Laboratoire de l'Informatique du Parallélisme

Ecole Normale Supérieure de Lyon
Unité de recherche associée au CNRS n°1398

Computing over the Reals with Addition and Order

Pascal Koiran

August 24, 1993

Research Report N$^{\text{O}}$ 93-25

Ecole Normale Supérieure de Lyon

46, Allée d'Italie, 69364 Lyon Cedex 07, France,
Téléphone : + 33 72 72 80 00; Télécopieur : + 33 72 72 80 80;
Adresses électroniques :
lip@frensl61.bitnet; lip@lip.ens−lyon.fr (uucp).

Computing over the Reals with Addition and Order

Pascal Koiran

August 24, 1993

Abstract

We provide a fairly complete picture of the complexity theory of additive real machines. This model of computation is a restriction of the real Turing machine of Blum, Shub & Smale, since addition and subtraction are the only legal arithmetic operations. Removing the order relation $<$ on $\mathbb{R}$ yields an even weaker class of machines, which is also studied. Our main results are:

- characterizations of the classes of recognizable boolean languages;
- equivalence of real and digital nondeterminism;
- a simpler proof of Meer's $\mathrm{P_{lin}} \neq \mathrm{NP_{lin}}$ result.

Keywords: computational complexity, real machines, polyhedra.

Résumé

Nous fournissons une description assez complète de la théorie de la complexité des machines réelles additives. Ces machines sont des variantes affaiblies de la machine de Turing réelle de Blum, Shub & Smale, car l'addition et la soustraction sont les seules opérations arithmétiques légales. En supprimant la relation d'ordre $<$ sur $\mathbb{R}$, on obtient une classe de machines encore moins puissantes, qui est étudiée également. Nos principaux résultats sont:

- des caractérisations des classes de langages booléens reconnaissables;
- l'équivalence du non-déterminisme réel et du non-déterminisme digital;
- une preuve plus simple du résultat $\mathrm{P_{lin}} \neq \mathrm{NP_{lin}}$ de Meer.

Mots-clés: complexité, machines réelles, polyèdres.

1 Introduction

The real Turing machine of Blum, Shub & Smale [2] is endowed with the four basic arithmetic operations (addition, subtraction, multiplication, division) and with the order relation $<$ on $\mathbb{R}$. It seems however that multiplication and division are of little help for a number of interesting problems such as the real-valued generalizations of the Traveling Salesman Problem or the Knapsack Problem. In order to study these problems, it is quite natural to work with real machines equipped with addition and subtraction only.

Let $\mathrm{P}^{<}_{\mathrm{add}}$ and $\mathrm{NP}^{<}_{\mathrm{add}}$ be the class of real languages (subsets of $\mathbb{R}^\infty = \bigcup_{n=1}^{\infty} \mathbb{R}^n$) that can be recognized in deterministic and non-deterministic polynomial time in this additive BSS model. It is possible to make fruitful comparisons with discrete complexity theory if we consider real machines with discrete inputs. In this paper we show that:

- $\mathrm{P}^{<}_{\mathrm{add}} \cap \Sigma^* = \mathrm{P/poly}$;
- $\mathrm{NP}^{<}_{\mathrm{add}} \cap \Sigma^* = \mathrm{NP/poly}$.

Here $\Sigma = \{0, 1\}$ is the binary alphabet and we use the notation

$$\mathcal{C} \cap \Sigma^* = \{L \cap \Sigma^*; L \in \mathcal{C}\}$$

where $\mathcal{C}$ is a class of real languages. The classes P/poly and NP/poly are the non-uniform counterparts of P and NP (see e.g. [1] for more details). We also consider a model in which no order relation is available (hence equality is the only test allowed). In this case real machines are not more powerful than Turing machines for discrete inputs, i.e.,

- $\mathrm{P}^{=}_{\mathrm{add}} \cap \Sigma^* = \mathrm{P}$;
- $\mathrm{NP}^{=}_{\mathrm{add}} \cap \Sigma^* = \mathrm{NP}$.

The consideration of the real Traveling Salesman Problem or the real Knapsack Problem suggests a further restriction of the standard BSS model, this time on the type of non-determinism allowed. In the standard BSS model a guess is a sequence of real numbers, but the natural non-deterministic algorithms for these two problems make guesses that are of a combinatorial nature (a permutation of the sequence of cities or a subset of the set of objects). It is therefore quite natural to study *digital non-deterministic* (or *weak*) real machines whose guesses are sequence of 0's and 1's, like those of Turing machines. We prove that digital non-determinism is in fact equivalent to the standard real determinism. More precisely,

- $\mathrm{NDP}^{<}_{\mathrm{add}} = \mathrm{NP}^{<}_{\mathrm{add}}$;
- $\mathrm{NDP}^{=}_{\mathrm{add}} = \mathrm{NP}^{=}_{\mathrm{add}}$.

where NDP denotes weak non-deterministic classes.

Additive machines and weak non-determinism have already been studied in [3]. The weak BSS model [4] and linear machines [6] are two other related models. Their computational power lies in-between those of the standard BSS model and of additive machines. In section 2 we shall give a simpler proof of Meer's $\mathrm{P_{lin}} \neq \mathrm{NP_{lin}}$ result, from which it follows that $\mathrm{P^{=}_{add}} \neq \mathrm{NP^{=}_{add}}$.

The remainder of the paper is organized as follows. In section 2 we introduce a few general notations and study real machines without order. In section 3 we establish a few properties of polyhedra that are useful for studying real machines with order (a polyhedron is a subset of $\mathbb{R}^n$ defined by a system of large or strict linear inequalities having integral coefficients). In particular, we give two purely geometrical results:

- a non-empty polyhedron defined by a system of "small" inequalities contains a "small" rational point;
- A projection of a polyhedron defined by a system of "small" inequalities can be defined by a system of "small" inequalities.

Real machines with order are discussed in section 4.

2 Computing over the reals with addition and equality

Throughout the paper, $\alpha = (\alpha_1, \ldots, \alpha_p)$ is the vector of constants occurring in the program of a real machine (we assume without loss of generality that one of these constants is equal to 1).

The size $s(n)$ of an integer n is the number of digits of its radix-2 representation, and the size of a rational number $x = p/q$ is the minimum value of $s(p) + s(q)$ where the integers p and q are such $x = p/q$.

We assume w.l.o.g. that all tests are of the form "$x \stackrel{?}{=} 0$".

Theorem 1 is in fact a special case of a more general result on a model in which a moderate usage of multiplication and division is allowed [4]. The general result relies on Gröbner bases or the primitive element theorem, but here some very elementary linear algebra is sufficient.

Theorem 1 $\mathrm{P^{=}_{add}} \cap \Sigma^* = \mathrm{P}$.

Proof. Clearly $\mathrm{P} \subset \mathrm{P^{=}_{add}} \cap \Sigma^*$. We prove the converse below. At any time in the computation of real machine on an input $u \in \{0,1\}^+$, the content of a variable x is of the form

$$x = \sum_{i=1}^{p} a_i \alpha_i$$

where the a_i's are integers of polynomial size. Let E_α be the vector space over $\mathbb{Q}$ spanned by $\alpha_1, \ldots, \alpha_p$ and let m be the dimension of E_α. We can assume without loss of generality that $(\alpha_1, \ldots, \alpha_m)$ is a base of E_α. Let (b_{ij}) be the matrix of rational coefficients such that

$$\alpha_i = \sum_{j=1}^{m} b_{ij} \alpha_j \ (1 \leq i \leq p).$$

Since $x = \sum_{j=1}^{m} \left(\sum_{i=1}^{p} a_i b_{ij} \right) \alpha_j$, the test $x \stackrel{?}{=} 0$ can be performed formally by checking that $\sum_{i=1}^{p} a_i b_{ij} = 0$ for $1 \leq j \leq m$. $\square$

We now turn to the study of digital non-determinism.

Theorem 2 $\mathrm{NDP}^{=}_{\mathrm{add}} = \mathrm{NP}^{=}_{\mathrm{add}}$.

The proof of this result relies on the following lemma.

Lemma 1 *Let $\mathcal{S}$ be a system of (dis)equations on $y \in \mathbb{R}^n$ of the form*

$$\begin{cases} \sum_{j=1}^{n} a_{ij} y_j = b_i \ (1 \le i \le p) \\ \sum_{j=1}^{n} c_{ij} y_j \neq d_i \ (1 \le i \le q) \end{cases}$$

with $a_{ij}, c_{ij} \in \mathbb{N}$ and $b_i, d_i \in \mathbb{R}$. Let us define the size of $\mathcal{S}$ as the number of real constants in the system plus the number of bits needed to represent the integral constants. It is possible to decide in $\mathrm{P}^{=}_{\mathrm{add}}$ whether $\mathcal{S}$ is satisfiable.

Proof. We first solve the system of equalities by Gaussian elimination. If this system is not satisfiable, $\mathcal{S}$ is of course not satisfiable. If it is satisfiable, we obtain (up to a permutation of indices) a parametrization of the solution space of the form

$$y_i = e_i + \sum_{j=k+1}^{n} f_{ij} y_j \ (1 \le i \le k)$$

where $y_{k+1}, \ldots, y_n$ are free variables (possibly, $k = n$). Next we substitute this expression in the inequalities to look for a contradiction. If we obtain an inequality of the form $d_i \neq d_i$, $\mathcal{S}$ is of course not satisfiable. Otherwise $\mathcal{S}$ is satisfiable, since $\mathbb{R}^{n-k-1}$ cannot be equal to the union of a finite number of hyperplanes.

It remains to show that this algorithm runs in polynomial time in the additive BSS model. Note that there are no multiplications or divisions between real variables in this algorithm. It is easy to multiply a real number y by an integer $x = \overline{x_1 \ldots x_m}$ using a sequence of "shift-and-add" operations, e.g.,

```
product:=0;
for i:=1 to m do
   if x[i]=0 then product:=2*product
              else product:=2*product+y.
```

If we have to divide y by x, we simply represent the quotient by the pair (y, x). The overall running time will be polynomial if the integers occurring during the execution of the algorithm are of polynomial size. This property holds for the Gaussian elimination procedure ([7], Theorem 3.3). It is not difficult to see that there cannot be any blowup in integer size in the rest of the algorithm. □

Note that for Theorem 2 it would be sufficient to show that the satisfiability of $\mathcal{S}$ is in $\mathrm{NP}^{=}_{\mathrm{add}}$.

Proof of Theorem 2. Clearly $\mathrm{NDP}^{=}_{\mathrm{add}} \subset \mathrm{NP}^{=}_{\mathrm{add}}$: a digital guess $g \in \{0,1\}^{n^{O(1)}}$ can be computed by comparing to zero each component of a real guess $h \in \mathbb{R}^{n^{O(1)}}$. We prove the converse below. Let $\mathcal{M}$ be a non-deterministic machine recognizing $L \in \mathrm{NP}^{=}_{\mathrm{add}}$. We construct a digitally non-deterministic machine $\mathcal{M}'$ recognizing L in polynomial time. On an input $u \in \mathbb{R}^n$, this machine first guesses the results of all tests in the program. Since $\mathcal{M}$ runs in polynomial time, this needs to be done for a time $T(n)$ which is polynomial in n. Adding dummy instructions if necessary, we can assume that all computations of $\mathcal{M}$ halt exactly in time $T(n)$. During a computation the content of any variable is always a linear function of the constants α, the input $u \in \mathbb{R}^n$ and the real guess $y \in \mathbb{R}^{p(n)}$ of $\mathcal{M}$. Hence for a given sequence g of test results, the constants α, inputs x and real guesses y corresponding to accepting computations are defined by a system $\mathcal{S}_g(\alpha, x, y)$ of linear equations and disequations (these constraints ensure that the computation is accepting and that the results of all tests are given by g). For a fixed α, an input u is recognized if and only if

$$\exists g\ \exists y\ \mathcal{S}_g(\alpha, u, y) \text{ is satisfied.}$$

After g has been guessed, the existence of y such that $\mathcal{S}_g(\alpha, u, y)$ is satisfied can be tested by the algorithm of Lemma 1. This takes polynomial time since the integral constants in the system are of polynomial size. □

The non-deterministic counterpart of Theorem 1 is a simple consequence of Theorem 2.

Theorem 3 $\mathrm{NP}^{=}_{\mathrm{add}} \cap \Sigma^* = \mathrm{NP}$.

Proof. Clearly $\mathrm{NP} \subset \mathrm{NP}^{=}_{\mathrm{add}}$. For the conserve we just have to show that $\mathrm{NDP}^{=}_{\mathrm{add}} \subset \mathrm{NP}$. For a fixed digital guess, $\mathcal{M}$ behaves like a deterministic real machine. More precisely, deciding whether a pair $(u, y) \in \{0,1\}^n \times \{0,1\}^{n^{O(1)}}$ corresponds to an accepting computation of $\mathcal{M}$ is a $\mathrm{P}^{=}_{\mathrm{add}}$ problem. According to Theorem 1 this problem is actually in P, whence the result. □

In the rest of this section we turn our attention to the more powerful model of *linear machines*. A linear machine can perform computations of the form

$$y \leftarrow \sum_{i=1}^{m} c_i x_i + b,$$

where b and the c_i's are constants of the machine. Equality is still the only comparator allowed. Meer proved a P≠NP result for this model in [6] ($\mathrm{P}_{\mathrm{lin}} \neq \mathrm{NP}_{\mathrm{lin}}$ with his notations). We now give a simpler proof of his result.

Theorem 4 *The following problem L is in $\mathrm{NP}_{\mathrm{lin}}$ but not in $\mathrm{P}_{\mathrm{lin}}$: given $x \in \mathbb{R}^n$, is there $\epsilon \in \{0,1\}^n$ such that $\sum_{i=1}^n \epsilon_i x_i = 1$?*

Proof. The problem is clearly in $\mathrm{NP}_{\mathrm{lin}}$. Let us assume that there exists a linear machine $\mathcal{M}$ which recognizes L in polynomial time. All tests performed by $\mathcal{M}$ are of the form $l(x) \stackrel{?}{=} 0$, where l is an affine function of the input x. Since $\mathcal{M}$ can perform only a finite number of tests in finite time, there exists some input $x \notin L$ such that the answers to the tests $l_1(x) \stackrel{?}{=} 0, \ldots, l_m(x) \stackrel{?}{=} 0$ of $\mathcal{M}$ on this input are all negative. Since $\mathcal{M}$ runs in polynomial time, at least one of the 2^n tests of the form

$\sum_{i=1}^n \epsilon_i x_i \stackrel{?}{=} 1$ has not been performed. Consider such an ϵ and an input $y \in \mathbb{R}^n$ such that $\sum_{i=1}^n \epsilon_i y_i = 1, l_1(y) \neq 0, \ldots, l_1(y) \neq 0$ (y exists since the m last constraints remove only a $(n-2)$-dimensional subspace from the hyperplane satisfying the first constraint). The outcomes of all tests are the same for y as for x, hence $\mathcal{M}$ declares that $y \notin L$ since $x \notin L$. This is a contradiction, since in fact $y \in L$. $\square$

A similar result was proved by Meer for a multidimensional version of L. Since L is clearly in $\mathrm{NP}^{=}_{\mathrm{add}}$, it follows from Theorem 4 that $\mathrm{P}^{=}_{\mathrm{add}} \neq \mathrm{NP}^{=}_{\mathrm{add}}$.

3 Some basic facts on polyhedra

In this section we show that:

- a non-empty polyhedron defined by a system of "small" inequalities contains a "small" rational point (Theorem 6);
- The satisfiability problem for a system of linear inequalities is in $\mathrm{NP}^{<}_{\mathrm{add}}$.
- A projection of a polyhedron defined by a system of "small" inequalities can be defined by a system of "small" inequalities (Theorem 8).

The following linear programming result will be useful (see [7], corollary 7.1i).

Lemma 2 (Caratheodory's theorem) *If a vector $b \in \mathbb{R}^n$ is a positive linear combinations of vectors $a_1, \ldots, a_m \in \mathbb{R}^n$ (i.e., if there exist coefficients $x_i \geq 0$ such that $b = \sum_{i=1}^m x_i a_i$), then there exists a linearly independent subset $\{a_{i_1}, \ldots, a_{i_k}\} \subset \{a_1, \ldots, a_m\}$ such that b is a positive linear combination of $a_{i_1}, \ldots, a_{i_k}$.*

We first establish the existence of "small" points for the case of closed polyhedra. Theorem 5 is a well-known result; the proof that we give here is due to W. Maass [5] and its principle will be used later in this section.

Theorem 5 *Let P be a polyhedra of $\mathbb{R}^n$ defined by a system $Ax \leq b$ of N linear inequalities, where the entries of A and b are integers of size L. If $P \neq \emptyset$, there exists a rational point $x \in P$ of size polynomial in L and n (i.e., $s(x) = (Ln)^{O(1)}$).*

Proof. The original system can be transformed into a system of the form

$$A'x' = b, \; x' \geq 0 \tag{1}$$

by writing each variable x_i as the difference of two non-negative variables, and adding a "slack variable" to each inequality. In the $N \times (2n+N)$ matrix A', the N columns for the N slack variables of x' form an idendity matrix. Hence A' has rank N. By hypothesis, (1) has a solution. It follows from Caratheodory's theorem that there exists also a satisfiable system of the form

$$A''x'' = b, \; x'' \geq 0 \tag{2}$$

where A'' is a $N \times N$ matrix made of N linearly independent columns of A'. Since A'' has full rank, (2) has in fact a unique solution which is given by Cramer's rule:

$x''_j = \det(A''_j)/\det(A'')$ for $j = 1, \ldots, N$, where A''_j results from A'' by replacing its j^{th} column by b. Since all but at most $2n$ columns of A'' contain exactly one 1 and else only 0's, one can bring by permutations of rows and columns each of the matrices A'', A''_j into the form

$$B = \begin{pmatrix} C & 0 \\ D & I \end{pmatrix}$$

where C is a square matrix of size $2n+1$. Hence the determinant of B is an integer of absolute value smaller than $(2n+1)!2^{L(2n+1)}$. $\square$

Theorem 6 *Let P be a polyhedra of $\mathbb{R}^n$ defined by a system $\mathcal{S}$ of N strict or large inequalities of the form*

$$Ax \leq b;\ A'x < b'$$

where the entries of A, A', b and b' are integers of size L. If $P \neq \emptyset$, there exists a rational point $x \in P$ of size polynomial in L and n (i.e., $s(x) = (Ln)^{O(1)}$).

Proof. Let P' be the polyhedron of $\mathbb{R}^{n+1}$ defined by the system $\mathcal{S}'$ obtained from $\mathcal{S}$ by the following transformations:

- add a new variable K and the constraint $K \geq 1$;
- replace the subsystem $Ax \leq b$ by $Ay \leq Kb$;
- replace the subsystem $Ax < b$ by $Ay \leq Kb - 1$.

Clearly $P \neq \emptyset$ if and only if $P' \neq \emptyset$. If $P \neq \emptyset$, P' contains a small point (K, y) according to Theorem 5, and the transformation $x_i = y_i/K$ yields a small point in P. $\square$

Note that the number N of inequations in the system does not come into play in the bound. This point is crucial for the applications of this result in sections 2 and 4.

Theorem 7 *Let $\mathcal{S}$ be a system of N inequations on $y \in \mathbb{R}^n$ of the form*

$$Ax \leq b;\ A'x < b'$$

with $a_{ij}, a'_{ij} \in \mathbb{N}$ and $b_i, b'_i \in \mathbb{R}$. Let us define the size of $\mathcal{S}$ as the number of real constants in the system plus the number of bits needed to represent the integral constants. It is possible to decide in $\mathrm{NDP}^{<}_{\mathrm{add}}$ whether $\mathcal{S}$ is satisfiable.

Proof. We transform $\mathcal{S}$ into a new system $\mathcal{S}'$ of the form $Cz \leq d$, like in the proof of Theorem 6 ($z = (K, y)$ with $K \in \mathbb{R}$ and $y \in \mathbb{R}^n$). As we noticed before, $\mathcal{S}$ is satisfiable if and only if $\mathcal{S}'$ is satisfiable. The satisfiability of $\mathcal{S}'$ can be checked with the proof technique of Theorem 5: we transform $\mathcal{S}'$ into an equivalent system of the form

$$C'z' = d,\ z' \geq 0 \tag{3}$$

by adding a slack variable to each inequality and writing each variable y_i as the difference of two non-negative variables (this needs not be done for the variable K since the constraint $K \geq 1$ already appears in the system). In the $N \times (N + 2n + 1)$ matrix C', the N columns for the N slack variables of z' form an idendity matrix. Hence C'

has rank N. It follows from Caratheodory's theorem that (3) is satisfiable if and only if there exists also a satisfiable system of the form

$$C''z'' = d, \; z'' \geq 0 \tag{4}$$

where C'' is a $N \times N$ matrix made of N linearly independent columns of C'. Since C'' has full rank, (4) has in fact a unique solution which is given by Cramer's rule: $z''_j = \det(C''_j)/\det(C'')$ for $j = 1, \ldots, N$, where C''_j results from C'' by replacing its j^{th} column by d. The overall algorithm is therefore quite simple:

1. guess non-deterministically a $N \times N$ submatrix C'' of C';

2. check that $\det(C'') \neq 0$;

3. check that $z''_j = \det(C''_j)/\det(C'') \geq 0$, for $j = 1, \ldots, N$.

It remains to show that this can be done in polynomial time in the additive BSS model. Obviously the last step does not actually require a division: it suffices to check that $\mathrm{sign}(\det(C''_j)).\mathrm{sign}(\det(C'')) \geq 0$. Computing the determinants is a little more difficult due to the presence of real numbers in the system. Note however that these real numbers b_i occur in a single column of C'' (and in at most one column of C''_j). In order to compute $\det(C'')$ (or $\det(C''_j)$), it is thus possible to perform an expansion along this column. The N (integral) subdeterminants can be computed by Gaussian elimination and the N integer$\times$real products by shift-and-add operations. This takes polynomial time (see Lemma 1). □

The methods of theorems 5 and 7 can be used to prove the following result: a projection of a polyhedron defined by "small" inequations can be described by small inequations. We give this result here because of its interest from a geometrical and logical point of view, but it will not be used in the rest of the paper. For the special case of closed polyhedra, this result was obtained previously in [8] by different methods.

Theorem 8 *Let P be a polyhedra of $\mathbb{R}^n$ defined by a system of inequalities of the form*

$$Ax \leq b; \; A'x < b'$$

where the entries of A, A', b and b' are integers of size L. The image of P by any projection of the form $(x_1, \ldots, x_m, \ldots, x_n) \mapsto (x_1, \ldots, x_m)$ is a polyhedron of $\mathbb{R}^m$ defined by a system of inequalities whose coefficients are of size $(Ln)^{O(1)}$.

Proof Sketch. We will not obtain directly a system of inequalities describing the projected polyhedron P', but a union of systems (this amounts to giving a polyhedral decomposition of P'). A single system for P' can be obtained by picking some inequalities in each system of the union.

By definition a point $y = (x_1, \ldots, x_m)$ belongs to P' iff there exists a point $z = (x_{m+1}, \ldots, x_n)$ such that $(y, z) \in P$, or in other words if the polyhedra of $\mathbb{R}^{n-m}$ defined by the system

$$Bz \leq \beta(y); \; B'z < \beta'(y) \tag{5}$$

is non-empty. Here B and B' are the matrices made of the $n-m$ last columns of A and A', $\beta_i(y) = b_i - \sum_{j=1}^{m} a_{ij}x_j$ and $\beta'_i(y) = b'_i - \sum_{j=1}^{m} a'_{ij}x_j$. Let us apply the transformation of Theorem 7: this condition is equivalent to the existence of a constant $K > 0$ such that a certain $(n-m)$-dimensional system $\mathcal{S}$ of the form

$$\sum_{j=1}^{n-m} b_{ij} u_j \leq K\beta_i(y) - \epsilon_i$$

is satisfiable ($\epsilon_i \in \{0,1\}$). The proof technique of Theorem 5 shows that the satisfiability of $\mathcal{S}$ is equivalent to the satisfiability of a least one of a finite number of systems of the form

$$K l_i(y) \geq c_i, \ i = 1, \ldots, N \tag{6}$$

where l_i is an affine function. The permutation argument of Theorem 5 shows that the coefficients of such a system are of polynomial size (in n and L). It is not difficult to see that the set of y's such that (6) is satisfied for some $K > 0$ can be described by a finite set of linear systems (whose coefficients have the required size). The details are left to the reader. □

From a logical point of view, the fact that projections of polyhedra are still polyhedra can be restated as "an existential formula in the theory of the reals with addition and order is equivalent to a quantifier-free formula". Theorem 8 asserts that this quantifier elimination does not produce any blowup in coefficient size.

4 Computing over the reals with addition and order

The following theorem is a special case of the $\mathrm{P}_{WRD} = \mathrm{P/poly}$ result of [4]. Here we can give a simpler proof than in the general case.

Theorem 9 $\mathrm{P}^{<}_{\mathrm{add}} \cap \Sigma^* = \mathrm{P/poly}$.

Proof. In order to show that $\mathrm{P/poly} \subset \mathrm{P}^{<}_{\mathrm{add}} \cap \Sigma^*$, we encode an advice function in the digits a_i of a real constant $\alpha = \overline{0.a_1 a_2 \ldots a_i \ldots}$. Let us choose an encoding scheme such that infinitely many a_i's are equal to 0. The value of a_1 can be easily computed since $a_1 = 0 \Leftrightarrow \alpha < 1/2$. In order to compute a_2, we shift α to the left with the instruction $\alpha := 2\alpha - a_1$. Now a_2 can be computed like a_1. By iterating this process, it is possible to retrieve as many digits of c as desired. Since the advice are of polynomial size, the whole decoding process takes polynomial time.

In order to prove the converse, we replace for each input size n the real constants α by rational approximations $\overline{\alpha}$ such that the outcomes of all tests in the program are unchanged. This constraint gives a system of (exponentially many) linear equalities and inequalities with coefficients of polynomial size. According to Theorem 6, there exists a solution $\overline{\alpha}$ of polynomial size. □

Theorem 10 $\mathrm{NP}^{<}_{\mathrm{add}} = \mathrm{NDP}^{<}_{\mathrm{add}}$.

Proof. The proof is identical to that of Theorem 2, except that we obtain a system $\mathcal{S}_g$ of linear equalities and inequalities. The satisfiability of this system can be tested with the non-deterministic algorithm of Theorem 7. □

Theorem 11 $\mathrm{NP}^<_{\mathrm{add}} \cap \Sigma^* = \mathrm{NP/poly}$.

Proof. Clearly $\mathrm{NP/poly} \subset \mathrm{NP}^<_{\mathrm{add}} \cap \Sigma^*$ (see the proofs of theorems 3 and 9). According to Theorem 10, it suffices to show that $\mathrm{NDP}^<_{\mathrm{add}} \cap \Sigma^* \subset \mathrm{NP/poly}$ to prove the converse. For a fixed digital guess, $\mathcal{M}$ behaves like a deterministic real machine. More precisely, deciding whether a pair $(u, y) \in \{0,1\}^n \times \{0,1\}^{n^{O(1)}}$ corresponds to an accepting computation of $\mathcal{M}$ is a $\mathrm{P}^<_{\mathrm{add}}$ problem. According to Theorem 9 this problem is actually in P/poly, whence the result. $\square$

Acknowledgment

I would like to thank Bruno Poizat for several useful suggestions.

References

[1] J.L. Balcázar, J. Diáz, and J. Gabarró. *Structural Complexity I.* EATCS Monographs on Theoretical Computer Science. Springer-Verlag, 1988.

[2] L. Blum, M. Shub, and 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(1):1–46, July 1989.

[3] F Cucker and M. Matamala. On weak nondeterminism. preprint, 1993.

[4] P. Koiran. A weak version of the Blum, Shub & Smale model. In *Proc.* 34^{th} *FOCS Symp.*, 1993. to appear.

[5] W. Maass. Bounds for the computational power and learning complexity of analog neural nets. In *Proc.* 25^{th} *STOC*, pages 335–344, 1993.

[6] K. Meer. A note on a $P \neq NP$ result for a restricted class of real machines. *Journal of Complexity*, 8:451–453, 1992.

[7] A. Schrijver. *Theory of Linear and Integer Programming.* Wiley, New-York, 1986.

[8] E. D. Sontag. Real addition and the polynomial hierarchy. *Information Processing Letters*, 20:115–120, 1985.