δ-uniform BSS Machines

Paolo Boldi* Sebastiano Vigna*
Dipartimento di Scienze dell'Informazione, Università degli Studi di Milano, Italy.

Abstract

A δ-uniform BSS machine is a standard BSS machine which does not rely on exact equality tests. We prove that, for any real closed archimedean field R, a set is δ-uniformly semi-decidable iff it is open and semi-decidable by a BSS machine which is locally time bounded; we also prove that the local time bound condition is nontrivial. This entails a number of results about BSS machines, in particular the existence of decidable sets whose interior (closure) is not even semi-decidable without adding constants. Finally, we show that the sets semi-decidable by Turing machines are the sets semi-decidable by δ-uniform machines with coefficients in $\mathbf{Q}$ or $\mathbf{T}$, the field of Turing computable numbers.

1 Introduction

The problem of extending classical recursion theory to the non-discrete world of real numbers has given rise to two complementary approaches: following the tradition of Turing, one can extend the notion of Turing machine by allowing input and output tape to contain (infinite) representations of real numbers; this approach is known as *Type 2 recursion theory* [18]. On the other hand, it is possible to consider the reals as basic atomic entities, on which exact computations and tests are permitted, as in the *BSS model* [1].

This paper focuses on the problem of (semi-)deciding subsets of the reals, giving a comparison between Turing, BSS, and δ*-uniform* decidability, the latter being a restriction of the BSS model in which machines cannot rely on exact tests. We prove that the essential gap between the standard and the δ-uniform BSS case is given by a topological condition on the semi-decidable sets (which must be open) and by a local bound on the accepting times; the gap from the δ-uniform case to the Turing model is determined by the presence of constants which are not computable in the sense of Turing.

Our motivations resemble the ones which led to the definition of *feasible real random access machines* [4]: however, in that case one had to introduce nondeterminism in the computation of a deterministic machine, which we would like to avoid. Moreover, since we are mainly interested in *decidability* questions (rather than function approximation problems), avoidance of nondeterminism allows us to use classical tools such as quantifier elimination on register equations.

*The authors have been partially supported by the Esprit Working Group No. 8556 (NeuroCOLT).

The paper is structured as follows: in Section 2 we succintly recall the models used in the rest of the paper, and introduce the δ-uniformity restriction on BSS machines. Section 3 is devoted to a survey of the results we need from field extension theory and topology. Then, we compare the behaviour δ-uniform BSS machines and Turing machines on archimedean fields, and finally we prove our main theorem, stating that, over real closed archimedean fields, a locally time bounded BSS machine semi-deciding an open set can be emulated by a δ-uniform machine (and thus, essentially, by a Turing machine). The last section is devoted to the proof of the existence of open sets which cannot be semi-decided with a local time bound; the techniques are then extended in order to produce BSS decidable sets whose interior (closure) is not even semi-decidable without adding constants.

2 Computational models

In this section we review briefly the models which will be used in the rest of the paper; moreover, δ-uniformity, a restriction on BSS machines, is introduced. Unless otherwise stated, the field R is always intended to be archimedean (i.e., a subfield of **R**—see Section 3).

2.1 The finite dimensional BSS model

A *finite dimensional machine* M over an ordered ring (or field) R consists of three spaces: the input space $\bar{I} = R^l$, the output space $\bar{O} = R^m$ and the state space $\bar{S} = R^n$, together with a finite directed connected graph with node set $\bar{N} = \{1, 2, \ldots, N\}$ ($N > 1$) divided into four subsets: *input*, *computation*, *branch* and *output* nodes.

Node 1 is the only *input node*, having fan-in 0 and fan-out[1] 1; node N is the only *output node*, having fan-out 0. They are endowed with linear functions with integer coefficients (named $I(-)$ and $O(-)$), mapping respectively the input space to the state space and the state space to the output space. Any other node $q \in \{2, 3, \ldots, N-1\}$ can be of the following types:

1. a *computation node*; in this case, q has fan-out 1 and there is a polynomial (or rational, if R is a field) function $g_q : \bar{S} \to \bar{S}$ associated with it;

2. a *branching node*; in this case, q has fan-out 2 and its two (distinguished) successors are $\beta^-(q)$ and $\beta^+(q)$; branching on $-$ or $+$ will depend upon whether or not the first coordinate of the state space is negative[2].

We can view M as a discrete dynamical system over the *full state space* $\bar{N} \times \bar{S}$. M induces

[1] If q is a node with fan-out 1, then $\beta(q)$ denotes the "next" node in the graph after q.

[2] Note that usually a test with a polynomial is assumed, but the present restriction can be made without loss of generality [1].

a *computing endomorphism* on the full state space:

$$\langle 1, \boldsymbol{x}\rangle \mapsto \langle \beta(1), \boldsymbol{x}\rangle \tag{1}$$
$$\langle N, \boldsymbol{x}\rangle \mapsto \langle N, \boldsymbol{x}\rangle \tag{2}$$
$$\langle q, \boldsymbol{x}\rangle \mapsto \langle \beta(q), g_q(\boldsymbol{x})\rangle \quad \text{if } q \text{ is a computation node} \tag{3}$$
$$\langle q, \boldsymbol{x}\rangle \mapsto \begin{cases} \langle \beta^-(q), \boldsymbol{x}\rangle & \text{if } x_1 < 0 \\ \langle \beta^+(q), \boldsymbol{x}\rangle & \text{if } x_1 \geq 0 \end{cases} \quad \text{if } q \text{ is a branching node.} \tag{4}$$

The *computation* of M under input $\boldsymbol{a}$ is the orbit generated in the full state space by the computing endomorphism starting from $\langle 1, I(\boldsymbol{a})\rangle$. If the orbit reaches a fixed point of the form $\langle N, \boldsymbol{b}\rangle$ for some $\boldsymbol{b} \in \bar{S}$ we say that the machine *halted*, and that its output is $O(\boldsymbol{b})$. The set of all inputs on which M halts is called the *halting set* of M, and it is denoted by Ω_M; the association $\boldsymbol{a} \mapsto O(\boldsymbol{b})$ defines a partial function φ_M, which is called the *partial function computed by the machine M*.

A set which is the halting set of some BSS machine is called *semi-decidable*; if moreover its complement is also semi-decidable, we shall say that the set is *decidable*. Similarly, a partial function is *computable* if it is computed by some BSS machine. A set A is *semi-decidable relative to B* if $A \cap B$ is semi-decidable. It is *decidable relative to B* if both $A \cap B$ and $A^c \cap B$ are semi-decidable.

2.2 δ-uniformity

In a BSS machine, the branching at a node q is decided using the signum of the first coordinate x_1 of $\bar{S}$; in a δ-uniform machine, the test is essentially replaced by $x_1 \geq -\delta$, where δ is not known. Thus, a successful nonnegativity test just claims that the argument was positive or in a neighbourhood of 0, while successful negativity test implies that the argument is strictly negative.

Formally, given a BSS machine M and a $\delta \geq 0$ (called a *threshold*), we define the δ-computing endomorphism much in the same way as we did before[3], but substituting the case 4 as follows:

$$\langle q, \boldsymbol{x}\rangle \mapsto \begin{cases} \langle \beta^-(q), \boldsymbol{x}\rangle & \text{if } x_1 < -\delta \\ \langle \beta^+(q), \boldsymbol{x}\rangle & \text{if } x_1 \geq -\delta \end{cases} \quad \text{if } q \text{ is a branching node.} \tag{5}$$

For every $\delta \geq 0$, this induces, as before, a δ-halting set (denoted by Ω_M^δ) and a δ-computed function φ_M^δ.

Definition 1 M is δ-uniform if and only if $\Omega_M^\delta = \Omega_M$ and $\varphi_M^\delta = \varphi_M$ for all $\delta \in (0, 1)$.

The definition of δ-uniformity is the mathematical formalization of the fact that the threshold is not known to the programmer. The notions of (semi-)decidable set and of computable function

[3] In the BSS model one assumes that a machine never performs a division by zero; here, we correspondingly assume that no division by zero is performed during a δ-computation. Equivalently, one can assume that division by zero causes divergence.

carry over to the δ-uniform case[4]. Note that every δ-uniformly semi-decidable set is *a fortiori* BSS semi-decidable (analogously for computability of functions), and that $\Omega_M = \Omega^0_M$ and $\varphi_M = \varphi^0_M$.

2.3 Type 2 Turing machines

Since any archimedean field is isomorphic to a subfield of the reals, its elements are approximable by converging sequences of rationals (by density of **Q**), and its operations are approximable using rational approximations of the arguments.

In particular, without loss of generality we can restrict our attention to sequences of dyadic numbers converging exponentially fast, or, again without loss of generality, to the *signed binary digit* representation. Such a representation is given by an infinite string $s \in \{\bar{1}, 0, 1, .\}^\omega$ of the form

$$s = b_n b_{n-1} \cdots b_0 . b_{-1} b_{-2} \cdots ,$$

where we assume that $b_n \neq 0$ and that the part on the left of the dot does not start with $1\bar{1}$ or $\bar{1}1$. The number $\bar{s}$ represented by s is defined by

$$\bar{s} = \sum_{i=n}^{-\infty} b_i 2^i ,$$

where the symbol $\bar{1}$ has value -1 (of course, not all representations will correspond to elements of R unless $R = \mathbf{R}$). This is occasionally generalized to finite sequences, by interpreting a finite sequence s as $s0^\omega$ (or $s.0^\omega$, if s does not contain a dot).

For several reasons [10, 9], this representation is particularly suitable for Turing machines, and will be used in order to represent elements of an archimedean field R as infinite sequences of symbols (to be given as a generalized input to a Turing machine).

The tape of an ordinary Turing machine is nonblank only on a finite number of cells, at any moment of a computation. Thus, in order to allow elements of R to be taken into consideration, one slightly generalizes the notion of machine. A (deterministic) Type 2 Turing machine [18] consists of

1. a finite number of read-only one-way input tapes (possibly none), each containing at start an infinite string belonging to $\{\bar{1}, 0, 1, .\}^\omega$ and representing an element of R;

2. a finite number of write-only one-way output tapes (possibly none), on which the machine is supposed to write representations of elements of R;

3. some other work tapes, initially blank.

[4]The choice of the interval $(0, 1)$ is immaterial in this definition: it is easy to see that a set (function) which is δ-uniformly semi-decidable (computable) w.r.t. $(0, 1)$ is also δ-uniformly semi-decidable (computable) w.r.t. the whole set of positives.

The finite control is defined as usual via a finite set of states and a transition function. The only differences with a standard Turing machine are the possibility of filling completely the input tapes, and of considering nonstopping machines as machines outputting elements of R.

A set $X \subseteq R^n$ is (Type 2) Turing semi-decidable iff there is a Type 2 Turing machine M with n input tapes which stops iff the input tapes are filled with signed binary digit representations of the coordinates of an $\boldsymbol{a} \in X$. Note that our definition implies that the halting does not depend on the particular representations chosen; the definition of relative semi-decidability follows as in the classical case.

In the rest of the paper, we shall often deal with input tapes of a Type 2 Turing machine which are guaranteed not to contain dyadic numbers (i.e., numbers of the form $j/2^k$ with $j, k \in \mathbf{Z}$). It is known that no Turing machine can convert signed into positive digit representations [5], but in the special case of nondyadic numbers we can safely assume that the machine can internally produce a positive representation of the same number. This is due to the following

Proposition 1 There is a Type 2 Turing machine which outputs the positive representation of its input, provided that it is non-dyadic.

Proof. We show that for every $a \in (-1, 1)$ which is not a dyadic (i.e., of the form $j/2^k$ with $-2^k < j < 2^k$) there is a Type 2 Turing machine reading a signed binary representation of a and outputting a binary representation of a.

We can assume without loss of generality that $a \in (0, 1)$, because otherwise we could detect in finite time whether $a < 0$ (since $a \neq 0$), output '$-0.$' and work on $-a$ (just exchange 1 and $\bar{1}$ in the representation of a).

Recall that a finite sequence s of signed binary digits (without integer part) determines a dyadic interval $[\bar{s} - 2^{-|s|}, \bar{s} + 2^{-|s|}]$, while a sequence s' of binary digits determines a dyadic interval $[\bar{s}', \bar{s}' + 2^{-|s'|}]$; all real numbers having a representation starting with s (or s') lie in the corresponding interval. This implies that the following Turing machine (where we assumed a primitive **input** which returns the next symbol on the input tape) is correct:

Turing machine converting signed to positive binary representation of non-dyadics;
 /* Reads a signed binary digit representation of a non-dyadic $a \in (0, 1)$ and
 outputs the fractional part of a binary representation of a. */
var $l, r : \mathbf{Q}$;
 s : **string of** $\{\bar{1}, 0, 1, .\}$;
begin
 $l \leftarrow 0$;
 $r \leftarrow 1$;
 $s \leftarrow$ **input**;
 forever
  if $\left(\left[\bar{s} - \frac{1}{2^{|s|}}, \bar{s} + \frac{1}{2^{|s|}}\right] \subseteq \left[l, \frac{l+r}{2}\right]\right)$ **do**
   output(0);
   $r \leftarrow \frac{l+r}{2}$
  od
  elseif $\left(\left[\bar{s} - \frac{1}{2^{|s|}}, \bar{s} + \frac{1}{2^{|s|}}\right] \subseteq \left[\frac{l+r}{2}, r\right]\right)$ **do**
   output(1);

```
      l ← (l+r)/2
    od
    else s ← s + input
  loop
end ▮
```

Note that the previous proposition does *not* state that there is a Type 2 Turing machine transforming a signed binary representation into a positive (binary) representation; in fact, the machine of Proposition 1 does not even compute a function in the sense of Type 2 computability, because its behaviour on the dyadics is dependent on the representation (however, it does compute the identity function when restricted to non-dyadic inputs).

Finally, we recall a known result: a set $X \subseteq \mathbf{R}^n$ is Turing semi-decidable iff there are recursively enumerable sequences of rationals r_k and rational vectors $\boldsymbol{a}_k$ such that $X = \bigcup_{k\in\mathbf{N}} B_{r_k}(\boldsymbol{a}_k)$ [9]. This equivalence is known to be true for $\mathbf{R}$, but it is easy to check that it does not depend on the completeness axiom, and is true in every archimedean field; moreover, it can be relativized to any oracle.

3 Algebraic and topological preliminaries

In this section we gather some definitions and properties which shall be used frequently in the sequel. The algebraic results quoted here can be found in [11] and [17].

Algebraic extensions. Let F be a subfield of E (i.e., E is an *extension* of F), and let $a \in E$. We say that a is *algebraic* over F if there exists a non-zero polynomial $p(x) \in F[x]$ such that $p(a) = 0$, *transcendental* otherwise; if every element of E is algebraic over F, we say that E is an *algebraic extension* of F.

Real closed fields. A field R is *(formally) real* if -1 is not a sum of squares. It is *real closed* if it is real but has no (proper) real algebraic extensions. A real closed field has unique ordering, the positive elements in this ordering being precisely the squares.

Representation of finitely generated extensions. Let $F \subseteq E$ be an extension, and $\alpha_1, \alpha_2, \ldots, \alpha_r \in E$. The (finitely generated) extension $F \subseteq F(\alpha_1, \ldots, \alpha_r)$ is the smallest subfield of E containing F and $\alpha_1, \ldots, \alpha_r$. In the rest of the paper, we shall deal with the case $F = \mathbf{Q}$ and $E \subseteq \mathbf{R}$.

We can assume without loss of generality that $\alpha_1, \ldots, \alpha_s$, with $s \leq r$, is a *transcendence base* for $\mathbf{Q}(\alpha_1, \ldots, \alpha_r)$, i.e., there is no nonzero polynomial in s variables and coefficients in $\mathbf{Q}$ which vanishes when evaluated over $\alpha_1, \ldots, \alpha_s$, and moreover the extension $\mathbf{Q}(\alpha_1, \ldots, \alpha_s) \subseteq \mathbf{Q}(\alpha_1, \ldots, \alpha_r)$ is algebraic; it holds that $\mathbf{Q}(\alpha_1, \ldots, \alpha_s) \cong \mathbf{Q}(x_1, \ldots, x_s)$, where the latter expression denotes the fields of rational functions with s arguments and coefficients in $\mathbf{Q}$.

Now, the *primitive element theorem*[5] states that there is an $\alpha \in \mathbf{Q}(\alpha_1, \ldots, \alpha_r)$, algebraic over $\mathbf{Q}(\alpha_1, \ldots, \alpha_s)$, such that $\mathbf{Q}(\alpha_1, \ldots, \alpha_r) = \mathbf{Q}(\alpha_1, \ldots, \alpha_s)(\alpha)$. But every *simple* alge-

[5] Since $\mathbf{R}$ has characteristic zero, it is perfect; thus, all finite extensions considered here are separable, and the primitive element theorem can always be applied.

braic extension, i.e., every algebraic extension $E \subseteq E(\alpha)$ induces a surjective homomorphism[6] $E[x] \to E[\alpha]$, given by the evaluation of x to α. The kernel of this homomorphism is an ideal, generated by an irreducible polynomial $p(x) \in E[x]$, which can be assumed monic without loss of generality, called the *minimum polynomial of* α. The important consequence is that $E[x]/\langle p(x)\rangle \cong E[\alpha]$; moreover, $E[\alpha]$ is a field, and it is thus equal to $E(\alpha)$.

By combining the two above observations, we have that

$$\mathbf{Q}(\alpha_1, \ldots, \alpha_r) \cong \mathbf{Q}(x_1, \ldots, x_s)[x]/\langle p(x)\rangle.$$

Thus, every α_i has a “coding” as an element of $\mathbf{Q}(x_1, \ldots, x_s)[x]/\langle p(x)\rangle$, given by this isomorphism; moreover, all field operations of $\mathbf{Q}(\alpha_1, \ldots, \alpha_r)$ can be performed symbolically in $\mathbf{Q}(x_1, \ldots, x_s)[x]/\langle p(x)\rangle$, as well as equality tests (by using the standard polynomial operations and Euclid’s algorithm); of course, this is not true of order comparisons.

A topology on the affine space R^n. The affine space R^n admits a “norm” $\|\cdot\| : R^n \to R$, defined[7] by

$$\|\langle a_1, a_2, \ldots, a_n\rangle\| = \sum_{i=1}^{n} |a_i|,$$

which induces a “metric” $\rho : R^n \times R^n \to R$, inducing in turn a topology by taking as base the open balls

$$B_\varepsilon(\boldsymbol{a}) = \{\boldsymbol{b} \mid \|\boldsymbol{a} - \boldsymbol{b}\| < \varepsilon\}$$

when $\boldsymbol{a}$ ranges over R^n and ε is a positive element of R. Note that we can equivalently let ε range over $\mathbf{Q}$ and $\boldsymbol{a}$ range over $\mathbf{Q}^n$, because $\mathbf{Q}$ is dense in R (recall that R is archimedean).

There is an equivalent way of defining this topology: the intervals $(a, b) = \{c \in R : a < c < b\}$ form a base for a topology on R, called the *interval topology* of R. Then, the standard product topology on R^n coincides with the topology we just described. In the sequel we shall always understand that R^n is endowed with this topology.

Lemma 1 R (and thus also R^n) is a regular Hausdorff space. Moreover, R^n is connected iff R is the field of the reals.

Proof. The first part is Exercise 3.2.II of [7]. The second part follows remembering that a product space is connected iff all its factors are connected, and that the only ordered archimedean order-complete field is $\mathbf{R}$. ∎

Lemma 2 The polynomial and rational functions are continuous (when defined).

[6] If $F \subseteq E$ is an extension and $\alpha \in E$, then $F[\alpha]$ denotes the ring obtained by evaluating in α the polynomials of $F[x]$.

[7] Note that the values of the norm are taken from R rather than from the reals; the more familiar Euclidean norm, which could be defined only in the case of R real closed, is equivalent to the ℓ^1 norm used here.

Lemma 3 Let $\boldsymbol{a} \in R^n$, and let $B_{\varepsilon_0}(\boldsymbol{r}_0), B_{\varepsilon_1}(\boldsymbol{r}_1), \ldots$ be a sequence of balls with rational center $\boldsymbol{r}_i$ and rational radius $\varepsilon_i \to 0$ such that $\boldsymbol{a} \in B_{\varepsilon_i}(\boldsymbol{r}_i)$ for all i; then, for any set $X \subseteq R^n$, $\boldsymbol{a} \in X^\circ$ iff there is a K such that for all $k \geq K$ we have $B_{\varepsilon_k}(\boldsymbol{r}_k) \subseteq X$.

Proof. If $\boldsymbol{a} \in X^\circ$ then there is a ball $B_\varepsilon(\boldsymbol{a}) \subseteq X$. By taking K big enough so that $\varepsilon_k < \frac{\varepsilon}{2}$ for all $k \geq K$ we obtain half of the claim. On the other hand, if $B_{\varepsilon_k}(\boldsymbol{r}_k) \subseteq X$, then $\boldsymbol{a}$ is trivially in X°. ∎

Note that if R is not archimedean, the above lemma is false (simply because $\mathbf{Q}$ is not dense in R in that case).

4 Type 2 Turing machines vs. δ-uniform BSS machines

In this section we give a series of results about δ-uniform and Type 2 machines which culminate in Theorem 2. Firstly we study the structure of δ-uniformly semi-decidable sets.

Theorem 1 A δ-uniformly semi-decidable set is open.

Proof. Let M be δ-uniform, $\boldsymbol{a} \in \Omega_M$, and consider the accepting computation path of $\boldsymbol{a}$. At each branch a certain polynomial in $\boldsymbol{a}$ is tested against 0; let $p_1, p_2, \ldots, p_i$ be the polynomials evaluated negatively, and $p_{i+1}, p_{i+2}, \ldots, p_j$ be the polynomials evaluated nonnegatively. Then we have

$$-\varepsilon = \max\{-1, p_1(\boldsymbol{a}), \ldots\ p_i(\boldsymbol{a})\} < 0 \leq \min\{p_{i+1}(\boldsymbol{a}), \ldots\ p_j(\boldsymbol{a})\},$$

and by continuitity and finiteness of the polynomials there is an open neighbourhood U of $\boldsymbol{a}$ such that for all $\boldsymbol{b} \in U$ we have

$$\max\{p_1(\boldsymbol{b}), \ldots\ p_i(\boldsymbol{b})\} < -\frac{\varepsilon}{2} < \min\{p_{i+1}(\boldsymbol{b}), \ldots\ p_j(\boldsymbol{b})\},$$

This means that the machine M, with threshold $\frac{\varepsilon}{2}$, would accept an entire open neighbourhood of $\boldsymbol{a}$. But $\Omega_M = \Omega_M^{\frac{\varepsilon}{2}}$. ∎

Note that the previous proof actually tells us more: if $\boldsymbol{a}$ is accepted by M along a certain accepting path, then for some δ there is a whole ball centered in $\boldsymbol{a}$ which is accepted *along the same* accepting path, and thus with the same accepting time.

Since $\mathbf{R}^n$ is connected, we obtain that

Corollary 1 The only δ-uniformly decidable sets of reals are $\mathbf{R}^n$ and $\varnothing$.

On the other hand, there are, for example, δ-uniformly decidable subsets of the real algebraic numbers, such as $\{a \mid a < \pi\}$.

We now state a couple of lemmata which show how to decide certain sets relatively to an input subset, and how to compute certain functions.

Lemma 4 The set $\{\langle a, b\rangle \mid a < b\}$ is δ-uniformly decidable relative to $\{\langle a, b\rangle \mid a \neq b\}$.

Proof. Since R is archimedean, given any $a \neq b$, for every $\delta \in [0, 1)$ there is an integer k such that either $k(a - b) > \delta$ or $k(a - b) < -\delta$. Thus, the following subroutine

```
subroutine > (x : R, y : R);
   /* Returns 1 if x > y, 0 if x < y. */
var k : integer;
begin
   k ← 1;
   forever
      if k(x − y) > 0 return(1);
      if k(x − y) < 0 return(0);
      k ← k + 1
   loop
end
```

will decide the set $\{\langle a, b\rangle \mid a < b\}$. ∎

In particular, this means that for registers which are integer it is decidable whether they are equal or not, and which is the greater. As a consequence, a δ-uniform machine can perform any discrete computation (using just three registers).

Lemma 5 Consider a subroutine of a BSS machine defined as follows:

```
subroutine [·](x : R);
   /* Returns an integer approximating x. */
var k : integer;
begin
   k ← 0;
   forever
      if (k > x) and (k − 1 ≤ x) return(k);
      if (−k > x) and (−k − 1 ≤ x) return(−k);
      k ← k + 1
   loop
end
```

Then, the δ-computed function $[\cdot]_\delta$ is total and satisfies

$$a + \delta < [a]_\delta \le a + \delta + 1.$$

In particular, we have that for any integer K

$$\frac{[aK]_\delta}{K} - a \in \left(\frac{\delta}{K}, \frac{\delta + 1}{K}\right].$$

Proof. The only nonobvious part of the claim is termination. Note that, for each k, the first conditional statement will be taken on the interval $[k - 1 - \delta, k - \delta)$, while the second one will be taken on the interval $[-k - 1 - \delta, -k - \delta)$. When k ranges from 0 to ∞, by the archimedean property, the union of the above intervals covers R. ∎

Lemma 6 There is a subroutine which on input $\langle a, K\rangle$ returns a rational approximation r of a such that $0 < r - a < \frac{2}{K}$. There is a subroutine which on input $\langle \boldsymbol{a}, K\rangle$ returns a rational approximation $\boldsymbol{r}$ of $\boldsymbol{a}$ such that $0 < \rho(\boldsymbol{a}, \boldsymbol{r}) < \frac{2}{K}$.

Proof. By the previous lemma we have

$$0 < \frac{\delta}{K} < \frac{[aK]_\delta}{K} - a \leq \frac{\delta + 1}{K} < \frac{2}{K}.$$

The second part of the claim is an easy computation on a coordinatewise nK-approximation $\boldsymbol{r}$ of $\boldsymbol{a}$:

$$0 < \sum_{i=1}^{n} |r_i - a_i| = \sum_{i=1}^{n} \left| \frac{[a_i nK]_\delta}{nK} - a_i \right| < n \left(\frac{2}{nK} \right) = \frac{2}{K}. \blacksquare$$

We now turn to Type 2 Turing machines and representations. Recall that a real number $a \in \mathbf{R}$ is *(Turing) computable* [14] iff there is a (Type 2) Turing machine without input tapes which does not stop and writes on its unique output tape a signed binary digit representation of a. The set of such numbers is denoted by $\mathbf{T}$. It is worth noting that the definition of computable real is essentially independent of the representation chosen [6, 9, 5], and that $\mathbf{T}$ is a real closed field [14].

Proposition 2 Let $p(x_1, \ldots, x_n) \in \mathbf{T}[x_1, \ldots, x_n]$. If $p(\boldsymbol{a}) \neq 0$, it is Turing decidable whether $p(\boldsymbol{a}) > 0$ (in the sense that there is a Type 2 Turing machine which decides the set $\{\boldsymbol{a} \in R^n \mid p(\boldsymbol{a}) > 0\}$ relative to $\{\boldsymbol{a} \in R^n \mid p(\boldsymbol{a}) \neq 0\}$).

Proof. In order to prove the statement, it is sufficient to show how to obtain, for any ℓ, a prefix $s = b_n b_{n-1} \cdots b_0.b_{-1} b_{-2} \cdots b_{-m}$ of length at least ℓ of a binary digit representation of the evaluation of a polynomial. As soon as the interval $[\bar{s} - 2^{-m}, \bar{s} + 2^{-m}]$ does not include 0 (this must eventually happen if $p(\boldsymbol{a})$ is nonzero), we can decide the disequation. But polynomials with coefficient in $\mathbf{T}$ are Type 2 computable functions (because constants in $\mathbf{T}$, sums and products are such). $\blacksquare$

Note that since any polynomial with n variables and coefficients in a finitely generated extension $\mathbf{T}(\alpha_1, \ldots, \alpha_r)$ can be seen as the evaluation of a polynomial in $n + r$ variables with coefficients in $\mathbf{T}$, we have the following

Corollary 2 Let $p(x_1, \ldots, x_n) \in \mathbf{T}(\alpha_1, \ldots, \alpha_r)[x_1, \ldots, x_n]$, with $\alpha_1, \ldots, \alpha_r \in R$. There is a Type 2 Turing machine that for every $\boldsymbol{a}$ with $p(\boldsymbol{a}) \neq 0$ accepts (rejects) $\langle a_1, \ldots, a_n, \alpha_1, \ldots, \alpha_r\rangle$ iff $p(\boldsymbol{a}) > 0$ ($p(\boldsymbol{a}) < 0$, respectively).

In order to prove our main theorem, we now show how to produce signed binary digit expansions on a δ-uniform machine. We assume the existence of a subroutine **IntRepr** which returns a signed binary digit description of an integer (this can be straightforwardly accomplished in a δ-uniform manner).

Lemma 7 The following subroutine is correct, in the sense that its δ-computed function satisfies the condition stated in the heading, regardless[8] of the value of δ:

[8]This does not mean that the output will be the same for all δ.

```
subroutine SBDprefix(x : R, ℓ : integer): string of {1̄, 0, 1, .};
   /* Returns at least ℓ symbols of a signed binary digit expansion of x */
var  y : R;
     s : string of {1̄, 0, 1, .};
begin
   y ← [x] − 1;
   s ← IntRepr(y) + ‘.’;
   y ← x − y;
   while (|s| < ℓ)
      y ← 2y;
      if (y ≤ 0) and (y ≥ 0)    s ← s + ‘0’
      elseif (y > 0) do
         s ← s + ‘1’;
         y ← y − 1
      od
      elseif (y < 0) do
         s ← s + ‘1̄’;
         y ← y + 1
      od
   loop;
   return(s)
end
```

Proof. We firstly show that at the start of the loop we have always $-1 < y < 1$; moreover, at the start of k-th iteration the invariant relation $\bar{s} + y/2^{k-1} = x$ holds. This implies that as ℓ grows the subroutine produces a signed binary digit representation of x.
First of all, by Lemma 6

$$x - 1 + \delta < [x]_\delta - 1 \leq x + \delta,$$

and this implies

$$1 > 1 - \delta > x - ([x]_\delta - 1) \geq -\delta > -1.$$

Moreover, $\bar{s} + y/2^0 = [x]_\delta - 1 + x - [x]_\delta + 1 = x$. Thus, the base condition is true.

After y has been doubled, the first **if** is executed only if $-\delta \leq y \leq \delta$, which implies $-1 < y < 1$ at the next iteration; moreover, $\bar{s}$ is not changed, y doubles and k grows by one, so the invariant is preserved.

The second **if** is executed if $\delta < y < 2$; after decrementation, we have again $-1 < \delta - 1 < y < 1$; $\bar{s}$ grows by $1/2^k$, y doubles and is decremented by one, and k grows by one. A straightforward calculation shows that again the invariant is preserved. Analogously for the third case. ▮

We are now going to provide the main results of this section. A δ-uniform machine on $R \not\subseteq \mathbf{T}$ without restrictions on the coefficients of the polynomials $g_q(\boldsymbol{x})$ is in general more powerful than a Type 2 Turing machine. Indeed, given a non-decidable set $S \subseteq \mathbf{N}$ such that $a = \sum_{i=0}^{\infty} \chi_S(i) 2^{-i-1} \in R$ (this must happen for some S unless $R \subseteq \mathbf{T}$), the set

$$X = \bigcup_{k \in S} B_{\frac{1}{2}}(k) \subseteq R$$

is trivially decidable by a δ-uniform BSS machine by unpacking the bits[9] of the constant a. Suppose there is a Turing machine semi-deciding X: then the same machine would semi-decide $X \cap \mathbf{N} = S$. Nevertheless, if the coefficients of the δ-uniform machine are presented as an additional input (or oracle) to the Type 2 Turing machine, the computational power is the same, as shown by the following

Theorem 2 Let $X \subseteq R^n$. Then X is δ-uniformly semi-decidable by a machine M with coefficients $\alpha_1, \dots, \alpha_r$ iff there exist a Type 2 Turing machine M' with $n+r$ input tapes such that for all $\langle x_1, \dots, x_n \rangle \in R^n$

$$\langle x_1, \dots, x_n \rangle \in X \iff M' \text{ halts on input } \langle x_1, \dots, x_n, \alpha_1, \dots, \alpha_r \rangle.$$

Proof. Clearly, by reading a sufficient number of digits from the input tapes M' can emulate the behaviour of M, and evaluate correctly each conditional test against $-\delta$ using Corollary 2, unless the polynomial evaluates exactly to $-\delta$. Thus, we dovetail the simulation of M for all dyadic thresholds, and show that for at least one dyadic the simulation terminates. Indeed, if the accepting path with threshold 0 contains tests which evaluate to 0, then letting $p_1, p_2, \dots, p_i$ be the polynomials evaluated negatively we have that for any dyadic threshold smaller than

$$\min\{-p_1(x_1, \dots, x_n, \alpha_1, \dots, \alpha_r), \dots, -p_i(x_1, \dots, x_n, \alpha_1, \dots, \alpha_r)\}$$

the accepting path will remain the same, but all tests will be strict (and thus evaluable by M) inequalities.

The other side of the claim is easily obtained by emulating the behaviour of M': one just has to use Lemma 7 in order to produce, one digit at a time, a signed binary digit representation of (the components of) the input and of the coefficients. Since by definition the behaviour of a Type 2 Turing machine does not depend on the specific representative chosen for the inputs, the resulting machine is δ-uniform. ∎

This theorem yields an immediate consequence:

Corollary 3 Let $X \subseteq R^n$. Then the following conditions are equivalent:

(i). X is Turing semi-decidable;

(ii). X is semi-decidable by a δ-uniform machine with coefficients in $\mathbf{Q}$;

(iii). X is semi-decidable by a δ-uniform machine with coefficients in $\mathbf{T}$.

Note that unless an approximated semantic is defined, this result is not extendible to the *functions* computed δ-uniformly (consider, for instance, the constant function π).

[9] We are tacitly combining Lemma 7, Proposition 1 and the fact that a δ-uniform machine can emulate a Turing machine.

5 δ-uniform vs. standard BSS machines

In this section we are going to approach the problem of relating decidability properties of standard and δ-uniform BSS machines for real closed fields (the previous results are valid in any archimedean field). It turns out that the key notion with this respect is a local boundedness of the accepting times.

Definition 2 For any BSS (possibly δ-uniform) machine M, if $\alpha_1, \alpha_2, \dots, \alpha_r \in R$ are the coefficients of the polynomials appearing in the description of M we let $E_M = \mathbf{Q}(\alpha_1, \dots, \alpha_r) \subseteq R$ be the *extension of* M. If $X \subseteq R^n$ is (δ-uniformly) (semi-)decided by a machine with coefficients $\alpha_1, \dots, \alpha_r$, we shall simply say that X is *(δ-uniformly) (semi-)decidable using* $\alpha_1, \dots, \alpha_r$.

Definition 3 Let M be a BSS machine. A point $\boldsymbol{a} \in \Omega_M$ is *critical (for M)* iff for every open neighbourhood $U \ni \boldsymbol{a}$ the set $\{\text{accepting time of } \boldsymbol{x} \mid \boldsymbol{x} \in U \cap \Omega_M\} \subseteq \mathbf{N}$ is unbounded. We say that M is *locally time bounded* iff all points in Ω_M are noncritical.

Theorem 3 Let R be a real closed field, and $X \subseteq R^n$. Then X is semi-decidable by a δ-uniform machine iff it is open and semi-decidable by a locally time bounded BSS machine; moreover, the machines can be chosen so that they use the same coefficients.

Proof. Right-to-left implication is obtained similarly to Theorem 2: we dovetail emulations of the δ-uniform machine for all dyadic thresholds, and notice that at least for one threshold all tests along an accepting path are strictly positive or negative, which implies that an entire neighbourhood will follow the same path, giving a local bound for the accepting time.

For the other side, recall from [1] that a machine M stops within time T accepting an input $\boldsymbol{a}$ iff there are $u_0, u_1, \dots, u_T, q_0, q_1, \dots, q_T$ in R, and $\boldsymbol{x}_0, \boldsymbol{x}_1, \dots, \boldsymbol{x}_T$ in R^n such that

$$\begin{aligned} q_0 &= 1 \\ q_T &= N \\ \boldsymbol{x}_0 &= I(\boldsymbol{a}) \\ \beta(q_{t-1}, \boldsymbol{x}_{t-1}u_{t-1}{}^2) - q_t &= 0 \\ \boldsymbol{x}_{t-1}(x_{t-1}u_{t-1}{}^2 + 1)(x_{t-1}u_{t-1}{}^2 - 1) &= 0 \\ g(q_{t-1}, \boldsymbol{x}_{t-1}) - \boldsymbol{x}_t &= 0 \end{aligned}$$

where we denoted with x_t the first coordinate of $\boldsymbol{x}_t$, and the polynomials β and g are derived from the computing endomorphism (see [1]). In fact, we can generate these *register equations* on a δ-uniform machine (no test is necessary).

We now use Lemma 6 in order to produce a sequence $\boldsymbol{r}_k$ of rationals such that $\boldsymbol{a} \in B_{2^{-k}}(\boldsymbol{r}_k)$. Since M is locally time bounded, Lemma 3 guarantees that $\boldsymbol{a}$ is accepted iff there is some K such that for all $k \geq K$ the register equations for input $\boldsymbol{x}$ at time k, prefixed by universal quantification over all $\boldsymbol{x}$'s in $B_{2^{-k}}(\boldsymbol{r}_k)$, are satisfied. The idea is to use the quantifier elimination algorithm [13] in order to obtain a set of disequations whose satisfiability is equivalent to that of the previous formula, and then to decide them.

In order to do so, we note that these two steps require the ability to perform exact computations in E_M only. Since the latter is a finite extension of $\mathbf{Q}$, we can use the coding provided by the isomorphism $E_M \cong \mathbf{Q}(x_1, \ldots, x_s)[x]/\langle p(x)\rangle$ described in Section 3, and apply the quantifier elimination algorithm to the formula so obtained. Computation in $\mathbf{Q}(x_1, \ldots, x_s)[x]/\langle p(x)\rangle$ is symbolic, and thus δ-uniform, except for order testing. But whenever we want to order-compare two elements, we can use Lemma 4 after a symbolic check for equality. This completes the proof (openness of X follows trivially by Theorem 1). ∎

We remark that the same proof yields also the following

Corollary 4 Let M be a BSS machine without critical points in Ω_M°; then Ω_M° is δ-uniformly semi-decidable with the same coefficients as M. (Thus, in particular, Ω_M° is semi-decidable without critical points.)

In the next section, by studying critical points we shall prove that local time boundedness is not inherited by output sets, so we cannot expect that open sets which are output sets of locally time bounded machines are δ-uniformly decidable. In fact, we shall prove that there are δ-uniform machines with an output set which is not δ-uniformly semi-decidable with the same coefficients. In the mean time, we conclude with the following

Corollary 5 Let R be a real closed field, and $X \subseteq R^n$. Then X is Turing semi-decidable iff it is open and semi-decidable by a locally time bounded BSS machine with rational (equivalently: computable) coefficients.

6 Negative results on critical points

It is of course natural to ask whether the local time bound appearing in Theorem 3 is actually a nontrivial one. In order to answer this question, we start giving some examples. We remember that the field R is now assumed to be real closed.

Proposition 3 Let $X \subseteq R$ be such that X and X^c are dense in R. If a BSS machine M semi-decides X, then all points of X are critical for M.

Proof. For each accepting path, the corresponding semi-algebraic set is finite, because of the density of X and X^c. This implies that for every open $U \subseteq R$ only a finite, and thus proper subset of $X \cap U$ can be accepted within any finite time. ∎

The previous proposition tells us that, for instance, $\mathbf{Q}$ is a semi-decidable set made of critical points, as well as $\mathbf{A}$ (the set of real algebraic numbers) as long as $\mathbf{A} \subset R$. As another example [12], consider the set X given by the unit open ball of R^2 augmented with all border points having both coordinates in $\mathbf{A}$ (we assume $\mathbf{A} \subset R$). Note that all other points on the border have both transcendental coordinates. If there were a locally time bounded BSS machine M semi-deciding X, then we would obtain by the Riemann mapping of R into S^1 a machine contradicting Proposition 3.

These considerations show that critical points on the border of the halting set can sometimes be an unavoidable phenomenon. However, we are going to show that this may be true even of some *internal* points.

First of all, observe that a critical point is always accepted along a path containing a test which evaluates exactly to zero: because of continuity, points which do not satisfy exact tests cannot be critical[10].

Theorem 4 The critical points of a BSS machine M are a closed subset of Ω_M. Moreover, if R has infinite transcendence degree they are nowhere dense in R^n.

Proof. By the very definition, every noncritical point has an open (in Ω_M) neighbourhood of noncritical points. We just need to show that, if R has infinite transcendence degree, for every open neighbourhood U of a critical point internal to Ω_M, $U \cap \Omega_M$ contains noncritical points. But $U \cap \Omega_M^\circ$ is open, and contains by density a point whose coordinates are algebraically independent over E_M. This means that, along the accepting path, the point gives rise only to strictly negative or positive tests, and it is thus noncritical. ∎

Note that the infinite transcendence condition is necessary: otherwise, we could build a machine which halts on R, so that to each accepting path corresponds a finite number of elements of R (this can be done by enumerating the polynomials having coefficients built with the transcendence base of R). Of course, as in the proof of Proposition 3, all points of R would be critical.

We now come to the main results of this section. The following theorem was inspired by an example given by Vasco Brattka [3]:

Theorem 5 For every $\alpha_1, \alpha_2, \ldots, \alpha_r \in R$ there is an open set $X \subseteq R$ BSS decidable using $\alpha_1, \ldots, \alpha_r$ such that every BSS machine M with $E_M \subseteq \mathbf{Q}(\alpha_1, \ldots, \alpha_r)$ semi-deciding X has (infinite) critical points.

Proof. Let b_i^j, $j < 0$, be the j-th binary digit of α_i after the decimal point, and let $A \subseteq \mathbf{N}$ be the oracle defined by

$$A = \{i - rj \mid 1 \leq i \leq r,\ j < 0, b_i^j = 1\}.$$

Consider now the universal recursive function $\upsilon^A(j, n)$ relativized to A (we see it as a one-argument function by Cantor pairing), and let f be a function recursive in A enumerating the domain of υ^A. Finally, let

$$X = \bigcup_{n \in \mathbf{N}} \left[\bigcup_{i \notin f([n])} \left(i-1, i - \frac{1}{2+n} \right) \cup \bigcup_{i \in f([n])} \left(i - \frac{1}{5 + \min f^{-1}(i)}, i \right) \right] \cup \mathbf{N},$$

where $[n] = \{0, 1, \ldots, n-1\}$. Intuitively, the set X is built as follows: all integers are part of X; moreover, increasingly bigger open intervals of the form $(i-1, i-1/(2+n))$ are added to

[10] This observation has been used in [19] in order to show that the halting sets of machines whose accepting paths never contain exact tests are Turing semi-decidable.

the right of $i - 1$ until we (possibly) find an n such that $i = f(n)$. If we find such a (least) n, we add to the left of i a small interval (just to make the set open). The following BSS program decides X:

BSS machine $M(x : R)$;
  /* Decides the set X. */
var i, n : integer;
begin
  if x is integer **return**(1);
  $i \leftarrow \lceil x \rceil$;
  $n \leftarrow 0$;
  forever
    if $x \in \left(i - 1, i - \frac{1}{2+n}\right)$ **return**(1);
    if $f(n) = i$ **exit**;
    $n \leftarrow n + 1$
  loop
  if $x \in \left(i - \frac{1}{5+n}, i\right)$ **return**(1);
  return(0)
end

Note that we can compute f because we can decide membership to A just by unpacking the binary digits of $\alpha_1, \ldots, \alpha_r$. The machine M certainly terminates, because for every non-integer input the conditions inside the loop cannot fail forever (if only by the archimedean property). Correctness is straightforward.

Suppose now by contradiction that there is a locally time bounded BSS machine satisfying the hypothesis; by Theorem 3 X is δ-uniformly semi-decidable using $\alpha_1, \ldots, \alpha_r$; by Theorem 2, there is a Type 2 Turing machine with additional inputs $\alpha_1, \ldots, \alpha_r$ semi-deciding X, or, equivalently, a machine with oracle A using a single input tape. Finally, this implies that there is a Turing machine M with oracle A which enumerates a sequence of open rational intervals whose union is X.

Consider now a Turing machine M' with oracle A working as follows: for each input $i \in \mathbf{N}$, we run M and find the first interval $(l, r) \ni i$ (remember that $\mathbf{N} \subseteq X$). Then, we search for the least $n \in \mathbf{N}$ such that either $f(n) = i$ or $i - 1/(2 + n) > l$. By the archimedean property, either the first or the second condition becomes ultimately true; moreover, if the second one becomes true then we have (by the definition of X) that

$$[i - 1, i] \subseteq \left(i - 1, i - \frac{1}{2 + n}\right) \cup (l, r) \subseteq X,$$

which implies that i is not in the range of f. Thus, M' decides the halting problem relativized to A, which is impossible. ∎

The previous theorem shows in particular that there are subsets of the reals BSS semi-decidable "without constants" (i.e., by a machine M with $E_M = \mathbf{Q}$) which are not Turing semi-decidable (this was in fact the original example [3]). Since the description of an open set as a union of balls provides a local time bound, we get also the following

Corollary 6 For every $\alpha_1, \alpha_2, \ldots, \alpha_r \in R$ there is an open set $X \subseteq R$ BSS decidable using $\alpha_1, \ldots, \alpha_r$ such that no BSS machine M with $E_M \subseteq \mathbf{Q}(\alpha_1, \ldots, \alpha_r)$ can enumerate a sequence of open balls (of arbitrary center and radius) whose union is X.

We now proceed to show that

Theorem 6 For every $\alpha_1, \alpha_2, \ldots, \alpha_r \in R$ there is a δ-uniform machine M with coefficients $\alpha_1, \ldots, \alpha_r$ such that every BSS machine M' with $E_{M'} \subseteq E_M$ semi-deciding the output set of M (which is open) has (infinite) critical points.

Proof. The main idea is to build the set X given by Theorem 5 by mapping linearly certain open intervals of R to the intervals appearing in the construction of X. Other intervals are used in order to output the integers (using a constant function). The program is as follows:

δ-uniform BSS machine $M(x : R)$;
  /* Outputs the set X defined in Theorem 5. */
var i, j, n : integer;
begin
  $j \leftarrow 0$;
  if $x \in \left(-\infty, -\frac{1}{2}\right)$ **return**(0);
  forever
    if $x \in \left(j - \frac{1}{2}, j\right)$ **return**(j);
    if $x \in \left(j, j + \frac{1}{2}\right)$ **do**
      $i, n \leftarrow \mathrm{sx}(j), \mathrm{dx}(j)$;   /* Cantor pairing inverse */
      if $i \in f([n])$ **return**$\left(i - 2(x - j)\dfrac{1}{5 + \min f^{-1}(i)}\right)$
      else return$\left(i - 1 + 2(x - j)\dfrac{n+1}{n+2}\right)$
    od;
    $j \leftarrow j + 1$
  loop
end

Note that the interval membership tests of M are all realizable *via* a δ-uniform subroutine which is partially correct, and is nonterminating exactly on the endpoints of the interval; thus, $\Omega_M = R \setminus \{k/2 \mid k \in \mathbf{N} \cup \{-1\}\}$. Moreover, we remark again that f can be computed by unpacking the bits of $\alpha_1, \ldots, \alpha_r$. It is straightforward to check that $\varphi_M(\Omega_M) = X$. ∎

Corollary 7 For every $\alpha_1, \alpha_2, \ldots, \alpha_r \in R$ there is a δ-uniform machine M with $E_M = \mathbf{Q}(\alpha_1, \ldots, \alpha_r)$ such that no δ-uniform machine machine M' with $E_{M'} \subseteq E_M$ can semi-decide the output set of M (which is open).

The same techniques used in the proof of the previous theorems can be used to prove the following purely BSS-theoretic result:

Theorem 7 For every $\alpha_1, \alpha_2, \ldots, \alpha_r \in R$ there is a closed (open) set X BSS decidable using $\alpha_1, \ldots, \alpha_r$ such that X° ($\bar{X}$, respectively) is not semi-decidable by any BSS machine M with $E_M \subseteq \mathbf{Q}(\alpha_1, \ldots, \alpha_r)$.

Proof. We define the oracle A and the function f as in the proof of Theorem 5. Let

$$X = \bigcup_{n \in \mathbf{N}} \bigcup_{i \notin f([n])} \left[i - 1, i - \frac{1}{2+n} \right],$$

and

$$Y = \bigcup_{n \in \mathbf{N}} \bigcup_{i \notin f([n])} \left(i - \frac{3}{4}, i - \frac{1}{2+n} \right).$$

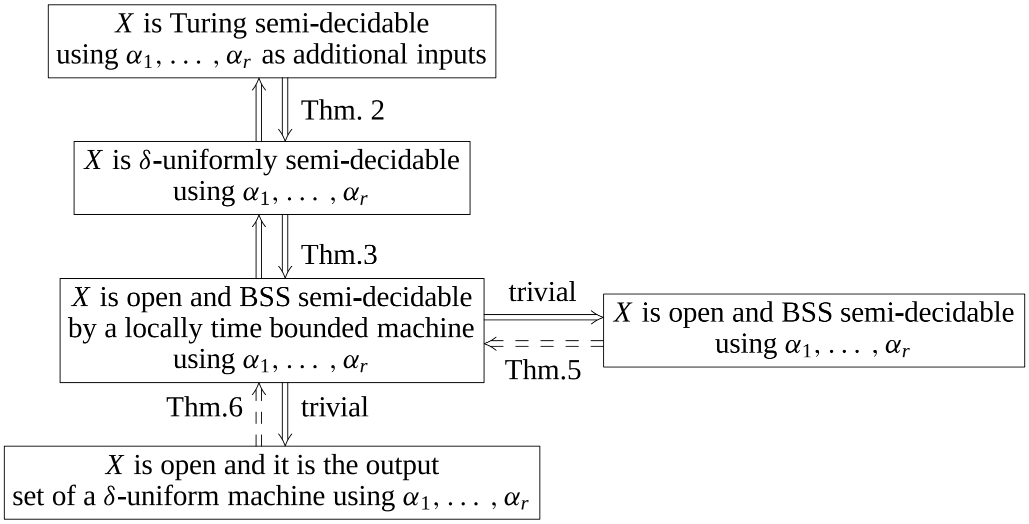
It is easy to modify the machine described in the proof of Theorem 5 in such a way to decide X or Y.

Suppose now by contradiction that there is a BSS machine M semi-deciding X° ($\bar{Y}$): the interior (closure) operator keeps in X (adds to Y) exactly the natural numbers which are not enumerated by f. Thus, the machine M, restricted to the integers, would decide the halting problem relativized to A. But M can be emulated on the rationals by a Turing machine with oracle A, by means of the same techniques used in the proof of Theorem 3. ∎

We just mention that a more comprehensive account of the previous results can be given in the framework of the theory of *degrees of unsolvability* [15]. In fact, the precise condition on E_M for the previous two theorems to hold is that there is no $\alpha \in E_M$ such that $\operatorname{dg} \alpha \geq (\operatorname{dg} \alpha_1 \vee \cdots \vee \operatorname{dg} \alpha_r)'$, where the degree of a real number is the degree of the set defined by its binary expansion—see [16, 8, 2]. Unless R is *closed by jump*, i.e., for each $\alpha \in R$ there is a $\beta \in R$ such that $\operatorname{dg} \beta \geq (\operatorname{dg} \alpha)'$, the previous theorem provides BSS decidable open (closed) sets whose closure (interior) is not BSS semi-decidable (even with additional constants). In a slogan, "equality is (at least) a jump"; it is an open problem to decide whether this is tight, i.e., to prove or disprove the following

Conjecture 1 For every $\alpha_1, \alpha_2, \ldots, \alpha_r \in R$ and every $X \subseteq R^n$ BSS semi-decidable using $\alpha_1, \ldots, \alpha_r$ there is a locally time bounded machine semi-deciding X using $\alpha_1, \ldots, \alpha_r, \beta$, where $\operatorname{dg} \beta = (\operatorname{dg} \alpha_1 \vee \cdots \vee \operatorname{dg} \alpha_r)'$.

We conclude by resuming our main results in the following diagram, where each arrow is labelled with the corresponding theorem, and dashed arrows represent nonimplications (note that R is required to be real closed, except for Theorem 2):



7 Acknowledgements

A number of people contributed with useful suggestions and discussions to this paper. In particular, an email exchange with Christian Michaux gave rise to a useful example, and Vasco Brattka commented deeply the first version of this work, pointing to useful references and suggesting the counterexample inspiring Theorem 5. We are also indebted with the participants of the Workshop on *Computability and Complexity in Analysis*, held at Schloß Dagstuhl in April 1997, for several useful inputs.

References

[1] Lenore Blum, Mike Shub, and Steve Smale. On a theory of computation and complexity over the real numbers: NP-completeness, recursive functions and universal machines. *Bull. Amer. Math. Soc.*, 21:1–46, 1989.

[2] Paolo Boldi and Sebastiano Vigna. The Turing closure of an archimedean field. Preprint, 1997.

[3] Vasco Brattka. Personal communication.

[4] Vasco Brattka and Peter Hertling. Feasible real random access machines. Informatik Berichte 193, FernUniversität Hagen, 1995.

[5] Thomas Deil. Darstellungen und Berechenbarkeit reeller Zahlen. Informatik-Berichte 51, FernUniversität, Hagen, 1984.

[6] Rudolf Freund. Real functions and numbers defined by Turing machines. *Theoretical Computer Science*, 23(3):287–304, May 1983.

[7] Steven A. Gaal. *Point Set Topology*. Pure and Applied Mathematics. Academic Press, 1964.

[8] Ker-I Ko. Reducibilities on real numbers. *Theoretical Computer Science*, 31(1–2):101–123, May 1984.

[9] Ker-I Ko. *Complexity Theory of Real Functions*. Progress in Theoretical Computer Science. Birkhäuser, 1991.

[10] Christoph Kreitz and Klaus Weihrauch. Theory of representations. *Theoretical Computer Science*, 38(1):35–53, May 1985.

[11] Serge Lang. *Algebra*. Addison-Wesley, 1971.

[12] Christian Michaux. Personal electronic communication.

[13] James Renegar. On the computational complexity and geometry of the first-order theory of the reals. Part I, II and III. *Journal of Symbolic Computation*, 13:255–352, 1992.

[14] H.G. Rice. Recursive real numbers. *Proc. Amer. Math. Soc.*, 5:784–791, 1954.

[15] Joseph R. Shoenfield. *Degrees of Unsolvability*. North-Holland, Amsterdam, 1971.

[16] Robert I. Soare. Recursion theory and Dedekind cuts. *Trans. Amer. Math. Soc.*, 140:271–294, 1969.

[17] B. L. van der Waerden. *Algebra*. Springer, Berlin, 8 edition, 1971.

[18] Klaus Weihrauch. Type 2 recursion theory. *Theoretical Computer Science*, 38(1):17–33, May 1985.

[19] Ning Zhong. Recursively enumerable subsets of R^q in two computing models: Blum-Shub-Smale machine and Turing machine. Preprint.