\epsilon}{\ell}, N_1) + \mathrm{Time}(\mathbb{P}, N_2)$$

if $N_1(md)^{2^{O(\omega)\ell\prod_k n_k}}$ parallel processors are used for operations and $N_2(md)^{2^{O(\omega)\ell\prod_k n_k}}$ parallel processors are used for calls (for any $N_1, N_2 \geq 1$).

If the formula has only integer coefficients of bit length at most L, then the set $\{y^{(i)}\}_i$ can be constructed with

$$[\hat{L}(\log \hat{L})(\log\log \hat{L}) + \text{Cost}(\bar{L}, D, r, \tfrac{\epsilon}{\ell})]\,(md)^{2^{O(\omega)\ell\prod_k n_k}}$$

sequential bit operations, and $(md)^{2^{O(\omega)\ell\prod_k n_k}}$ calls to $\mathbb{P}$, where $\hat{L} = L + \log(\ell) + |\log(\epsilon)| + |\log(r)|$ and $\bar{L} = LD^{O(1)}$. It can then be constructed in time

$$(\log \hat{L})[2^{\omega}\ell(\prod_k n_k)\log(md)]^{O(1)} + \text{Time}(\bar{L}, D, r, \tfrac{\epsilon}{\ell}, N_1) + \text{Time}(\mathbb{P}, N_2)$$

if $(\hat{L}^2 + N_1)(md)^{2^{O(\omega)\ell\prod_k n_k}}$ processors are used for bit operations and $N_2(md)^{2^{O(\omega)\ell\prod_k n_k}}$ processors are used for calls (for any $N_1, N_2 \geq 1$). The points $\{y^{(i)}\}_i$ will then have rational coordinates with numerator and denominator bounded in bit length by $O(\log(\ell) + |\log(r)| + |\log(\epsilon)|)$.

Proof. Replace the quantifier free formula $P(y, x^{[1]},\ldots,x^{[\omega]})$ in (1.6) with the formula

$$\bar{P}(y, x^{[1]},\ldots,x^{[\omega]}) := P(y, x^{[1]},\ldots,x^{[\omega]}) \wedge (||y||^2 \leq r^2).$$

Let $\{h_i\}_i$ denote the set of polynomials $h_i\colon \mathbb{R}^{\ell} \to \mathbb{R}$ as constructed in proposition 2.2, assuming that $\{g_i\}_i$ is replaced with $\{g_i\}_i \cup \{y \mapsto ||y||^2 - r^2\}$, and let $\mathcal{R}\{h_i\}_i$ denote the set of polynomials $R\colon \mathbb{R}^{\ell+1} \to \mathbb{R}$ as in proposition 2.3.

Propositions 2.2, 2.3 and 3.1 together easily imply that one can efficiently construct a set $\{y^{(i)}\}_i$ of $(md)^{2^{O(\omega)}\ell\prod_k n_k}$ points $y^{(i)} \in \mathbb{R}^\ell$ with the following property; for each connected component of SOLUTIONS(r) there exists $y^{(i)}$ within distance ϵ of the component.

All that remains to be accomplished is the discarding of those points $y^{(i)}$ that are not within distance ϵ of any connected component of SOLUTIONS(r). Determining which $y^{(i)}$ to discard is equivalent to determining which sentences

$$(\exists\ y \in \mathbb{R}^\ell)(Q_1 x^{[1]} \in \mathbb{R}^{n_1})\ldots(Q_\omega x^{[\omega]} \in \mathbb{R}^{n_\omega})[\bar{P}(y, x^{[1]}, \ldots, x^{[\omega]}) \wedge (||y - y^{(i)}||^2 \leq \epsilon^2)]$$

are false. Relying on the algorithm of theorem 1.1, this can be accomplished efficiently.

The operation and time bounds stated in the theorem are easy consequences of the propositions cited along with the bounds provided by theorem 1.1. □

In closing this section we complete the proof of proposition 1.3. Let $\{h_i\}_i$ denote the set of polynomials $h_i: \mathbb{R}^\ell \to \mathbb{R}$ as constructed in proposition 2.2 (not assuming that $\{g_i\}_i$ is replaced as in the above proof). Let $\mathcal{R}\{h_i\}_i$ denote the set of polynomials $R: \mathbb{R}^{\ell+1} \to \mathbb{R}$ as in proposition 2.3. Then for each connected component of SOLUTIONS, there exists $R \in \mathcal{R}\{h_i\}_i$ such that R factors linearly $R(U) = \prod_i \xi^{(i)} \cdot U$ where for some i, $\mathrm{Aff}(\xi^{(i)})$ is in the component. If the coefficients occurring in the formula (1.6) are all integers of bit length at most L, propositions 2.2 and 2.3 show that the coefficients of the corresponding univariate polynomials (3.1) and (3.2) obtained from $R \in \mathcal{R}\{h_i\}_i$ will be integers of bit length at most $L(md)^{2^{O(\omega)}\ell\prod_k n_k}$. Lemma 1.4 is now a consequence of the observations immediately following the proof of proposition 3.1 along with lemma 2.9.

4. BOUNDS FOR UNIVARIATE POLYNOMIAL ZERO APPROXIMATION

In this section we briefly review some known upper bounds on the operations and time required to compute approximations to the real zeros of real univariate polynomials.

Recall that P_d was defined as the set of all non-constant real univariate polynomials of degree at most d, and $P_{L,d}$ was defined as the subset of P_d consisting of polynomials all of whose coefficients are integers of bit length at most L.

Perhaps the best known method for approximating real zeros of univariate polynomials is via Sturm sequences (e.g., see Henrici [14]). Given $f \in P_d$ and $0 < \epsilon < r$, the method proceeds by bisection and computes ϵ-approximations to all of the real zeros $\bar{s}$ of f satisfying $|\bar{s}| \leq r$. The number of operations required is only $d^{O(1)}\log(1 + \frac{r}{\epsilon})$; the operations can be implemented in parallel, requiring time $[\log(\frac{dr}{\epsilon})]^{O(1)}$ if $d^{O(1)}$ parallel processors are used.

For readers unfamiliar with Sturm sequences, similar bounds can be obtained by invoking theorem 1.1 to design an algorithm for approximating the real zeros of univariate polynomials. First, the algorithm of theorem 1.1 is used to determine if the interval $[-r,r]$ contains a zero of f; of course this is equivalent to determining if the sentence

$$(\exists\, s \in \mathbb{R})[(f(s) = 0) \wedge (s \leq r) \wedge (s \geq -r)]$$

is true. If the sentence is false we terminate. Otherwise we bisect the interval and query which of the two smaller intervals contains a zero. And so on. This approach, together with theorem 3.2, produces the second algorithm of theorem 1.2.

The bit complexity algorithm of theorem 1.2 is obtained by combining theorem 3.2 with Neff [16]. As mentioned in the introduction, Neff showed that there is an efficient parallel bit-model algorithm for approximating all zeros of a complex univariate polynomial; assuming ϵ is an integral

power of 2, he showed that ϵ-approximations to all zeros can be obtained in time $[\log(L + d + |\log(\epsilon)|)]^{O(1)}$ using $(L + d + |\log(\epsilon)|)^{O(1)}$ processors. Taking the real part of each of these approximations, we obtain a set of real numbers containing ϵ-approximations to all of the real zeros of the polynomial; although some of the real numbers in the set thus obtained may not approximate any zero, theorem 3.2 is still applicable--the algorithm developed to prove the theorem "weeds out" points which are not approximations.

In Renegar [17], an algorithm is presented which obtains approximations to all zeros (including the complex zeros) of a polynomial $f \in P_d$. The main results of that paper are presented assuming that an upper bound R on the absolute values of all of the zeros is known apriori. The operation bound presented is of the form $d^{O(1)} \log \log(3 + \frac{R}{\epsilon})$. (Specific small exponents are presented rather than relying on "O(1)".) The significance of the bound is its extremely low dependence on R/ϵ; as discussed in the introduction of the present paper, it is proven in [17] that this dependence is optimal.

Although the main results in [17] are stated in the introduction of that paper under the assumption that an upper bound R on the absolute values of all of the zeros is known apriori, the algorithm and analysis of section 8 of that paper were written to establish the following; if $f \in P_d$, $x \in \mathbb{C}$ and $r > 0$ are such that no zeros of f are contained in the region $\{y \in \mathbb{C};\ r < ||y\text{-}x|| \leq 80d^5 r\}$, then ϵ-approximations to all zeros of f in $\{y;\ ||y\text{-}x|| \leq r\}$ can be obtained with $d^{O(1)} \log \log(3 + \frac{r}{\epsilon})$ operations.

In the application of the present paper we are given $r > 0$ and wish to obtain a set of $d^{O(1)}$ points containing ϵ-approximations to all real zeros $\bar{s}$ of f satisfying $|\bar{s}| \leq r$. If we know apriori that the region $\{y \in \mathbb{C};\ r < ||y|| \leq 80d^5 r\}$ contains no zeros of f then, by the preceding paragraph, this can be accomplished quickly (i.e., quickly in terms of r and ϵ). If the region does contain a zero of f (as can be efficiently determined using the algorithm of theorem 1.1), then we can design a simple bisection algorithm, that calls on the decision algorithm of theorem 1.1, to construct $k \leq d$ points $x_1,\ldots,x_k \in [-r,r]$ and radii $r_1,\ldots,r_k \leq r$ with the properties that (i) all real zeros of f that are

contained in $[-r, r]$ are contained in $\bigcup_j [x_j-r_j, x_j+r_j]$ and (ii) if $\bar{s} \in \mathbb{C}$ satisfies $f(\bar{s}) = 0$ and $||\bar{s}-x_j|| > r_j$ then $||\bar{s}-x_j|| \geq 80d^5 r_j$. This construction will only require $d^{O(1)}$ operations. For these smaller intervals we can rely on the algorithm of section 8 of Renegar [17] to obtain approximations to the zeros within. The observation of the preceding paragraph shows that for each of these smaller intervals, all zeros within the interval can be approximated quickly. In all, relying on the bounds of theorem 1.1 and the preceding paragraph, ϵ-approximations to all zeros within the interval $[-r, r]$ can be obtained with $d^{O(1)} \log \log(3 + \frac{r}{\epsilon})$ operations. This result, combined with theorem 3.2, yields the first algorithm in theorem 1.2.

REFERENCES

[1] S.J. Berkowitz, "On computing the determinant in small parallel time using a small number of processors," Information Processing Letters 18 (1984), 147-150.

[2] A. Borodin, J. von zur Gathen, J. Hopcroft, "Fast parallel matrix and GCD computations," Information Control 52 (1982) 241-256.

[3] L. Blum, M. Shub, S. Smale, "On a theory of computation and complexity over the real numbers: NP-completeness, recursive functions and universal machines," Bulletin of the American Mathematical Society 21 (1989), 1-46.

[4] L. Blum and S. Smale, "The Godel incompleteness theorem and decidability over a ring," to appear in "From Topology to Computation; Proceedings of the Smalefest," Springer-Verlag.

[5] W. Brown, J. Traub, "On Euclid's algorithm and the theory of subresultants," Journal of the Association for Computing Machinery 18 (1971), 505-514.

[6] J. Canny, The Complexity of Robot Motion Planning, MIT Press, Cambridge, MA, 1988.

[7] A.L. Chistov and D.Y. Grigor'ev, "Subexponential time solving systems of algebraic equations, I and II," LOMI preprints E-9-83, E-10-83, Leningrad.

[8] G.E. Collins, "Quantifier elimination for real closed fields by cylindrical algebraic decomposition," Second GI Conference on Automata Theory and Formal Languages. Lecture Notes in Computer Science 33 (1975) 134-183, Springer-Verlag.

[9] L. Csanky, "Fast parallel matrix inversion algorithms," SIAM Journal on Computing 5 (1976), 618-623.

[10] M. Coste and M.F. Roy, "Thom's lemma, the coding of real algebraic numbers and the computation of the topology of semi-algebraic sets," Journal of Symbolic Computation 5 (1988), 121-129.

[11] D. Yu. Grigor'ev, N.N. Vorobjov, "Solving systems of polynomial inequalities in subexponential time," Journal of Symbolic Computation 5 (1988), 37-64.

[12] D. Grigor'ev, "The complexity of deciding Tarski algebra," Journal of Symbolic Computation 5 (1988), 65-108.

[13] J. Heintz, M.-F. Roy, P. Solernó, "Sur la complexité du principe de Tarski-Seidenberg," Bull. Soc. Math. France 118 (1990), 101-126.

[14] P. Henrici, Applied and Computational Complex Analysis, Vol. 1, Wiley-Interscience, New York.

[15] D. Lazard, "Résolution des systèmes d'équations algébriques," Theoretical Computer Science 15 (1981), 77-110.

[16] C.A. Neff, "Specified precision polynomial root isolation is in NC," Proceedings of the 21st Annual IEEE Symposium on the Foundations of Computer Science (1990); full version to appear in the Journal of Computer and System Science.

[17] J. Renegar, "On the worst-case arithmetic complexity of approximating zeros of polynomials," Journal of Complexity 3 (1987), 90-113.

[18] J. Renegar, "On the worst-case arithmetic complexity of approximating zeros of systems of polynomials," SIAM Journal on Computing 18 (1989), 350-370.

[19] J. Renegar, "On the computational complexity and geometry of the first order theory of the reals. Part I: introduction; preliminaries; the geometry of semi-algebraic sets; the decision problem for the existential theory of the reals," to appear in the Journal of Symbolic Computation.

[20] J. Renegar, "On the computational complexity and geometry of the first order theory of the reals. Part II: the general decision problem; preliminaries for quantifier elimination," to appear in the Journal of Symbolic Computation.

[21] J. Renegar, "On the computational complexity and geometry of the first order theory of the reals. Part III: quantifier elimination," to appear in the Journal of Symbolic Computation.

[22] J. Renegar, "Recent progress on the complexity of the decision problem for the reals," to appear in the Proceedings of the DIMACS Workshop on Algebraic Methods in Geometric Computations, American Mathematical Society.

[23] A. Tarski, A Decision Method for Elementary Algebra and Geometry, University of California Press (1951).

[24] N.N. Vorobjov Jr., "Bounds of real roots of a system of algebraic equations," Notes of Sci. Seminars of Leningrad Dept. of Math. Steklov Inst. 137 (1984), 7-19.