ON THE INTRACTABILITY OF HILBERT'S NULLSTELLENSATZ AND AN ALGEBRAIC VERSION OF "$NP \neq P$?"

Michael Shub
Mathematical Sciences Department
IBM Watson Research Center
Yorktown Heights, NY 10598
shub@watson.ibm.com
Steve Smale
Department of Mathematics
University of California at Berkeley
Berkeley, CA 94720
smale@math.berkeley.edu

June 21, 1994

Section 1. Introduction

In this paper we relate an elementary problem in number theory to the intractability of deciding whether an algebraic set defined over the complex numbers (or any algebraically closed field of characteristic zero) is empty.

More precisely, we first conjecture:

The Hilbert Nullstellensatz is intractible.

The Hilbert Nullstellensatz is formulated as a decision problem as follows.

Given $f_1, \ldots, f_\ell : \mathbb{C}^m \to \mathbb{C}$, complex polynomials of degree d_i, $i = 1, \ldots, \ell$, decide if there is a $z \in \mathbb{C}^m$ such that $f_i(z) = 0$ for all i.

There is an algorithm for accomplishing this task. From Hilbert, the answer is no if and only if there are polynomials $g_i : \mathbb{C}^m \to \mathbb{C}$, $i = 1, \ldots, \ell$ with the property

$$\sum_{i=1}^{\ell} g_i f_i = 1 \qquad (*)$$

Brownawell has made the most decisive next step by finding a good bound on the degrees of these g_i. With that one may decide if (*) has a solution by linear algebra,

The authors were partially supported by NSF grants. The work was carried out at Instituto de Matemática Pura e Aplicada, Rio de Janeiro.

since (*) is a finite dimensional linear system with the g_i's as unknowns. This procedure is called the "effective Nullstellensatz".

To say what "intractable" means in our conjecture, it is necessary to have a formal definition of algorithm in this context. That is done in *Blum-Shub-Smale* (referred to here as *BSS*). In that paper, algebraic algorithms (called algorithms over $\mathbb{C}$) are described in terms of "machines" which make arithmetical computations and branch according to whether a variable (the first state variable) is zero or not.

In *BSS*, furthermore, one has the concept of a polynomial time algorithm, in particular, a polynomial time decision algorithm over $\mathbb{C}$. This may be expressed in the present setting as:

$$T(f) \leq s(f)^c \quad \text{(the } c \text{ power of } s(f) \text{ all } f\text{)}$$

Here $f = (f_1, \ldots, f_\ell)$ is the input and $T(f)$ is the number of operations (arithmetic, branching) used to accomplish the decision, and $s(f)$ is the total number of coefficients of the f_i (input size). Also c is a universal constant. The input size is:

$$s(f) = \sum_{i=1}^{\ell} \binom{m + d_i}{d_i}$$

Our conjecture is now formally the mathematical statement: There is no polynomial time algorithm over $\mathbb{C}$ which decides the Hilbert Nullstellensatz.

An algebraic version of the computer science problem "$NP \neq P$?" is also introduced in *BSS*. From that paper it follows that the Nullstellensatz is a universal decision problem in a certain sense. It is "NP complete over $\mathbb{C}$". It follows that "$NP \neq P$ over $\mathbb{C}$." if and only if our main conjecture is true.

In other words we may assert: the algebraic version of $NP \neq P$ is true if and only if the Hilbert Nullstellensatz is intractible.

Valiant has also an algebraic theory of NP completeness which differs from ours in his focus on "formula size" which is not equivalent to a computational notion. Moreover his model is not uniform and doesn't permit branching on a variable $x \neq 0$.

A *computation* of *length* ℓ of the integer m is a sequence of integers, $x_o, x_1, \ldots, x_\ell$ where $x_o = 1$, $x_\ell = m$ and given k, $1 \leq k \leq \ell$, there are i, j, $0 \leq i, j < k$ such that $x_k = x_i \circ x_j$ where $\circ$ is addition, subtraction or multiplication. We define $\tau : \mathbb{Z} \to \mathbb{N}$ (the natural numbers) by $\tau(m)$ is the minimum length of a computation of m.

The following is easy to check.

Proposition $\tau(m) \leq 2 \log m$.

If m is of the form 2^{2^k}, then $\tau(m) = \log \log m + 1$. The same is essentially true even if m is any power of 2.

We raised the question as to whether $\tau(m) \leq (\log \log m)^c$, where c is independent of m. Welington *de Melo* and Benar F. *Svaiter* showed by a counting argument that the answer is no. H. Lenstra also tells us that Jeff Shallit answered our question as well. Carlos Gustavo *Moreira* subsequently gave quite sharp estimates on this problem.

Yet our second question remains unanswered.

Problem. Is the a constant c such that

$$\tau(k!) \leq (\log\ k)^c \quad \text{all } k\,?$$

Given a sequence of integers a_k we say that a_k is *easy* to *compute* if there is a constant c such that $\tau(a_k) \leq (\log\ k)^c$, all $k > 2$, and *hard* to *compute* otherwise. We say that the sequence a_k is *ultimately easy* to *compute* if there are non-zero integers m_k such that $m_k a_k$ is easy to compute and *ultimately hard to compute* otherwise.

Main Theorem. If the sequence of integers $k!$ is ultimately hard to compute then the Hilbert Nullstellensatz is intractable, and consequently the algebraic version of "$NP \neq P$" is true.

To prove the Main Theorem, we consider an intermediate decision problem which we call *twenty questions*:

$$\begin{aligned} &\text{Input} \quad (k, ht(k), z) \in \mathbb{N} \times \mathbb{N} \times \mathbb{C} \\ &\text{Decide if} \quad z \in \{1, 2, \dots, k\}. \end{aligned}$$

Here $ht(k)$ is defined to be the largest natural number less than or equal to $\log\ k$.

Theorem 1. If the Hilbert Nullstellensatz is tractable, i.e., if $NP = P$ over $\mathbb{C}$, then there is a machine $\mathcal{M}$ over $\mathbb{C}$ (in the sense of *BSS*) and a constant c such that $\mathcal{M}$ decides twenty questions in time bounded by $(\log\ k)^c$.

Theorem 2. If a machine over $\mathbb{C}$ (in the sense of *BSS*) decides twenty questions in time bounded by $(\log\ k)^c$ for some constant c, then the sequence $k!$ is ultimately easy to compute.

The Main Theorem follows immediately from Theorems 1 and 2. Theorem 1 is fairly simple in our computational setting, its proof is carried out in section 2. Most of the substance of our paper is in the proof of Theorem 2. For this τ must be extended to polynomial rings. The algebraic and transcendental constants used by the machine must be circumvented. These arguments are carried out in sections 3 and 4.

The complexity of deciding twenty questions was considered in a slightly different context in *Shub*. The paper by *Heintz-Morgenstern* is related to our work here.

Section 2. Proof of Theorem 1 (of section 1)

We prove Theorem 1 by embedding "twenty questions" in a decision problem (Y, Y_{yes}) which is in NP over $\mathbb{C}$.

Then if $NP = P$ over $\mathbb{C}$, (Y, Y_{yes}) is in P and there is a machine $\mathcal{M}$ which decides "twenty questions" in time bounded by $(\log\ k)^c$, c a constant. Here $\mathcal{M}$ is the restriction of the machine which decides (Y, Y_{yes}) in polynomial time.

The decision problem (Y, Y_{yes}) is described as follows.

$Y = \mathbb{C}^\infty$ and $Y_{\text{yes}} = \bigcup_{k \in \mathbb{N}}\ Y_{\text{yes},k}$ where

$$Y_{\text{yes},k} = \{(k, ht(k), z_1, \dots, z_{ht(k)}, 0, \dots) | z_1 \in \{1, \dots, k\}\}.$$

The embedding of "twenty questions" in (Y, Y_{yes}) is simply:

$$(k, ht(k), z) \to (z, ht(k), z, 1, \dots, 1, 0, 0, \dots)$$

where the number of ones is $ht(k) - 1$.

The proof of Theorem 1 is finished by the next proposition.

Proposition. (Y, Y_{yes}) is in NP over $\mathbb{C}$.

Proof. The NP machine operates on variables

$$(u_1, u_2, z_1, \dots, z_n, w_o, \dots, w_n, v_{jo}, \dots, v_{jn} \quad \text{for} \quad j = 1, 2, 3, 4)$$

It checks if u_2 is an integer by addition of 1's. It checks if the input size (given with the input by definition) is $6u_2 + 5$. If so $n = u_2$. It checks if $w_n = 1$, $w_i(w_i - 1) = 0$ and $v_{ji}(v_{ji} - 1) = 0$ for $i = 0, \dots, n$ and $j = 1, 2, 3, 4$. It checks if $u_1 = \sum_{i=0}^{n} 2^i w_i$. It sets $x_j = \sum_{i=0}^{n} 2^i v_{ji}$ for $j = 1, 2, 3, 4$. Finally it checks if $u_1 = z_1 + \sum_{j=1}^{4} x_j^2$. If so it outputs Yes. Note that if the tests are verified, the w's and v's are 0 or 1; u_1, the x_j and hence z_1 are non-negative integers and $u_2 = ht(u_1)$. The time required is a constant times u_2.

Finally we show that every element of $Y_{\text{yes},k}$ has a positive test. Let

$$(k, ht(k), z_1, \dots, z_{ht(k)}, 0, \dots) \in Y_{\text{yes},k}$$

Then z_1 is a non-negative integer so that $k - z$, is sum of four integers squared,

$$k - z_1 = x_1^2 + x_2^2 + x_3^2 + x_4^2$$

Q.E.D.

Section 3. Easy to compute sequences in rings

In this section we prove the facts about easy to compute sequences which are needed for the proof of Theorem 2 of the introduction. These concepts are close to those of algebraic complexity theory; see for example *Heintz, Heintz-Morgenstern*.

Given a ring (or field) R and generators $g_o, \dots, g_n$ of R, a *computation* of *length* ℓ of the element $r \in R$ is a sequence of elements $r_{-n}, \dots, r_o, r_1, \dots, r_\ell$ where $r_{-i} = g_i$ for $0 \le i \le n$, $r_\ell = r$ and moreover: given k between 1 and ℓ (inclusive), there are p, q with $-n \le p, q < k$ such that $r_k = r_p \circ r_q$ where $\circ$ is the operation of addition, subtraction, or multiplication (or division by a non-zero element if R is a field).

Define $\tau = \tau_{g_o, \dots, g_m} : R \to \mathbb{N}$ by $\tau(r)$ is the minimum length of a computation of r.

Note that the $\tau : \mathbb{Z} \to \mathbb{N}$ of the introduction is a special case.

Proposition 1. Let $(g_o, \dots, g_n)$ and $(h_o, \dots, h_m)$ be two sets of generators of a ring R. Then there is a constant $c > 0$ such that

$$\tau_{g_o, \dots, g_n}(r) \le \tau_{h_o, \dots, h_m}(r) + c, \quad \text{all } r \in R.$$

The proof is straightforward.

Proposition 1 allows one to define hard and easy sequences of elements of R, independently of the choice of generators, exactly as in the introduction for $\mathbb{Z}$.

Proposition 2. Let G and H be finitely generated rings (or fields). Let $\phi : G \to H$ be a ring homomorphism of G onto H. If $g_k \in G$ is an easy to compute sequence then so is $\phi(g_k) \in H$.

Proof. Let $e_1, \ldots, e_n$ be a set of generators of G. Then $\phi(e_1), \ldots, \phi(e_n)$ is a set of generators of H. Thus

$$\tau_{\phi(e_1),\ldots,\phi(e_n)}(\phi(g_k)) \leq \tau_{e_1,\ldots,e_n}(g_k), \quad \text{all } k.$$

Q.E.D.

Proposition 3. Let R be a finitely generated integral domain and K its quotient field
(i) If $f_k \in K$ is an easy to compute sequence in K then there are easy to compute sequences p_k, q_k in R such that $f_k = p_k/q_k$ all k.
(ii) Let $f_k \in K[t_1, \ldots, t_n, \lambda_1, \ldots, \lambda_m]$ be an easy to compute sequence where $t_1, \ldots, t_n$ are variables and $\lambda_1, \ldots, \lambda_m$ are elements of an extension of K.
Then there are easy to compute sequences $p_k \in R[t_1, \ldots, t_n, \lambda_1, \ldots, \lambda_n]$ $q_k \in R$ such that $f_k = p_k/q_k$ all k.

Proof. We can assume that the generators are $g_1, \ldots, g_\ell$, $t_1, \ldots, t_n$, $\lambda_1, \ldots, \lambda_m$ where the g_i generate R. Now use the instructions for computing f_k to compute (p_k, q_k). For example,

$$(p_k, q_k) + (p_j, q_j) = (p_i q_j + p_j q_i, q_i q_j)$$

Q.E.D.

Theorem 1. Let $f_i \in \mathbb{Q}(t, \lambda_1, \ldots, \lambda_m)$ be an easy to compute sequence of non-trivial rational functions in the variable t and transcendentally independent complex numbers $\lambda_1, \ldots, \lambda_m$. Then there is an easy to compute sequence of integral polynomials $p_i \in \mathbb{Z}[t]$ such that $p_i \neq 0$ all i and for $z \in \mathbb{Q}$, $p_i(z) = 0$ whenever $f_i(z, \lambda_1, \ldots, \lambda_m) = 0$.

For the proof we use two lemmas.

Lemma 1. Let a polynomial $f \in \mathbb{Z}[t_o, t_1, \ldots, t_m]$ have degree d. If f is zero on every integer point in the cube in $\mathbb{R}^{m+1}$ centered at $(0, \ldots, 0)$ with side having length $(d+1)$, then f is identically zero.

The proof is a straight forward induction on m.

Lemma 2. Let $f_i \in \mathbb{Z}[t, \lambda_1, \ldots, \lambda_m]$ be an easy to compute sequence of non-trivial integral polynomials in the variable t and transcendentally independent complex numbers $\lambda_1, \ldots, \lambda_m$. Then there is an easy to compute sequence of non-trivial integral polynomials $p_i \in \mathbb{Z}[t]$ such that for $z \in \mathbb{Q}$, $p_i(z) = 0$ whenever $f_i(z, \lambda_1, \ldots, \lambda_m) = 0$.

For the proof we may assume that 1, t, $\lambda_1, \ldots, \lambda_m$ are the generators of $\mathbb{Z}\,[t, \lambda_1, \ldots, \lambda_m]$ that we use for defining computational length.

Let n_i be the computational length of f_i. Then the degree of f_i is less than $2^{n_i} + 1$. Using Lemma 1, considering the λ_i as variables, there is an $(m+1)$-tuple of integers $(k_{io}, \ldots, k_{im}) = k$ such that $f_i(k_{io}, \ldots, k_{im}) \neq 0$ and $|k_{i,j}| \leq 2^{n_i} + 1$, all i, j. By the

proposition of section 1, $\tau(k_{i,j}) \leq 2(n_i+1)$. Write $f_i(t, \lambda_1, \ldots, \lambda_m) = \sum_I a_{i,I}(t)\lambda^I$ where I is a multi-index (a finite sum, of course). Since λ_j, $j = 1, \ldots, m$, are transcendentally independent, $f_i(z, \lambda_1, \ldots, \lambda_m) = 0$ for a rational number z if and only if $a_{iI}(z) = 0$ for all I.

Let $k_i' = (k_{i1}, \ldots, k_{im})$. Then if $f_i(z, \lambda_1, \ldots, \lambda_m) = 0$ for some rational z, we see that $p_i(t) = \sum_I a_{i,I}(t)(k_i')^I$ vanishes at $t = z$. Note also that $p_i(k_{i,o}) \neq 0$.

Finally by computing k_{ij} first and substituting k_{ij}, $j = 1, \ldots, m$ in the instructions for computing f_i, p_i is computed with computational length at most $n_i + 2m(n_i + 1)$ and so p_i is an easy to compute sequence.

Now we return to the proof of Theorem 1.

By Proposition 3(i) one finds easy to compute sequences p_i, q_i' in $\mathbb{Q}[t, \lambda_1, \ldots, \lambda_m]$ such that $p_i'q_i' = f_i$. By Proposition 3(ii) we find easy to compute sequences $p_i'' \in \mathbb{Z}[t, \lambda_1, \ldots, \lambda_m]$, $q_i'' \in \mathbb{Z}$ such that $p_i' = p_i''q_i''$. Thus p_i'' is not zero. Also $p_i''(z, \lambda_1, \ldots, \lambda_m) = 0$ if z is rational and $f_i(z, \lambda_1, \ldots, \lambda_m) = 0$. Now using p_i'' in Lemma 2 finishes the proof.

Section 4. Proof of Theorem 2 (of Section 1)

The proof is preceded by two propositions.

Let a machine $\mathcal{M}$ solve a decision problem (K^∞, Y) where K is a field of characteristic 0 branching on $x = 0$ or $x \neq 0$. Suppose there is a sequence n_k of positive integers with $\mathcal{M}$ halting at time $T(k)$ on inputs of size n_k. Let $K^{n_k} \subset K^\infty$ be the n_k-fold cartesian product of K and $Y_k = Y \cap K^{n_k}$. Under the hypothesis that Y_k is a proper, non-empty subvariety of K^{n_k}, we define the k^{th} *canonical path* as the computation path which at each branch node is taken by a Zariski dense set of inputs in K^{n_k}.

Thus a canonical path may be described as a certain sequence $\gamma_1\gamma_2 \ldots \gamma_\ell$, $\ell < T(k)$ where each γ_j is a branch node and γ_{j+1} is the node encountered by almost all inputs subsequent to γ_j. We omit γ_j in the case that all inputs arriving at γ_j take the same branch (see *Cucker-Shub-Smale*). Branching is determined by a condition $x_1 = 0$ or not. Then x_1 is represented by a rational function G_j defined almost everywhere on K^{n_k}.

It is easy to see that the computational length of G_j is bounded by $c_1 j + c_2$ where c_1, c_2 are constants.

Let H_k be the product of the numerators of the G_j.

Proposition 1. The rational function H_k defined on K^{n_k} vanishes on Y_k but is not identically zero. It's computation length is bounded by $c_1\, T(k) + c_2$.

Proof. The machine must answer no on a Zariski dense set of points of K^{n_k}, so Y_k must be contained in the union of the varieties $V_j = \{x | G_j(x) = 0\}$. This proves the first assertion.

The last assertion is a special case of the remark preceding Proposition 1 and the Proof of Proposition 3.3(i).

Proposition 2. Let $\widehat{\mathcal{M}}$ be a machine over a field $\widehat{K}$ which is a finite algebraic extension of a field K. Then there is a machine $\mathcal{M}$ over K and a constant $c > 0$ such that for any

decision problem (Y, Y_{yes}), $Y \subset \widehat{K}^\infty$, decided by $\widehat{\mathcal{M}}$, $(Y \cap K^\infty, Y_{\text{yes}} \cap K^\infty)$ is decided by $\mathcal{M}$; and moreover the stopping time satisfies:

$$T_{\mathcal{M}}(y) \leq cT_{\widehat{\mathcal{M}}}(y), \quad y \in Y \cap K^\infty.$$

Proof. We may assume that at any computation node, the computation performed is either addition, multiplication, subtraction or division of two elements of $\widehat{K}$. We may regard $\widehat{K}$ as a vector space over K of some fixed dimension q. Thus $\widehat{K}$ can be represented as K^q where the imbedding $K \subset \widehat{K}$ is the inclusion of K in K^q as the first coordinate. Now addition and multiplication are represented by fixed symmetric bilinear maps

$$B_+ : K^q \times K^q \longrightarrow K^q$$
$$B_\times : K^q \times K^q \longrightarrow K^q$$

Division of b by a is accomplished by solving the linear system $B_\times(a, y) = b$ for y by Gaussian elimination. This requires on the order of q^3 steps. To define $\mathcal{M}$, replace $\widehat{K}^\infty$ by $(K^q)^\infty$. The input of K^∞ in $\widehat{K}^\infty$ is replaced by the input of K^∞ as the first coordinates in $(K^q)^\infty$. Multiplication nodes are replaced by $B_\times$ and addition nodes by B_+. Subtraction nodes are replaced by -1 followed by B_+. Branching is done on the first coordinate of the first K^q.

One can see using the isomorphism between K^q and $\widehat{K}$ that on inputs $y \in Y \cap K^\infty$, $\mathcal{M}$ gives the same answers as $\widehat{\mathcal{M}}$ with the desired time bound where c is on the order q^3. Q.E.D.

Proof of Theorem 2. Assume that $\widehat{\mathcal{M}}$ decides "twenty questions" in time bounded by $(\log k)^c$. Let $\mu_1, \ldots, \mu_\ell$ be the non-rational constants of $\widehat{\mathcal{M}}$ so that we may view $\widehat{\mathcal{M}}$ as a machine over $\mathbb{Q}(\mu_1, \ldots, \mu_\ell)$. Now $\mathbb{Q}(\mu_1, \ldots, \mu_\ell)$ is a finite algebraic extension of a finitely generated purely transcendental extension $\mathbb{Q}(\lambda_1, \ldots, \lambda_m)$ of $\mathbb{Q}$. Thus by proposition 2, there is a machine $\mathcal{M}$ over $\mathbb{Q}\ (\lambda_1, \ldots, \lambda_m)$ which solves "twenty questions" restricted to $\mathbb{Q}(\lambda_1, \ldots, \lambda_m)$. Thus on input $(k, ht(k), z)$, $z \in \mathbb{Q}(\lambda_1, \ldots, \lambda_m)$, $\mathcal{M}$ decides if $z \in \{1, \ldots, k\}$ in time bounded by $c_1(\log k)^c$.

Then by proposition 1 there are non-trivial polynomial functions
$f_k \in \mathbb{Q}\ (t, \lambda_1, \ldots, \lambda_m)$ which vanish on $\{1, \ldots, k\}$ and whose computational length in $\mathbb{Q}(t, \lambda_1, \ldots, \lambda_m)$ is bounded by $c_2(\log k)^c + c_3$. Here c_3 is a bound for the computational length of rational constants introduced by the machine $\widehat{\mathcal{M}}$ and c_3 depends only on $\widehat{\mathcal{M}}$. Therefore f_k is an easy to compute sequence of non-trivial rational functions. By Theorem 1 of section 3, there is an easy to compute sequence of non-trivial polynomials $p_k \in \mathbb{Z}[t]$ vanishing on $\{1, \ldots, k\}$. By Lemma 1 of section 3, there is an integer m such that, $p_k(m) \neq 0$, $|m| \leq 2^r + 1$ where r is the computational length of p_k. So $\tau(m) \leq r + 1$. We may assume $|m|$ is minimal with these properties. Then p_k is zero at each integer between 0 and m. Evaluating p_k at m gives a computational length of at most $2r + 1$ for $p_k(m)$. Since $p_k(m)$ is an integral multiple of $k!$, the sequence $k!$ is ultimately easy to compute. Q.E.D.

References

Blum, L. Shub, M., and Smale, S., On a theory of computation and complexity over the real numbers: NP-completeness, recursive functions and universal machines. Bull. Amer. Math. Soc. Vol 21 (1989) pp 1–46.

Brownawell, W.D., Bounds for the degrees in the Nullstellensatz. Ann. Math. (second series) 126 (1987) (3) pp 577–591.

Cucker, F., Shub, M., and Smale, S., Separation of Complexity Classes in Koiran's Weak Model, to appear, Theoret. Comput. Sci.

Heintz, J., On the computational complexity of polynomials and bilinear mappings. A survey, in Springer LN Comput. Sci., Vol 356 (1989) pp 269–300, Springer, N.Y.

Heintz, J., and Morgenstern, J., On the intrinsic complexity of elimination theory. Jour. of Complexity, 9 (1993) pp 471–498.

de Melo, W., and Svaiter, B.F., to appear.

Moreira, G.G., to appear.

Shub, M., Some Remarks on Bezout's Theorem and Complexity Theory in *From Topology to Computation. Proceedings of the Smalefast* (Eds., Hirsch, M.W., Marsden, J.E., and Shub, M.) Springer, 1993 pp 443–455.

Valiant, L.G., Completeness classes in algebra. Conf. Rec. of $11^{\underline{\text{th}}}$ Ann. ACM Stoc. (1979) pp. 249–261. ACM, N.Y.