A Survey on real Structural Complexity Theory

K. Meer *
RWTH Aachen
Lehrstuhl C für Mathematik
Templergraben 55
D - 52062 Aachen
meer@rwth-aachen.de

C. Michaux †
Université de Mons-Hainaut
Faculte des Sciences
Avenue Maistriau
B - 7000 Mons
chrmich@sun1.umh.ac.be

February 25, 1996

Abstract

In this tutorial paper we overview research being done in the field of structural complexity and recursion theory over the real numbers and other domains following the approach by Blum, Shub, and Smale [10].

1 Introduction

Quite a lot of computational problems arising in classical fields of mathematics (such as numerical analysis, optimization, geometry etc.) deal with the real or complex numbers as underlying structure. However, theoretical foundations of computability and complexity - as being developped for example by Gödel, Church, Turing and others - have been restricted to discrete problems only. The Turing machine, widely accepted by theoretical computer scientists as formal concept of a computing machine, intrinsically only works over finite alphabets. Classical complexity theory thus is mainly concerned with problems from combinatorics, number theory and logic.
As a consequence classical real number algorithms seem to be modelled (at least theoretically) inadequately (cf. [101]). Generally the way of investigating such algorithms (for example Newton's method, Gaussian elimination etc) is not that of changing all inputs into a bit representation analyzing the problem on that level; elements from the underlying structure are rather taken as entities. As another example consider the task of deciding, whether a given complex point belongs to the Mandelbrot set (cf. [6]). Here the Turing machine seems not to provide the right approach to deal with. Hence the following question arises: what is a satisfactory framework to handle recursion and complexity issues for problems in continuous domains?
Notions of computation over arbitrary rings like the real random access machine, algebraic circuits, straight line programs have already been considered at least since the 80's in algebraic complexity theory. Research in that field has been extremely fruitful and is in fact closely related to the Blum-Shub-Smale model we want to consider. This especially is the case for lower bound results. Here the currently available methods are mainly based on " non-uniform" techniques developped in algebraic complexity theory. For a survey on that field we refer to [108],[118] as well as the forthcoming book [13]. However, contrary to one of the fundamental features of the Turing model, the non-uniformity of this approaches implies that only problems of fixed input dimension are dealed with. Thus concepts of uniform complexity classes are not included.

In 1989 Blum, Shub and Smale [10] introduced a model of computation and complexity over the reals as well as arbitrary rings. It gives a structural approach to complexity theory over these domains, similar to the Turing machine over the integers. Their model is uniform and thus allows to transform main ideas of discrete complexity theory into a more general framework. By regarding

*partially supported by EC Working group NeuroCOLT under contract 8556
†partially supported by Wissenschaftsministerium des Landes Nordrhein-Westfalen and by EC Working group NeuroCOLT under contract 8556

real numbers as entities it enables to combine the theory of computation with analysis, geometry and topology. The famous P versus NP question reappears as a major open problem in that approach, too. For some previous works in this direction see the references in [10]. May be the closest approach was given by Skandalis (see [96]) who already developped a uniform recursion model in general structures (in particular over fields). But his model does not explicitely allow machine constants and no complexity aspects are included.

Just like the Turing machine the Blum-Shub-Smale model (henceforth we use the shorthand BSS-model) is an idealization of a computing machine. One hope in analyzing it is to get new and different insights into the complexity of problems. As one important example we mention the linear programming problem. Even though it is well known to be solvable in polynomial time in the discrete setting ([51] and [52]) the existence of so-called strongly polynomial-time algorithms is still a major open problem (see [111]).

Let us point out that important work also has been done related to the difficult question how to implement real number algorithms practically. Since this area is out of the scope of this paper, we just refer to the survey [92] and the references given there.

The aim of the present paper is to overview research in the Blum-Shub-Smale approach since 1989. Clearly numerous former results can be revisited in the context of the BSS-setting. However we want to focus on recent work directly addressed to the BSS-theory.

Our survey is organized as follows: in section 2 the BSS-model over the real numbers is introduced together with the basic related definitions and notions (computability, decision-problems, decidability, complexity classes, completeness etc). Subsection 2.2 deals with first results building up the starting point for doing complexity theory over $\mathbb{R}$ (existence of $NP_{\mathbb{R}}$-complete problems, decidability of class $NP_{\mathbb{R}}$). The third section collects research connecting BSS theory and discrete complexity issues. It mainly addresses the refinding of classical complexity classes as so called boolean parts of real ones. To this aim some variations of the BSS model (additive, linear, weak machines) are analyzed. We then turn to separation results and lower bound methods. The "weak" models under consideration in section 3 allow a positive answer to the $P \neq NP$-question in the according models by non-uniform methods. Application of such methods to the original setting are discussed in 4.3.

Beside complexity theoretic questions also recursion theory has been investigated for the Blum-Shub-Smale approach. Progress in that area is surveyed in section 5. As these first sections will show many problems naturally lead to consideration of varied models (different operations and/or cost-measures) and different underlying structures. Thus many questions treated so far can be formulated for more general structures as well. Results into this direction are summarized in section 6. Moreover a machine-independent model-theoretic approach to capture real complexity classes is presented in section 7. The paper concludes with some remarks concerning further research directions and a current bibliography.

Throughout the sequel notions like discrete or classical complexity theory always refer to the Turing machine model together with the bit-size measure. The same holds for notions of complexity classes without an additional index (like $P, NP, P/poly, \ldots$). For a comprehensive treatment of the classical theory see [2],[39] and the survey by Stockmeyer [106].

Even though we tried to present an almost complete survey we are quite sure of having forgotten to mention (or even do not know) some related work. We apologize for this.

2 The Blum-Shub-Smale model

Let us start with defining the computational model of the Blum-Shub-Smale machine as well as introducing the main related complexity theoretical concepts. In this section we deal with the real numbers as underlying structure, whereas more general settings are considered in section 6 .

2.1 Definitions

Essentially a BSS-machine can be considered as a Random Access Machine over $\mathbb{R}$, which is able to perform the basic arithmetic operations at unit cost and which registers can hold arbitrary real numbers.

Definition 2.1 *([10])* Let $Y \subset \mathbb{R}^\infty := \bigcup_{k\in\mathbb{N}} \mathbb{R}^k$. A *BSS-machine* M *over* $\mathbb{R}$ *with admissible input set* Y [1]is given by a finite set I of instructions labelled by $0, \ldots, N$. A configuration of M is a quadruple $(n, i, j, x) \in I \times \mathbb{N} \times \mathbb{N} \times \mathbb{R}^\infty$. Here n denotes the currently executed instruction, i and j are used as addresses (copy-registers) and x is the actual content of the registers of M. The initial configuration of M's computation on input $y \in Y$ is $(1, 1, 1, y)$. If $n = N$ and the actual configuration is (N, i, j, x), the computation stops with output x .
The instructions M can perform are of the following types :

computation: $n : x_s \leftarrow x_k \circ_n x_l$, where $\circ_n \in \{+, -, *, :\}$ or
$n : x_s \leftarrow \alpha$ for some constant $\alpha \in \mathbb{R}$.
The register x_s will get the value $x_k \circ_n x_l$ or α resp. All other register-entries remain unchanged. The next instruction will be $n+1$ and the copy-registers are changed selectively according to $i \leftarrow i+1$ or $i \leftarrow 1$ and similarly for j.

branch: n: **if** $x_0 \geq 0$ **goto** $\beta(n)$ **else goto** $n+1$. According to the answer of the test the next instruction is determined (here $\beta(n) \in I$). All other registers are not changed.

copy: $n : x_i \leftarrow x_j$, i.e. the content of the "read"-register is copied into the "write"-register. The next instruction is $n+1$; all other registers remain unchanged.

All α appearing among the computation-instructions built up the (finite) set of *machine constants of* M . □

Remark 2.2 The kind of operations allowed depends on the underlying structure. A branch $x \geq 0$? for example does only make sense in an ordered set, see below.
The copy-registers and -instruction are necessary in order to deal with arbitrary long inputs from $\mathbb{R}^\infty$. The way of changing the entries in the copy-registers (the "addressing") seems to be rather restrictive apart from the fact that there is no indirect addressing. However it is general enough for our purposes, see Remark 2.6 below. □

Now to any BSS-machine M over Y there corresponds in a natural way the function Φ_M computed by M. It is a partial function from Y to $\mathbb{R}^\infty$ and is given as the result of M's computation on an input $y \in Y$.

The following notions are crucial for recursion- and complexity-theory.

Definition 2.3 Let $A \subset B \subset \mathbb{R}^\infty$ and M be a BSS-machine over B .

a) The *output-set* of M is the set $\Phi_M(B)$. The *halting-set* of M is the set of all inputs y for which $\Phi_M(y)$ is defined.

b) A is called *recursively enumerable over* B iff A is the output-set[2] of a BSS-machine over B. (If $B = \mathbb{R}^\infty$ A is simply called *recursively enumerable* .)

c) A pair (B, A) is called a *decision-problem.* It is said *decidable* iff there exists a BSS-machine $\tilde{M}$ with admissible input set B such that $\Phi_{\tilde{M}}$ is the characteristic function of A in B. In that case $\tilde{M}$ *decides* (B, A).

□

As can be seen easily (B, A) is decidable iff A and $B \setminus A$ are both halting sets over B.

Questions on recursion theory will be treated in section 5. We first turn to complexity issues. In order to deal with complexity we have to define as well a cost-measure for BSS-algorithms as a size-measure for problem-instances.

Definition 2.4 a) For $x \in \mathbb{R}^\infty$ such that $x = (x_1, \ldots, x_k, 0, 0, \ldots)$ it is

$$size(x) := k$$

[1]In the sequel we will say a BSS-machine over Y.
[2]This terminology agrees with the terminology classically used for Turing machines. In [10], the term "recursively enumerable"is used for halting sets. But for BSS-machines over the reals, both classes of halting sets and output sets are equal (see [10] and Section 5 below).

b) Let M be a BSS-machine over $Y \subset \mathbb{R}^\infty, y \in Y$. The *running time* of M on y is defined by

$$T_M(y) := \begin{cases} \text{number of operations executed by } M & \text{if } \Phi_M(y) \text{is defined} \\ \infty & \text{else} \end{cases}$$

□

The above definition provides a first important difference with classical complexity theory. Part a) states that any real number - independently of its magnitude - is considered as entity. Similarly b) defines the cost of any basic operation to be 1 - no matter about the operands.

We can now define the most important complexity classes $P_{\mathbb{R}}$ and $NP_{\mathbb{R}}$ for real decision problems.

Definition 2.5 *([10])*

a) A decision problem $(B, A), A \subset B \subset \mathbb{R}^\infty$ belongs to class $P_{\mathbb{R}}$ (deterministic polynomial time) iff there exist a BSS-machine M with admissible input-set B and constants $k \in \mathbb{N}, c \in \mathbb{R}$ such that M decides (B, A) and

$$\forall\, y \in B \quad T_M(y) \le c \cdot size(y)^k \quad .$$

b) (B, A) belongs to class $NP_{\mathbb{R}}$ (*nondeterministic polynomial time*) iff there exists a BSS-machine M with admissible input-set $B \times \mathbb{R}^\infty$ and constants $k \in \mathbb{N}, c \in \mathbb{R}$ such that the following conditions hold :

i) $\Phi_M(y, z) \in \{0, 1\}$

ii) $\Phi_M(y, z) = 1 \;\Rightarrow\; y \in A$

iii) $\forall y \in A\ \exists z \in \mathbb{R}^\infty\ \Phi_M(y, z) = 1$ and $T_M(y, z) \le c \cdot size(y)^k$.

c) (B, A) belongs to class *co-*$NP_{\mathbb{R}}$ iff $(B, B \setminus A) \in NP_{\mathbb{R}}$.

□

Remark 2.6 As is shown in [86] the class $NP_{\mathbb{R}}$ would not change if a more general way of addressing is used in the definition of BSS-machines.
In a similar way as above one defines further complexity classes, for example $EXP_{\mathbb{R}}$ and $NEXP_{\mathbb{R}}$ (here the running time is bounded to be single-exponential). □

The class $P_{\mathbb{R}}$ can be considered as a theoretical formalization of problems being efficiently solvable; the running time increases only polynomially with the problem size (even if in practize big constants k and c can destroy this effect for small input sizes). Nondeterminism in part b) of the definition refers to the vector z. The $NP_{\mathbb{R}}$-machine is not allowed to answer "yes" if the input does not belong to A and for each $y \in A$ there must be a "guess" z that proves this fact in polynomial time. However there is no indication how to get the right guess.
Obviously it is $P_{\mathbb{R}} \subset NP_{\mathbb{R}}$. The question whether $P_{\mathbb{R}} \ne NP_{\mathbb{R}}$ can be considered as the main unsolved problem in real complexity theory and is the analogue to the classical P versus NP problem in the Turing-theory (see [18] and [39]). Informally the difference between $P_{\mathbb{R}}$ and $NP_{\mathbb{R}}$ is the difference of fast proof-finding versus fast proof-checking.

Example 2.7 *([10])* For $k \in \mathbb{N}$ consider the sets

$$F^k := \{f | f \text{ polynomial in } n \text{ unknowns with real coefficients, } deg(f) \le k\ , n \in \mathbb{N}\} \ ,$$

$$F^k_{zero} := \{f \in F^k | f \text{ has a real zero } \}$$

and

$$F^k_{zero,+} := \{f \in F^k | f \text{ has a real zero with all components being nonnegative } \}$$

where a polynomial is represented as an element of $\mathbb{R}^\infty$ by writing down the coefficients of all its monomials in a given order. Then (F^k, F^k_{zero}) and $(F^k, F^k_{zero,+})$ belong to $NP_{\mathbb{R}}$ for all $k \in \mathbb{N}$ by guessing a (nonnegative) zero x, plugging it into f and evaluating $f(x)$. If $k \geq 4$ the according problems lead into the heart of complexity theory as we will see in the next subsection. □

In order to compare problems with respect to the difficulty of solving them the notion of polynomial time reducibility is central. The underlying idea is that a problem (B_1, A_1) is at least as hard to solve as problem (B_2, A_2) if each instance of the latter can be reduced fast (i.e. in polynomial time) into one of the first.

Definition 2.8 Let $(B_1, A_1), (B_2, A_2)$ be decision problems.

a) (B_2, A_2) is *reducible in polynomial time* to (B_1, A_1) iff there exists a BSS-machine M over B_2 s.t. $\Phi_M(B_2) \subset B_1$, $\Phi_M(y) \in A_1 \Leftrightarrow y \in A_2$ and M works in polynomial time.
Notation : $(B_2, A_2) \leq_{\mathbb{R}} (B_1, A_1)$

b) $(B_1, A_1) \in NP_{\mathbb{R}}$ is *$NP_{\mathbb{R}}$-complete* iff (B_1, A_1) is universal in $NP_{\mathbb{R}}$ w.r.t. $\leq_{\mathbb{R}}$ (i.e. any problem in $NP_{\mathbb{R}}$ is reducible to it in polynomial time).

c) $(B_1, A_1) \in$ *co-$NP_{\mathbb{R}}$* is *co-$NP_{\mathbb{R}}$-complete* iff it is universal w.r.t. $\leq_{\mathbb{R}}$ in *co-$NP_{\mathbb{R}}$* .

□

Complete problems are essential for complexity classes because they represent the most hardest problems in it. As in the classical theory the relation $\leq_{\mathbb{R}}$ is both transitive and reflexive. This implies $P_{\mathbb{R}} = NP_{\mathbb{R}}$ iff it exists a $NP_{\mathbb{R}}$-complete problem in class $P_{\mathbb{R}}$. That is the reason why studying complete problems bears such importance.

2.2 Basic complexity results

Before continuing the work on problems like $P_{\mathbb{R}}$ versus $NP_{\mathbb{R}}$ some basic questions must be solved. Of course if problems in $NP_{\mathbb{R}}$ would not be decidable it would not make sense to speak about their complexity. Moreover, because of its importance one is interested in knowing whether $NP_{\mathbb{R}}$-complete problems exist and how they look like. Proving completeness results for decision problems in principle is possible by reducing known complete problems to those in question. Nevertheless it remains the task to find a "first" complete problem. This is one of the major results in [10].

Theorem 2.9 *([10])*

a) For any $k \geq 4$ the problem (F^k, F^k_{zero}) is $NP_{\mathbb{R}}$-complete.

b) All problems in class $NP_{\mathbb{R}}$ are decidable in single exponential time.

The theorem is established by adapting Cook's famous $NP-$completeness result for the 3-Satisfiability problem (see [18] and [39]) in Turing theory to the BSS-model. By using a kind of symbolic dynamics the computation of any machine can be described via a system of polynomial equations . Accepting computations are then related to satisfiable systems which themselves correspond to polynomials in F^k_{zero} . The decidability of problems in $NP_{\mathbb{R}}$ is much harder to show than in discrete complexity theory. This is due to the fact that the space of guesses is a continuum. The problem is closely connected with so called quantifier-elimination over real closed fields : the problem whether a $f \in F^k$ has a zero can be formulated via the first-order formula

$$\exists x_1, \ldots, x_n \; f(x_1, \ldots, x_n) \; = \; 0 \; .$$

Answering this question can be done by transforming the formula into a sentence without quantifiers and check its validity. An algorithm for quantifier elimination in real closed fields was first given by Tarski [109]. The stated time bounds are qualitatively established in [43],[46] and [90].
Let us remark that the general problem of quantifier elimination (i.e. the formula contains both universal and existential quantifiers) is intrinsically of doubly exponential complexity ([32]). For more about the complexity of elimination theory beside the above mentioned papers see [45].

As follows immediately the problem of deciding whether a degree 4 polynomial has no zero is $co\text{-}NP_{\mathbb{R}}$-complete (cf. also section 6).

If $P_{\mathbb{R}} \neq NP_{\mathbb{R}}$ is assumed then $k = 4$ is a sharp bound in Theorem 2.9. This is a result by Triesch [112], who proved (F^k, F^k_{zero}) to belong to $P_{\mathbb{R}}$ for $k \in \{1, 2, 3\}$.

Some more completeness results have been obtained, among them for example

- (QS, QS_{yes}) : does a system of quadratic polynomials have a common zero ($NP_{\mathbb{R}}$-complete [10]).
- is a semi-algebraic set, given by polynomials of degree at most d, non-empty ($NP_{\mathbb{R}}$-complete for $d \geq 2$ [10])
- $(F^k, F^k_{zero,+})$ ($NP_{\mathbb{R}}$-complete for $k \geq 4$ [68])
- is a semi-algebraic set, given by polynomials of degree at most d, convex ($co\text{-}NP_{\mathbb{R}}$-complete for $d \geq 2$ [28]).

For further results cf. also [28] .

Especially with respect to classical complexity theory $(F^2, F^2_{zero,+})$ is interesting. It can be seen as a non-convex quadratic programming problem (with linear constraints). Quadratic Programming is known to be NP-complete in the Turing-setting; however there is not a corresponding result to Triesch's one for nonnegative zeros of degree 2 polynomials and the question is related to the existence of so called strongly polynomial algorithms for mathematical programming problems (see [89]). $(F^2, F^2_{zero,+})$ bounds the complexity of many important combinatorial problems if considered over the reals (for example 3-Satisfiability, Hamiltonian Circuit, Travelling Salesman all are reducible to it in polynomial time, cf. [68]). Thus it would be surprising to establish a result like $(F^2, F^2_{zero,+}) \in P_{\mathbb{R}}$. On the other hand there are good reasons to conjecture $(F^2, F^2_{zero,+})$ not to be any more complete in the real setting. We return to this question later on.

Beside the time measure for computations also the used space resources can be considered. In fact the according counterparts to time classes - for example PSPACE - are extremely important in discrete complexity theory. In principle this concept can similarly be defined in the BSS-model; however a result by Michaux shows the irrelevancy of this notion to this broad context.
In [80] and [83], Michaux showed that there exists an universal constant c such that for any decision problem and for any input, the size of space needed to the computation is at most the size of the input plus c. So this implies that any decision problem (over any ordered rings) can be solved in linear space. Unfortunately the price to pay is an exponential loss of time in the calculations. In Koiran [56] it is shown that the same result holds in a weak version BSS-model (where multiplication is not allowed) but with a polynomial slowdown. So in this model any polynomial time decision problem can be solved by an algorithm working in polynomial time and linear space. The reader is also invited to look to Poizat's book [88, chapter 8] where the result is discussed and proved by using circuits in a very clever way. In particular he shows the following :

Theorem 2.10 *In the BSS-model over the structure* $(\mathbb{R}; +, -, \frac{x}{2}, =, <)$ *every polynomial time decision problem can be solved by an algorithm working in polynomial time needing constant additional space.*[3]

This question is treated in the general setting of recursion theory for structures of finite type (see section 6 for a definition) in [38].

Finally let us mention that the space notion nevertheless seems to be important if it is combined with time-resources. A first result into this direction is given in [42]. We'll come back to it in section 7.

[3]The symbol $\frac{x}{2}$ stands for the function which divides x by 2.

3 Relations with discrete complexity theory, weak and linear Blum-Shub-Smale models

Even though in the BSS-approach reals are entities and its representation by bit-strings (if possible) seems to be of no importance a-priori, there are tight relations also to classical complexity theory. At least for combinatorial problems this is not fully surprising due to the fact that here bit- and algebraic-size measure coincide.
In order to study such relations closer it turned out to be fruitful considering not only the BSS-model itself but also some modifications. To this aim especially the "weak BBS-model" introduced by Koiran [54] as well as linear variants of BSS-machines are important (later on we will come back to other variations of the BSS-model in greater generality).

The weak BSS-model was inspired by the idea to bring the BSS-theory nearer to classical complexity theory by using a different cost-measure for the operations. Consider the following program :

input $x \in \mathbb{R}$
for $i = 1$ *to* n
$\underline{do}$ $x \leftarrow x^2$
$\underline{od}$
output x

Here within n steps the value x^{2^n} is computed, i.e. a polynomial of degree 2^n can be computed in n steps. To avoid such an effect (which is very untypical for the Turing-model) Koiran introduced a different cost measure for the allowed operations. The key idea is to consider the real constants of a machine as separated inputs in order to speak about the bit size of numbers appearing during a computation.

Let M be a BSS-machine with real constants $\alpha_1, \ldots, \alpha_s$. For any input-size n M realizes an algebraic computation tree (cf. [118]). If any node ν is passed by M during this computation, the value computed by M up to this node is of the form $f_\nu(\alpha_1, \ldots, \alpha_s, x_1, \ldots, x_n)$, where $f_\nu \in \mathbb{Q}(\alpha_1, \ldots, \alpha_s, x_1, \ldots, x_n)$ is a rational function with rational coefficients only. Now the *weak cost* of the according operation is fixed as maximum of $deg(f_\nu)$ and the maximum height of all coefficients of f_ν (here the height of a rational $\frac{p}{q}$ is given by $\lfloor log(|p|+1) + log(|q|) \rfloor$).

Definition 3.1 *([54])* The *weak BSS-model* is given as the BSS-model together with the weak cost-measure, i.e. the weak running time of a BSS-machine M on input $x \in \mathbb{R}^\infty$ is the sum of the weak costs related to all operations M performs until $\Phi_M(x)$ is computed. Weak deterministic and non-deterministic polynomial time as well as weak polynomial time reducibility are defined in a straightforward manner (and denoted by an index w, i.e. P_w, NP_w etc.). □

Koiran [54] studies boolean parts[4] of (weak) real complexity classes :

Definition 3.2 *a) ([23], [26], [54])* Let $\mathcal{C}$ be a complexity class over the reals ; the *boolean part* $BP(\mathcal{C})$ *of* $\mathcal{C}$ denotes $\{L \cap \{0,1\}^*; L \in \mathcal{C}\}$.

b) ([41] and also [26]) If $\mathcal{C}$ is a non-deterministic complexity class, then DC (*digital* $\mathcal{C}$) denotes the subclass of those problems in $\mathcal{C}$ which membership can be established by guessing only elements from $\{0,1\}^*$ (for example $DNP_{\mathbb{R}}, DNP_w$).

□

The aim now is to recover classical (boolean) complexity classes as boolean parts of real ones. A characteristic result is the following : (For a definition of the mentioned classical complexity classes, see [2],[39] .)

Theorem 3.3 *([54])*

$$BP(P_w) = P/poly$$

[4] The idea that classes of boolean problems can be helpful to the understanding of BSS-classes of complexity is already present in the preprint of [41](circulating in 1991). There one can find some interesting considerations about the class of problems solvable in digital nondeterministic polynomial time.

The proof idea of this main theorem in [54] is crucial for most results into this direction. To switch from weak BSS-computations to Turing-computations the main task is to deal with the real constants of a weak BSS-machine M. Koiran's idea is to substitute them by rationals the bit-sizes of which are small and to perform the same computation, but now with the rational constants. In order to realize such a substitution - which has to guarantee the same results of the computation at least on inputs from $\{0,1\}^n$ for every $n \in \mathbb{N}$- the following result on semialgebraic sets is established :

Theorem 3.4 *([54]) Let $S \subset \mathbb{R}^s$ be a semialgebraic set with defining polynomials*

$$P_i(\alpha_1, \ldots, \alpha_s) > 0 \ , \ i = 1, \ldots, N \ , \ P_i \in \mathbb{Z}[\alpha_1, \ldots, \alpha_s] \ ,$$

let D be the max deg(P_i), and H the maximum height of the coefficients in the P_i's . If S is nonempty then there exist constants a, b depending only on s and an $\bar{\alpha} \in \mathbb{Q}^s \cap S$ such that the height of $\bar{\alpha}$ is bounded by $a \cdot H \cdot D^b$.

The theorem states that there exist small rational points in a semialgebraic set defined by "small" polynomials over $\mathbb{Z}$. Note that the height of $\bar{\alpha}$ is independent of the number N of polynomials. Analyzing weak polynomial time computations for a fixed input dimension a situation as in Theorem 3.4 can be derived. Due to the fact that only inputs from $\{0,1\}^*$ have to be considered, the real constants $\alpha_1, \ldots, \alpha_s$ are exchanged with $\bar{\alpha}$. Since $\bar{\alpha}$ is small the arising algorithm can be executed in polynomial time also by a Turing-machine. However, for different input dimensions different rational points $\bar{\alpha}$ can appear. This forces the resulting process to be non-uniform - the reason why $P/poly$ comes into play.

It should be noted that similar results in the theory of neural networks have been obtained independently by Sontag and Siegelmann [104]. For readers more interested in the relation with neural networks see also [62].

Beside the weak model also linear real models give strong connections with discrete complexity theory. This is due to the fact that Turing-computations can be simulated by very easy arithmetic operations.

Definition 3.5 If the set of operations in the BSS-model is reduced to additions/
subtractions or to linear operations (i.e. addition/subtractions and scalar multiplications with a fixed finite set of reals) we get *additive* resp. *linear* BSS-machines. If only test-operations "$x = 0$?" can be performed we get BSS-machines *branching on equality* .
Notationally we will indicate the kind of branching as upper index whereas the kind of model as lower index (for example $P^{=}_{lin}$ stands for problems being decidable in deterministic polynomial time by a linear machine branching on equality). □

We summarize some further results which more or less all make use of a result similar to that given by Theorem 3.4. They (and other ones) can be found in [23],[24],[30],[54],[56].

The class $P/poly$ can also be found again as boolean part of $P^{\leq}_{add}$ resp. $P^{\leq}_{lin}$, i.e.

$$P/poly \ = \ BP(P^{\leq}_{add}) \ = \ BP(P^{\leq}_{lin}) \ ([56])$$

For the corresponding non-deterministic classes one gets

$$NP/poly \ = \ BP(DNP_w) \ = \ BP(NP^{\leq}_{add}) \ = \ BP(NP^{\leq}_{lin}) \quad ([30],[56])$$

The uniform boolean classes P and NP can also be recovered as boolean part of complexity classes but for additive or linear BSS-machines branching on equality

$$P \ = \ BP(P^{=}_{add}) \ = \ BP(P^{=}_{lin})$$

and

$$NP \ = \ BP(NP^{=}_{add}) \ = \ BP(NP^{=}_{lin}) \ ([56])$$

(here the proofs rely on a symbolic computation avoiding to deal with the machine constants as real numbers). These results are generalized for the polynomial-time hierarchy upon $NP^{=}_{add}$ and

$NP^{\leq}_{add}$ in [25].
Additive machines have also been studied for parallel computations [19]. Let NC^{add} denote the class of decision problems decidable by a uniform family $\{C_n\}_{n\in\mathbb{N}}$ of additive circuits with poly-logarithmic depth and polynomial size. The corresponding uniformity condition forces C_n to be computable by a P_{add}-machine in polynomial time. Then

$$BP(NC^{=}_{add}) = NC\ ,\ BP(NC^{\leq}_{add}) = NC/poly \text{ and } NC^{\leq}_{add} \subsetneq P^{=}_{add} \quad ([19])\ .$$

As a consequence of Koiran's results in [54] one obtaines that Stockmeyer's polynomial-time hierarchy ([105] and [106]) would collapse to its second level if the classes P_w and DNP_w coincide ([30], [54]). A related result for parallel weak computations is given in [30].

For boolean parts of complexity classes in the (full) BSS-model some results have been obtained by the above techniques, too. Cucker and Grigoriev [23] proved PSPACE/poly to equal the boolean part of parallel polynomial time (as worked out in [30] one can also reduce considerations to weak parallel polynomial time). Koiran showed that $BP(P^{=}_{\mathbb{R}}) \subset BPP$ (problems recognizable in polynomial time by a probabilistic Turing machine with bounded error probability, [54]) and that the real counterpart $PH_{\mathbb{R}}$ of the polynomial-time hierarchy does not contain all subsets in $\{0,1\}^*$ (see [54], an extension can be found in [23]). For an exact definition of $PH_{\mathbb{R}}$ as well as specification of complete sets for the different levels of it we refer to [22].

Finally let us mention that different classes of non-determinism within complexity theory as well over the integers together with the algebraic cost-measure as over general structures of finite type are studied in [47] and [48].

4 Separation results, lower bounds

In this section for the real models considered so far we overview complexity results not concerned with boolean parts. Special interest will be given to separation results and to lower bounds (which often provide separations of certain complexity classes).
One serious problem concerning lower bound methods is that most of them use non-uniform ideas, i.e. restrict themselves to a fixed input dimension trying to prove lower bounds for that dimension. Thus methods from algebraic complexity theory play an important part. We do not want to go into detail for such methods and refer the interested reader to [58], [59], [108], [118] . However one argument used for some of the following results should be presented.

Theorem 4.1 *Let $X \subset \mathbb{R}^n$ (resp. $X \subset \mathbb{C}^n$) be an irreducible algebraic variety given as zero-set of an irreducible polynomial f. Furthermore in the real case let $dim(X) = n-1$. Then any BSS-algorithm deciding the set $(\mathbb{R}^n, X)$ (resp. $(\mathbb{C}^n, X)$) has to compute a non-trivial multiple of f on a Zariski-dense subset of X.*

The proof is given for example in [14],[93] . The main idea is to show that there exists at least one computation path along which "Zariski-almost" all inputs are branched (using Baire's category theorem). Such a "typical" path is then shown to compute necessarily a test-function vanishing on X. Application of the Risler-Dubois- (over $\mathbb{R}$) resp. the Hilbert-Nullstellensatz (over $\mathbb{C}$) yields the assertion.
The theorem states that for certain polynomials testing whether they vanish is not easier than evaluation (modulo the factor hidden in the assertion). We will see later on that sometimes it is possible to switch from the (unknown) multiple to f itself.

4.1 Weak BSS-model

Using the above argument several separation results for the weak model follow :

Theorem 4.2 *([30])*

$$P_w \subsetneq P_{\mathbb{R}} \subset NP_{\mathbb{R}} = NP_w$$

First of all this means that the P_w versus NP_w question is solved in Koiran's model (one can indeed show the stronger separation $P_w \subsetneq NP_w \cap co-NP_w$). The relation $P_w \subsetneq P_{\mathbb{R}}$ is established by considering for fixed $n \in \mathbb{N}$ an irreducible polynomial p of degree 2^n such that the according variety X can be decided in (strong) polynomial time by using a repeated squaring. According to Theorem 4.1 a weak machine solving the problem has to compute a multiple of p in weak polynomial time : this is not possible since the degree of p has to be polynomial in the time of the computation. The equation $NP_{\mathbb{R}} = NP_w$ is interesting in its own. Full non-determinism is not stronger than the weak one. The reason why is that high degree intermediate results can be guessed in order to reduce the weak costs. Moreover Cucker, Shub, and Smale show that the problems (F^4, F^4_{zero}) and (QS, QS_{yes}) to be complete also in the weak setting, i.e. w.r.t. weak polynomial time reductions. Note that the question whether every $NP_{\mathbb{R}}$-complete problem is NP_w-complete too is highly non-trivial : if the answer would be "yes" Theorem 4.2 would imply $P_{\mathbb{R}} \neq NP_{\mathbb{R}}$ (because otherwise every problem in $NP_{\mathbb{R}} = P_{\mathbb{R}}$ would be $NP_{\mathbb{R}}$-complete).

Some corollaries following from Theorem 4.2 should be mentioned. For example it can be shown that neither $P_{\mathbb{R}}$ nor NP_w are subsets of PAR_w, a consequence of which is $DNP_w \subsetneq NP_w$ (here PAR_w denotes all problems being decidable in weak polynomial time using an exponential number of processors, cf. [30]; similar for $PAR_{\mathbb{R}}$). The latter implies typical combinatorial problems like Knapsack or 3-Satisfiability not to be NP_w- complete because they are members of class DNP_w. Similar arguments are used for the main result in [72]. As mentioned above one major open problem in mathematical programming is the question for existence of strongly polynomial time algorithms solving Linear Programming or non-convex Quadratic Programming (i.e. the question whether these problems belong to $P_{\mathbb{R}}$,cf. [89]). In discrete complexity theory the first is known to be in class P whereas the second is $NP-$complete. Thus assuming $P \neq NP$ both have different complexity behaviour. Vavasis [114] conjectured a similar result within the real framework to need very different proof methods (if it is true at all). This is supported by the results in [72], [73] . There it is shown that both kinds of programming problems belong to DNP_w (the search of a solution can be restricted to very special directions, which are codable by digits). Thus on the one hand they are not NP_w-complete in Koiran's model, on the other hand together with Theorem 4.1 also $NP_{\mathbb{R}}$-completeness of any $DNP_{\mathbb{R}}$ problem would have strong consequences: multiples of resultant polynomials have to be computed (non-uniformly) fast in that case [72]. We will address this question later on again.
It is thus reasonable to conjecture that Quadratic Programming is not any longer complete, if real models are considered.

Before switching to linear models let us mention few further results : Both DNP_w and $DNP_{\mathbb{R}}$ possess complete problems [26]. The class DNP_w can be characterized as subclass NP^1_w of NP_w, where instead of a polynomial number of reals only a single real is guessed (i.e. guesses are reduced to a one-dimensional space, cf. [72]). A similar characterization for the full model is unknown. Finally, we have seen that $DNP_w \subsetneq NP_w$, but it is not clear whether $P_w \subsetneq DNP_w$. However for equational weak machines the real Knapsack-problem can be used to separate also these two classes, i.e. $Knapsack \in DNP^=_w \setminus P^=_w$ ([30]) . Results on weak exponential time can also be found in [30].

4.2 Linear and additive machines

Considering the order-free linear resp. additive BSS-model the P versus NP question can be answered.

Theorem 4.3 *([70],[71])* $P^=_{lin} \neq NP^=_{lin}$ *and* $P^=_{add} \neq NP^=_{add}$.

The proofs again use non-uniform methods; to show that deciding integer-solvability of linear equational systems separates the according classes. (The result was independently proven by Smale, see [94]). Koiran [56] mentioned the Knapsack-problem to establish the same separation.
For the ordered case things seem to be much more difficult. At least non-uniform approaches will probably not be sufficient to show $P^{\leq}_{lin} \neq NP^{\leq}_{lin}$: according to a result by Meyer auf der Heide [78] the Knapsack-problem can be solved non-uniformly in deterministic polynomial time.

The concept of digital nondeterminism, which turned out to be important for both the weak and the full BSS-model, is of no special meaning in linear and additive settings. As proved by Koiran we have

Theorem 4.4 *([56])* $DNP^{\leq}_{lin} = NP^{\leq}_{lin}, DNP^{=}_{lin} = NP^{=}_{lin}, DNP^{\leq}_{add} = NP^{\leq}_{add}$ *and* $DNP^{=}_{add} = NP^{=}_{add}$.

The basic idea to switch from arbitrary to digital nondeterminism is guessing the coding of an accepting computation path instead of guessing real numbers which force the according machine to take that path. Then it has to be checked whether the guessed path can appear during a computation (the same idea is also used to gain the above mentioned result $DNP_w = NP^1_w$).
Additionally let us mention that all the above NP-classes are decidable in the corresponding framework. For $NP^{=}_{lin}, NP^{=}_{add}$ cf. [71], for $NP^{\leq}_{lin}$ see [35] and for $NP^{\leq}_{add}$ cf. [37] as well as [25] and [103]. The existence of complete problems in the additive models is settled in [26]. For linear machines a positive answer is given in [40], even though in that case no universal machine exists (see [71] and section 6 below). This seems to be related to the non-existence of so called universal machines (cf. [71] and section 6 below).

4.3 Full BSS-model

Because of its meaning for real complexity theory it is important to study lower bounds for problems in $NP_{\mathbb{R}}$, especially complete ones. As mentioned above, the problem (QS, QS_{yes}) of deciding whether a system of quadratic polynomials has a real solution is $NP_{\mathbb{R}}-$complete. Shub [93] has related this problem with Theorem 4.1. Instead of dealing with arbitrary quadratic systems he considers homogeneous systems of n polynomials in n unknowns and asks for non-trivial solvability. The problem clearly belongs to $NP_{\mathbb{R}}$. It is well known ([93],[116]) that for every $n \in \mathbb{N}$ the solvable systems built up an algebraic variety given as the zero-set of the so-called (irreducible) *Resultant polynomial* RES_n. Application of Theorem 4.1 now yields

Theorem 4.5 *([93]) If* $P_{\mathbb{R}} = NP_{\mathbb{R}}$ *then there exists a (non-uniform) BSS-algorithm computing for every* $n \in \mathbb{N}$ *a non-trivial multiple of* RES_n *in polynomial time.*

Computation of resultant polynomials seems to be a hard task (cf. [16]). Thus Shub's idea is to attack the "$P_{\mathbb{R}} = NP_{\mathbb{R}}$?" question by proving lower bounds for the latter problems. The relevance of such lower bounds is also substantiated by the following result

Theorem 4.6 *([72]) If some problem in* $DNP_{\mathbb{R}}$ *is* $NP_{\mathbb{R}}-$*complete then there exists a (non-uniform) BSS-algorithm computing for every* $n \in \mathbb{N}$ *a non-trivial multiple of* RES_n *in polynomial time.*

Thus a problem like Linear or Quadratic Programming is hard to solve in the BSS-model only if computation of resultants is easy (in the above sense). BSS-algorithms for solving special subcases of LP problems are investigated in [11].
However the known lower bounds for computation of RES_n are far away from being exponential (which would be necessary to solve the $P_{\mathbb{R}} = NP_{\mathbb{R}}$? question in the negative). This is probably due to the lack of uniform lower bound methods. The currently best known bound for testing RES_n to vanish is of order $\Omega(n^3)$ ([60]). Here the problem in exploiting Theorem 4.1 is the ignorance of the multiple. To circumvent this problem in [60] there is used a special complexity measure for polynomials which is related to its computational complexity and can be transferred from a multiple of the generating polynomial to this polynomial itself. The idea behind is that

- for computing a polynomial every appearing variable has to be used at least once and
- in n steps polynomials of degree 2^n can be computed (see above). Thus it is reasonable to include into the complexity measure a term related to the logarithm of the degree (this is the approach undertaken by Strassen to prove his "degree-bound" , cf. [108]. For the resultant polynomials RES_n it yields a lower bound of order $n + log n$).

Let V_f be the number of variables a polynomial f really depends on and D_f the maximum degree of a single variable in f. Then the above together with Theorem 4.1 gives

Theorem 4.7 *([60])*

a) *At least $V_f + log_2(D_f) - 1$ many computational steps are necessary for computing a polynomial f from inputs $x_1, \ldots, x_n$.*

b) *Testing the resultant RES_n to vanish needs at least $\Omega(n^3)$ computational steps.*

Even though we have not yet dealed with computations over the complex numbers (see section 6) let's mention one further extremely interesting application of Theorem 4.1 given by Shub and Smale ([99]), which fits into the context.
As seen above the number $m := 2^{2^n}$ can be computed using $log(logm)$ operations of the form $\{+, -, *\}$. In general, this is not true for any natural number ([99]). Shub and Smale pose the question whether at least for the factorials $k!, k \in \mathbb{N}$ a similar statement holds.

Definition 4.8 *([99])* A sequence $(a_k) \in \mathbb{N}^{\mathbb{N}}$ is said *ultimately easy to compute* if there exist a constant c and non-zero integers m_k such that every element of the sequence $(a_k \cdot m_k)$ is computable in at most $(log(a_k \cdot m_k))^c$ many steps starting from the integer 1 and using operations from $\{+, -, *\}$. Otherwise it is said *ultimately hard to compute* .

Shub and Smale now prove

Theorem 4.9 *([99]) If the sequence $(k!), k \in \mathbb{N}$ is ultimately hard to compute then $P_{\mathbb{C}} \neq NP_{\mathbb{C}}$ over the complex numbers (for exact definition see section 6 below).*

The result is interesting due to the fact that it combines a problem from number-theory with the $P_{\mathbb{C}} - NP_{\mathbb{C}}$ question. One important step of the proof once more is Theorem 4.1. By some tricky further arguments it is shown that the test functions which are performed along the typical path (given by 4.1) in case that $P_{\mathbb{C}} = NP_{\mathbb{C}}$ could be used to construct a fast computation of certain multiples of $k!$.
As a further consequence it is possible to relate the $P_{\mathbb{C}} = NP_{\mathbb{C}}$ question with the problem how many integral zeros a polynomial $f \in \mathbb{Z}[t]$ admits. More explicitely, in [7] [chapter 6] it is shown that in case this number is polynomially bounded in the complexity of computing f from 1 and t it follows $P_{\mathbb{C}} \neq NP_{\mathbb{C}}$.
For a result into the direction of how fast certain integers can be computed see [77] (also Shallit and Moreira have dealt with this question, cf. [99]).

Another application of Theorem 4.1 is established in [21] in order to study the class $NC_{\mathbb{R}}$ of problems being decidable in parallel polylogarithmic time using a polynomial number of processors (i.e. BSS-machines). Within the discrete theory the exact relation between polylogarithmic parallel and polynomial time computations is still unknown. Cucker considers the problem

$$FER := \{x \in \mathbb{R}^\infty \mid size(x) = n \text{ and } x_1^{2^n} + x_2^{2^n} = 1\}$$

and shows

Theorem 4.10 *([21]) FER belongs to class $P_{\mathbb{R}}$ but is not decidable by a parallel BSS-machine using an arbitrary number of processors each working within time $log^k(n)$ ($k \in \mathbb{N}$ arbitrary, fixed).*

The reason why the theorem holds is that - apart from Theorem 4.1 - a single processor would have to compute a polynomial of degree 2^n in polylogarithmic time.
Thus over the reals it follows

Theorem 4.11 *([21])*

$$NC_{\mathbb{R}} \subsetneq P_{\mathbb{R}} \quad .$$

Beside the (somehow artificial) problem FER there are some more natural problems establishing this separation, too. In [31] the notion of $P_{\mathbb{R}}$-*complete* problems is introduced adapting from classical complexity theory the idea of *parallel polylogarithmic time (PPT) reducibility* within class $P_{\mathbb{R}}$. The following completeness result exhibits a most difficult problem in $P_{\mathbb{R}}$ w.r.t. this type of reductions :

Theorem 4.12 *([31]) The problem of deciding whether a given algebraic circuit yields 1 as result on a given input is $P_{\mathbb{R}}$-complete w.r.t. PPT-reductions.*

Together with the above separation result this implies the non-existence of a PPT-algorithm for the circuit-evaluation problem (as well as for any other $P_{\mathbb{R}}-$complete one).

Cucker extends the results to

Proposition 4.13 *([21])*

$$NP_{\mathbb{R}} \subsetneq EXP_{\mathbb{R}} \text{ and } PAR_{\mathbb{R}} \subsetneq EXP_{\mathbb{R}}$$

using the fact that (F^4, F^4_{zero}) is solvable in parallel polynomial time by an exponential number of processors ([46], [90]).

A further result related with real exponential time is given in [75]. It combines the real complexity of the NP-complete 3-Satisfiability problem with exponential time classes using methods from the discrete theory (namely "sparseness").

Proposition 4.14 *([75])*

$$3 - Satisfiability \notin P_{\mathbb{R}} \Rightarrow NEXP_{\mathbb{R}} \neq EXP_{\mathbb{R}} \ .$$

As we have seen the Knapsack problem can be used to provide some separations in restricted models. But also in the full setting a generalized version gives fruitful results : Cucker and Shub [29] investigate the relations between deterministic and nondeterministic polynomial time if the time resources are restricted to a fixed polynomial.

Theorem 4.15 *([29]) For every $d \in \mathbb{N}, d \geq 1$ deterministic polynomial time $O(n^d)$ is strictly included in nondeterministic polynomial time $O(n^d)$, i.e.*

$$DTIME_{\mathbb{R}}(O(n^d)) \subsetneq NTIME_{\mathbb{R}}(O(n^d)) \ .$$

The proof applies another lower bound result well known in algebraic complexity theory, which provides bounds in terms of the number of connected components the set to be decided has (cf. [4],[107]). Now for each d a generalized Knapsack problem is constructed which has enough connected components to yield the claim.

Let us mention that for all above separations (from Theorem 4.10 on) similar results within the Turing model are not known to be true.

The methods used so far for getting lower bounds on the time complexity of problems are closely connected with algebraic geometry. In [100] Smale investigates lower bounds on the number of <u>branches</u> (called *topological complexity*) necessary for solving one of the fundamental numerical problems, namely zero-approximation.

Theorem 4.16 *([100]) Let $f : \mathbb{C} \to \mathbb{C}$ be a polynomial of degree d , with leading coefficient 1. Then it exists a universal constant $\epsilon(d) > 0$ such that for all $0 < \epsilon < \epsilon(d)$ the topological complexity of any BSS-algorithm computing $\epsilon-$approximations to all zeros of f is at least $(log_2(d))^{\frac{2}{3}}$.*

(Even though the claim deals with complex numbers the theorem refers to real algorithms identifying $\mathbb{C}$ with $\mathbb{R}^2$).

The proof technique leads into the area of algebraic topology by relating the topological complexity with the Schwarz-genus of functions and the latter with the cup-length of certain adjoint cohomology-rings. The result itself is also interesting with respect to the question why models different to the Turing-machine are necessary to deal with numerical problems from a theoretical point of view (cf. [102]).
Theorem 4.16 has been strengthened by Vassil'ev [114] and extended to multivariate polynomials by Levine [57] and Vassil'ev [114]. Lower bounds on the topological complexity of other computational problems can be found in [10], [49] and [87].
Finally in the direction $P = NP$ let us mention that Emerson in his paper [33] proves that some

of the theorems proved by Baker, Gill and Solovay about relativization of the problem $P = NP$ are still true in the BSS-setting :
he shows the existence of oracles A, B and C such that ${P_{\mathbb{R}}}^A = {NP_{\mathbb{R}}}^A$, ${P_{\mathbb{R}}}^B \neq {NP_{\mathbb{R}}}^B$ and ${NP_{\mathbb{R}}}^C \neq co - {NP_{\mathbb{R}}}^C$.
This result is also part of recursion theory we will discuss in the next section.

5 Recursion Theory

A large part of the BSS-paper (sections 7-10) is devoted to the foundations of a recursion theory over the reals and over arbitrary ordered rings. In this section we review these foundations and subsequent developments.
Let us recall part of definition 2.3 :

Definition 5.1 Let $A \subset \mathbb{R}^\infty$ and M be a BSS-machine over $\mathbb{R}^\infty$.

a) The *output-set* of M is the set $\Phi_M(A)$. The *halting-set* of M is the set of all inputs y for which $\Phi_M(y)$ is defined.

b) The subset A is said decidable if both A and its complement A^c are halting sets.

□

Investigations in the world of halting and output sets over the reals are led with respect to classical results in the world of Turing recursion. So the authors of [10] tried to show that the main classical results still hold in their recursion theory over the reals and over an arbitrary ordered ring.
In the sequel all the results listed without references are proved in [10].
The first basic result concerning halting sets is the following :

Theorem 5.2 *Any halting set A over $\mathbb{R}$ is the countable union of basic semialgebraic sets.*

The proof is easy by tracing through the definitions of a BSS-machine M and of the running time of M. In fact it shows that for any fixed value of the running time the set of accepted inputs is a basic semialgebraic set.
The proof also shows that the result is still valid in an arbitrary ordered ring R. For logicians at least it is convenient to remark that semialgebraic sets over R are exactly the subsets which are definable over R by first-order formulas without quantifiers in the natural language for ordered rings :$(+, ., -, 0, 1, <)$.
An immediate consequence is that an halting set over $\mathbb{R}$ has at most countably many connected components (a semialgebraic set over $\mathbb{R}$ has a finite number of connected components). This allows to give examples of halting sets which are undecidable, independent on the existence of a universal BSS-machine.
For example the *Cantor Middle third set* is an halting set which is not decidable (see [10, page 8]). More generally complements of Julia sets are easily shown to be halting sets over $\mathbb{R}$. But most of them are undecidable over $\mathbb{R}$ (see [10, Section 10]). In the same area we also have to cite Sullivan's result (unpublished but see [9]) that the Mandelbrot set is undecidable over $\mathbb{R}$ (the proof uses theorem 5.2). In [17] degrees of unsolvability for Julia sets are investigated.
Theorem 5.2 can be made more precise : the countable union has to be effective, i.e it can be listed by a BSS-machine. This result is valid in any ordered ring. It implies that the coefficients which appear in the polynomial inequalities defining the semialgebraic sets lie in a finitely generated subring of the ring R. This and some easy considerations show that the set $\mathbb{Q}$ of rational numbers is a halting set over $\mathbb{R}$ but is undecidable (independently proved by Mansfield [66] and by Mercier [79], see also[81]).
Any subset S of the natural numbers can be encoded in the digits of the binary expansion of a real r in the interval $[0, 1]$: the n^th digit of r is 1 or 0 according to whether n is in S or not. This fact has as immediate consequence that, in the case of $\mathbb{R}$, the following is true :

Theorem 5.3 *([81]) A countable union of basic semialgebraic subsets of $\mathbb{R}^\infty$ is a halting set over $\mathbb{R}$ if and only if the set of real coefficients which appear in the defining inequalities (of the basic semialgebraic sets) lie in a finitely generated subring of $\mathbb{R}$.*

In particular any subset of $\mathbb{Z}$ becomes decidable over $\mathbb{R}$, any function from $\mathbb{N}$ to $\mathbb{N}$ is computable over $\mathbb{R}$([20]). In fact the trace on $\mathbb{Z}$ of the BSS-computable world over $\mathbb{R}$ is trivial : everything is computable. Let us compare with the fact that BSS-computable world over $\mathbb{Z}$ is the classical one!

In [10], a universal BSS-machine is build over an arbitrary ordered ring R. This universal machine mimics the behaviour of any BSS-machine over R. By diagonalisation, this yields an halting set being undecidable. So, just as in the classical situation, the halting problem for BSS-machines over an ordered ring is undecidable (by a BSS-machine over R).

In the classical setting this result is the crux to the proof of Gödel Theorem. It allows to find a subset of $\mathbb{N}$ which is definable in Peano Arithmetic (i.e. in the structure $< \mathbb{N}, +, . >$) but which is Turing undecidable.

In [9], Blum and Smale show that :

Theorem 5.4 *Any algebraic number ring R (i.e. a finite algebraic extension of $\mathbb{Z}$) has a subset which is first-order definable over R (in the natural language for ordered rings) but BSS-undecidable over R.*

Together with some technical results, it gives a Gödel Theorem for these subrings of $\mathbb{R}$.

Conversely assuming that R is an ordered field of infinite transcendence degree which is dense in its real closure, they show all definable (in the sense above) subsets of R are decidable over R if and only if R is real closed. The proof of this result is similar to the proof by van den Dries' Theorem which is a converse to Tarski's Theorem (see [115]).

A classical result in Turing recursion is that the class of halting sets is equal to the class of output sets (called here property HO). This is quite trivial. The same is still true over the reals but, again, this is a consequence of Tarski's Theorem (see [10]). Michaux [82] shows that the real closed fields are the only ordered rings of infinite transcendence degree which are dense in their real closure and verify this property HO (let us remark that property HO is a priori weaker that the property "all definable subsets are decidable"). Byerly [15] completes the classification of subrings of the reals with property HO by determining finitely generated subrings of the reals which have this property. Saint Jones in her dissertation [91] pursues the investigation of the relationship between definability and property HO in subrings of the reals. The results for subrings of the reals are summarized in the following theorem ":

Theorem 5.5 *A subring R of the reals has property HO iff one of the following conditions holds :*

- *R is a real closed field.*
- *R is of finite transcendence degree (over $\mathbb{Q}$) and it is a recursive ring relative to the Dedekind cuts of members of a transcendence base of R over $\mathbb{Q}$.*

The classical theorem which states that the class of Turing computable functions is the class of (partial) functions obtained from the projections, the zero function and the successor function by applying composition, juxtaposition, primitive recursion and minimalization is still valid in the BSS-setting (see [10]) with some minor rearrangements : the basic functions are the polynomial applications (with coefficients in the rings) and the characteristic function of the ordering ($\chi(x) = 1, 0, -1$ according to $x > 0$, $x = 0$ or $x < 0$; primitive recursion and minimalization are on variables the domain of which is included in $\mathbb{Z}$.

In her thesis Saint Jones proves that analogues of Kleene's s-m-n Theorem and of Rice's Theorem still hold in the BSS-context. She also builts operators similar to the classical JUMP and studies hierarchies generated by these operators in the case of the subrings of $\mathbb{R}$.

In this direction of hierarchies we have to mention earlier work by Cucker [20] (see also [67]); namely he gives a syntactical characterization of the arithmetical hierarchy over the reals, in terms of alternations of countable connectives (conjunctions and disjunctions) applied to multiindexed sets of quantifier-free formulas of the language of ordered rings. He also describes complete problems for each level of the hierarchy.

Let us end this section by mentionning several other papers. In [27] a subclass of the class of decidable sets over the reals is defined and investigated : the class TB of subsets of $\mathbb{R}^\infty$ accepted by BSS-machines whose running time is a well defined function in the input size. Closedness of

the class TB is shown under topological operators; a precision to Theorem 5.3 is brought for sets in the class TB; time hierarchies are introduced inside the class.
In [117] a relation between computability in the BSS-model and a notion of computability in category theory is shown.
In [10] [section 9] there is an attempt to give a pseudo-diophantine characterization of recursively enumerable sets over the reals. Open questions related to this are solved in [97].
We will come back to recursion in the case of BSS-theory of computation over a general structure (next section).
There is a different approach to equip the reals with a recursive structure. We just refer to [53],[119] and the litterature cited there.

6 General Structures

It should be clear that the questions treated so far not at all are necessarily restricted to the real numbers as a field. In paragraph 3 we have yet seen that changing the model - by restricting the kind of operations - can be fruitful. In this section we want to outline results for more general approaches, namely considering arbitrary domains.
Obviously, as soon as we define on any domain a set of allowed operations as well as a size- and cost-measure, complexity issues can be dealt with. (Notions of paragraph 2 transform in a straightforward manner; any element of the domain is considered as entity). This was already done in [10] for the complex numbers and for arbitrary ordered rings. In fact, the proof of Theorem 2.9 also establishes the central meaning of Hilbert-Nullstellensatz over $\mathbb{C}$ or any other algebraically closed field.

Theorem 6.1 *Over the complex numbers the "Effective Hilbert-Nullstellensatz" is $NP_{\mathbb{C}}$-complete :*

$$(EHN): \quad \begin{array}{c} \textit{given } f_1,\ldots,f_s \in \mathbb{C}[z_1,\ldots,z_n]\ ,\ deg(f_i) \le 2 \\ \textit{does there exist a joint zero of all } f_i\ ? \end{array}$$

The same holds true for any other algebraically closed field.

Note that apart from the lack of an ordering the allowed test instructions over $\mathbb{C}$ are of form : is $z = 0$?

Remark 6.2 It's well known that in (EHN) no common zero exists iff there is a decomposition $1 = \sum f_i \cdot g_i\ ,\ g_i \in \mathbb{C}[z_1,\ldots,z_n]$. This is interesting w.r.t. the question how to verify, whether $f_i = 0, 1 \le i \le n$ is <u>not</u> solvable, i.e. whether the classes $co - NP_{\mathbb{C}}$ and $NP_{\mathbb{C}}$ coincide or not. However, a result by Brownawell ([12] , see also [45], [101]) says that in general the g_i can have an exponential degree, thus just guessing the g_i and checking the above identity won't proof (EHN) to belong to $co - NP_{\mathbb{C}}$. The relation between classes $NP_{\mathbb{C}}$ and $co - NP_{\mathbb{C}}$ is still open (similar as over the reals).

An extremely interesting connection among algebraically closed fields with respect to $P \neq NP$? question is given in [7] [chapter 6]. Here it is shown that there basically is only one $P \neq NP$? problem for all these fields. The main step in the proof is the following transfer theorem about "elimation of constants":

Theorem 6.3 *([7]) Let (Y, Y_0) be a decision problem solved by a BSS-machine M over $\mathbb{C}$. Let $(Y_{\bar{Q}}, Y_{0\bar{Q}})$ be its restriction to $\bar{\mathbb{Q}}$, the algebraic closure of $\mathbb{Q}$ in $\mathbb{C}$ (i.e. $Y_{\bar{Q}} := Y \cap \bar{\mathbb{Q}}^{\infty}$ and $Y_{0\bar{Q}} := Y_0 \cap \bar{\mathbb{Q}}^{\infty}$). Then there exist a constant $c \in \mathbb{N}$ as well as a BSS-machine M' over $\bar{\mathbb{Q}}$ solving $(Y_{\bar{Q}}, Y_{0\bar{Q}})$ such that the running time $T_{M'}(y)$ of M' for all $y \in Y_{\bar{Q}}$ is bounded from above by $T_M(y)^c$.*

This result allows to conclude $P = NP$ over $\bar{\mathbb{Q}}$ if $P = NP$ over $\mathbb{C}$. Together with the inverse implication, which was shown in [84] by model-theoretic means, this results in

Theorem 6.4 *([7], [84]) Let $K \subset L$ be algebraically closed fields. Then*

$$P = NP \textit{ over } K \iff P = NP \textit{ over } L.$$

Note that the above transfer principle can also be applied to show the existence of non-complete problems in $NP_{\mathbb{C}} \setminus P_{\mathbb{C}}$, if $P_{\mathbb{C}} \neq NP_{\mathbb{C}}$ is assumed ([64]).

We have mentioned above a few $P \neq NP$ theorems for restricted BSS-models. There are further results of that type for some different kind of structures and size- resp. cost-measures. We refer to [11], [71], [93], [94] .
More general completeness results - namely for structures of finite type - have been studied independently in [41], [48] and [76] . The assumption of finite type structures ensures the existence of complete problems. The flavour of the established complete problems is that of a generalized satisfiability problem.
In the remaining of this section we will discuss these generalizations.

Definition 6.5 *A structure is a set R equipped with a set of operations, a set of relations and a set of constants. We say that such a structure is of finite type if the set of operations and the set of relations are finite.*

An example is the set of real numbers equipped with rings operations, the relation of order and constants 0 and 1. On a structure of finite type we can define BSS-machine where the basic computation instructions are given by the operations and the branching instructions are ruled by the set of relations; moreover one can extend all the definitions of subsection 2.1 . BSS-recursion theory for structures of finite type is introduced in [41], but the first aim of that paper is to dicusse a general "$P = NP$?" problem. There it is shown that the natural NP-complete problem over such a structure A is the satisfiability problem for circuits with parameters in A. This problem is undecidable in general; it is proved in [41] that :

Theorem 6.6 *NP-problems over a structure A are decidable (in the sense of BSS-computability over A) if and only if the structure admits effective quantifier elimination for first- order formulae in the natural language of the structure.*

In fact it is shown that $P = NP$ over A is true iff A admits a polynomial time BSS-algorithm which performs elimination of one block of existential quantifiers (see also [71] where trigonometric functions are added to the reals, [5] which deals with the case of BSS-theory over p-adic fields). The class DNP is also defined in this broad context. All these results are fully discussed in Poizat's book [88]. One namely finds :

Theorem 6.7 *Call structure M "standard" if it satisfies the following conditions (cf. definition 3.2) :*

- $BP(P_M) = P$
- $NP_M = DNP_M$
- *there exists a boolean problem which is NP_M-complete.*

Then in a standard structure the question $P_M = NP_M$ is equivalent to the classical question $P = NP$.

In the same model-theoretic flavour a result by Michaux [84] shows that the equality $P = NP$ is preserved by elementary extensions. The idea of relating complexity issues in a structure with those in an elementary extension is yet present in [34] and [61] .
Issues in recursion theory over structures of finite type are covered by Friedman and Mansfield in [38]: semi-algebraic characterization of halting sets, existence of a universal machine and of pairing systems,... In [1] and [3] general BSS recursion theory is also developped (s-m-n theorem is proved in full generality) but the presentation differs from [38] and [41]. In [3] the arithmetical hierarchy is investigated for general structures of finite type : this hierarchy splits into two hierarchies, one based on the halting sets, the other over the output sets.[5] Hemmerling (in [48]) also looks to general BSS-recursion for structures of finite type and reproved results with great care.
In [91] BSS-theory over the reals is discussed in relation with the Baire class 1 functions.

[5] See also [91] which deals with these hierarchies in the case of the subrings of the reals.

An interesting generalization of BSS-machines is given by Hotz, Vierke and Schieffer. Here analytic functions are included by allowing to take limits. Even though the addition of such operations may destroy decidability properties (because of the lack of quantifier elimination), interesting other features appear. One example are the relations of such machines with stability questions of dynamical systems. For this approach consider [50].

7 Descriptive complexity theory

We have seen in the last section that model theoretic aspects come into play very naturally by analyzing different operations or structures. However model theory is related to complexity theory also from a very different point of view, namely that of "descriptive complexity theory". This branch of model theory considers the logical complexity of defining a property and combines computational complexity with logical definability. It thus provides a machine-independent approach to complexity issues.
For the Turing-model this direction was taken up in the 70th by Fagin's characterization of NP as follows : a class L of finite structures is in NP iff there exists an existential second-order sentence Ψ such that L is precisely the class of finite models for Ψ (see [36], and [44] for a survey on this field). In [42] so called $\mathcal{R}$-structures are introduced which allow to built up the basic ingredients for a logical approach to complexity in the BSS-model. Let's consider the following example, which provides a "logical description" of the fact that (F^4, F^4_{zero}) belongs to $NP_{\mathbb{R}}$.

Example 7.1 Let f be a real polynomial of degree four in n unknowns ($n \in \mathbb{N}$). Consider the discrete set $A := \{0, 1, \ldots, n\}$ (A builts up the "universe" of the $\mathcal{R}$-structure to be defined) as well as the following functions :

- the unary predaccessor function $pred : A \to A$, defined by

$$pred(i) := i - 1, pred(0) := 0$$

- the constant nullary functions o and end, defined by

$$o() := 0 \in A \text{ and } end() := n \in A$$

- a function $C : A^4 \to \mathbb{R}$ which naturally defines a homogeneous degree four polynomial in $\mathbb{R}[x_0, x_1, \ldots, x_n]$, namely

$$g = \sum_{i,j,k,l} C(i,j,k,l) \cdot x_i \cdot x_j \cdot x_k \cdot x_l$$

  The arbitrary polynomial f can be obtained by setting $x_0 = 1$ in the appropriate polynomial g.

The above is a typical example of an $\mathcal{R}$-structure . It consists of

- the primary part $\mathcal{A}$: a (for our purposes) generally ordered universe A together with a finite set Ψ_a of functions and relations on it (this means that the primary part is a "finite structure" , cf. [44])
- the secondary part $\mathcal{R}$: the set $\mathbb{R}$ of real numbers together with its basic arithmetic operations $(\mathbb{R}, +, -, *, :, \leq, (c_r)_{r\in\mathbb{R}})$ (where every element $r \in \mathbb{R}$ is assumed to be named by a constant c_r)
- a finite set W of functions $\Psi_w : A^k \to \mathbb{R}$

The triple $\mathcal{D} := (\mathcal{A}, \mathcal{R}, \mathcal{W})$ is called an $\mathcal{R}$-structure of vocabulary (Ψ_a, Ψ_w). The interpretation of the relations and functions generally varies with the universe A (which makes it possible to relate decision-problems with $\mathcal{R}$-structures). The basic feature of such $\mathcal{R}$-structures is given by the possibility to strictly separate the "discrete" part of BSS-machines (like indices of tuples, time, indices of registers and the finite control of a machine) from its real-arithmetic part (as machine-constants, computation).

Now in order to deal with complexity issues different logics on $\mathcal{R}$-structures are considered in [42]. These logics again strongly reflect the above mentioned distinction of discrete and continuous aspects.

Example 7.1 *(continuation)* We again consider the description of a degree four polynomial f given as a $\mathcal{R}$-structure $\mathcal{D}$. Those polynomials belonging to F^4_{zero} can be characterized by the following sentence :

$$\Psi : \ (\exists X : A \to \mathbb{R})(\exists Y : A^4 \to \mathbb{R}) \ (X(0) = 1 \wedge Y(\overline{0}) = C(\overline{0}) \wedge$$
$$Y(\overline{n}) = 0 \wedge \forall \ \overline{u} \in A^4(\overline{u} \neq \overline{0} \to Y(\overline{u}) = Y(pred^*(\overline{u})) + C(\overline{u}) \prod_{i=1}^{4} X(u_i))\Big).$$

Here the function X represents a zero of f ; Y sums up the values of all monomials of f when X is plugged in. The function $pred^* : A^4 \to A^4$ defines the predaccessor of any 4-tuple in A if the ordering on A is extended in an obvious way (i.e. lexicographically) to A^4. It can be easily seen that such an extension on A^k is FO-definable for any arity k. Obviously the $\mathcal{R}$-structure $\mathcal{D}$ satisfies Ψ if and only if $f \in F^4_{zero}$. □

The example includes the basic ingredients which built up first and second order sentences over $\mathcal{R}$-structures : In order to maintain the difference of discrete and continuous aspects, first order logics allows all real arithmetic operations included in the second part of the $\mathcal{R}$-structure, but quantifying only variables taking values in the first part. Similarly, second-order logic quantifies functions and relations within the universe as well as from the universe into $\mathbb{R}$ (but not functions over real variables).
In fact this is the right model theoretic approach to capture real complexity classes. The given example is just a special case of the following

Theorem 7.2 *([42]) Let (F, F_{yes}) be a decision problem of $\mathcal{R}$-structures . Then $(F, F_{yes}) \in NP_{\mathbb{R}}$ iff there exists an existential second-order structure Ψ such that F_{yes} is the set of all models $\mathcal{D}$ for Ψ .*

Beside the above theorem further characterizations can be obtained. Hence for example the class $P_{\mathbb{R}}$ can be captured by introducing a fixed-point logic on $\mathcal{R}$-structures . Also stronger completeness results for classes $NP_{\mathbb{R}}$ and $P_{\mathbb{R}}$ can be established by consideration of so called first-order reductions. Another interesting remark refers to the meaning of space resources : even though Michaux showed the use of space alone does not provide reasonable complexity classes over $\mathbb{R}$, it turnes out that space- and time-restrictions together capture important classes. For example polynomial time together with constant space is captured by a special class of functions over $\mathcal{R}$-structures, which in the according discrete setting exactly defines *logspace* computations. Thus descriptive complexity theory can be used to define analogues of (at least some) space-classes also in the BSS-approach. For more details see [42].

8 Concluding Remarks

In this final section we want to stress on directions which are yet outlined in [10] but have not been mentioned so far.
In order to use the above theory also for a structural analyzation of numerical algorithms, ideas like round-off errors and approximate solutions should be incorporated. Similarly, probabilistic features could be considered by taking into account either input spaces equipped with a probability measure or probabilistic BSS-machines.
Steps into the latter direction are worked out in [28] and [24] . Real analogues of well known discrete probabilistic complexity classes are introduced. In [28] this is done by changing the accepting conditions of a nondeterministic BSS-machine according to a probability measure on the space of guesses. Some very interesting results are proved in [24](see also [54]) . Here real analogues of the classes ZPP, R, PP and BPP are defined over $DNP_{\mathbb{R}}$ (i.e. the guessing space is reduced

to $\{0,1\}^*$ and equipped with a finite distribution for every dimension n.) In this framework the inclusions

$$ZPP_{\mathbb{R}} \subset R_{\mathbb{R}} \subset DNP_{\mathbb{R}} \subset PP_{\mathbb{R}} \subset PAR_{\mathbb{R}}$$

hold. Furthermore the authors show

$$P_{\mathbb{R}} = BPP_{\mathbb{R}} ,$$

i.e. randomization with bounded probability error doesn't increase the power of polynomial time decision machines. The proof relies on a simulation result of probabilistic circuits by deterministic ones heavily using the ability to code an arbitrary subset of integers as a single real. Also lower bounds on the sizes of such simulating circuits are given. Finally for equational probabilistic machines the boolean parts of all above mentioned classes capture their discrete counterparts.

Often decision problems are also closely connected to search problems. That is beside deciding a problem one is also interested in finding a solution. However, since solutions cannot be expected to be computable in general one is forced to study approximate algorithms. As a typical example zeros of polynomials in general are not given as rational functions of the coefficients (see [95]).
A formal notion of real algorithms solving problems approximately is given in [10] and [101]. Then naturally also terms like conditioning come into play. This leads directly into the scope of numerical analysis. We thus only cite very briefly some work connected with the BSS model. Of course classical problems and algorithms in the field of numerical analysis can be revisited in that context.
For the problem of computing polynomial zeros within a certain accuracy the above ideas are worked out by Shub and Smale in a series of papers (up to now five, see [98] for the first of it and also [93]). Regarding polynomial systems over $\mathbb{C}$ generically having roots they perform an extensive analysis on the real complexity of following a homotopy by a (projective) Newton method. This analysis includes dependency on the approximation rate, the conditioning of the system as well as probabilistic results.
Lower bounds on the zero-approximation problem have yet been mentioned in section 4.
Incorporation of round-off errors for Newton's method using the BSS model was also studied. Because these topics leave the scope of our paper we just refer to [63] (see also [65] for a general discussion). The same holds true for the important problem of practically implementable algorithms solving polynomial equations. We once again refer to [92].

We have tried to present the main research streams which appeared so far in the area of real structural complexity theory. Depending on the questions one is engaged in this leads to problems in many different disciplines - one of the especially interesting features of the BSS-theory.
Let's close with the following list of papers which also provide introductory expositions and remarks concerning real complexity theory : [6], [22], [55], [70], [74], [83], [85], [86], [94], [95], [101], [102], [110] .

A current bibliography is collected in [48]. Finally we want again to stress on the forthcoming book by Blum, Cucker, Shub, and Smale [7], which will give an extensive treatment of the area. Its first chapter provides an introduction into the ideas of real number complexity theory and will be published separately ([8]).

Acknowledgement We want to thank T. Lickteig for helpful comments on a previous version of this paper.

References

[1] I.V. Ashaev, V.Y. Belyaev, A.G. Myasnikov, *Approaches to the theory of generalized computability,* preprint, Omsk State University, 1992.

[2] J.L. Balcázar, J. Diaz, J. Gabarró, *Structural Complexity I,II,* EATCS Monographs of Theoretical Computer Science **11** and **22** (1988,1990), Springer-Verlag

[3] V.Y. Belyaev, A.G. Myasnikov, *Theorems on hierarchy in generalized recursion theory,* preprint, Omsk State University, 1992

[4] M. Ben-Or, *Lower bounds for algebraic computation trees*, in : Proceedings 15th ACM STOC (1983), 80-86

[5] E. Bishop, *Machines and Complexity over the p-adic Numbers*, PhD-thesis, University of Berkeley (1991)

[6] L. Blum, *A Theory of Computation and Complexity over the real numbers*, Proceedings of the International Congress of Mathematicians Kyoto 1990 **II**, Math. Soc. Japan, Tokyo (1991), 1491-1507, Springer Verlag

[7] L. Blum, F. Cucker, M. Shub, S. Smale, *Complexity and real computation*, forthcoming book, Springer Verlag.

[8] L. Blum, F. Cucker, M. Shub, S. Smale, *Complexity and real Computation : A Manifesto*, to appear in Int. J. of Bifurcation and Chaos

[9] L. Blum , S. Smale, *The Gödel Incompleteness Theorem and Decidability over a Ring*, in : From Topology to Computation , Proceedings of the Smalefest, Springer - Verlag (1993), 321-339

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

[11] V.E. Brimkov, S.S. Danchev, *Real Data-Integer Solution Problems within the Blum-Shub-Smale Computational Model*, Institute of Mathematics, Bulgarian Academy of Science, Preprint 1995

[12] W.D. Brownawell, *Bounds for the degrees in the Nullstellensatz*, Annals of Mathematics **126** (1987), 577 - 591

[13] P. Bürgisser, M. Clausen, A. Shokrollahi, *Algebraic Complexity Theory*, forthcoming

[14] P. Bürgisser, T. Lickteig, M. Shub, *Test Complexity of Generic Polynomials*, Journal of Complexity **8** (1992), 203 - 215,

[15] R.E. Byerly, *Ordered subrings of the reals in which output sets are recursively enumerable*, Pro. Amer. Math. Soc., **118**,2 (1993), 597-601

[16] J. Canny, I. Emiris, *An efficient algorithm for the sparse mixed resultant*, Proceedings AAECC, Puerto Rico 1993, Lecture Notes in Computer Science **263**, Springer Berlin (1993), 89 - 104

[17] C.T. Chong, *Positive Reducibility of the Interior of Filled Julia Sets*, Journal of Complexity **10** (1994), 437-444

[18] S.A. Cook, *The complexity of the theorem proving procedures*, Proceedings 3rd ACM STOC 1971, 151-158

[19] M. Cosnard, M. Matamala, *ON NC-Real Complexity classes for additive circuits and their relations with NC*, CRM Barcelona, Preprint No. 221, 1993

[20] F. Cucker, *The arithmetical hierarchy over the reals*, Journal of Logic and Computation **2**, no. 3 (1992), 375-395

[21] F. Cucker, $P_{\mathbb{R}} \neq NC_{\mathbb{R}}$, Journal of Complexity **8** (1992), 230-238

[22] F. Cucker, *On the complexity of quantifier elimination : The structural approach*, The Computer Journal **36** No. 5 (1993), 400-408

[23] F. Cucker, D.Y. Grigoriev, *On the power of real turing machines over binary inputs*, to appear in : SIAM Journal on Computing

[24] F. Cucker, M. Karpinski, P. Koiran, T. Lickteig, T. Werther, *On real Turing machines that toss coins*, in : Proceedings of the 27^{th} Symposium on Theory of Computing (1995), 335-342

[25] F. Cucker , P. Koiran, *Computing over the reals with addition and order. Higher complexity classes*, Journal of Complexity **11**,No.3 (1995), 358-376

[26] F. Cucker , M. Matamala, *On weak non-determinism*, to appear in : Math. Systems Theory.

[27] F. Cucker , J.L. Montaña , L.M. Pardo, *Time Bounded Computations over the Reals*, Int. Journal of Algebra and Computation **2**, No.4 (1992), 395-408

[28] F. Cucker, F.Rossello, *On the complexity of some problems for the Blum-Shub-Smale model*, Proc. LATIN'92, Lecture Notes in Computer Science **583** (1992), 117-129

[29] F. Cucker , M. Shub, *Generalized Knapsack problems and fixed degree separation*, to appear in : Theoretical Computer Science

[30] F. Cucker, M. Shub, S.Smale, *Separation of complexity classes in Koiran's weak model*, Theoretical Computer Science **133** (1994), 3-14

[31] F. Cucker , A. Torrecillas, *Two P - complete problems in the theory of the reals*, in : Automata, Languages and Programming , Proceedings of the 18th International Colloquium Madrid , Spain , July 8-12 , 1991 , J. Leach Albert, B. Monien , M. Rodriguez Artalejo (Eds.) (1991)

[32] J.H. Davenport, J Heintz, *Real quantifier elimination is doubly exponential*, Journal of Symbolic Computation **5** (1988), 29-35

[33] T. Emerson, *Relativization of the P=?NP question over the reals (and other orderd rings)*, Theoretical Computer Science **133** (1994), 15-22

[34] B.C. Eaves, U.G. Rothblum, *A theory on extending algorithms for parametric problems*, Mathematics of Operations Research **14**, 3 (1989), 502-533

[35] B.C. Eaves, U.G. Rothblum, *Dines - Fourier - Motzkin quantifier elimination and an application of corresponding transfer principles over ordered fields*, Mathematical Programming **53** (1992) , 307 - 321

[36] R. Fagin, *Generalized first-order spectra and polynomial-time recognizable sets*, SIAM-AMS Proceedings **7** (1974), 43-73

[37] J. Ferrante, C. Rackoff, *A Decision Procedure for the First Order Theory of Real Addition with order*, SIAM Journal Comput. **4** No.1 (1975), 69 - 76

[38] H. Friedman and R. Mansfield, *Algorithmic procedures*,Trans. A.M.S. **332**, 1 (1992), 297-312

[39] M. Garey, D. Johnson, *Computers and Intractability*, Freeman and Company (1979), New York

[40] C. Gassner, *On NP-completeness for Linear Machines*, preprint 1996

[41] J.B. Goode, *Accessible telephone directories*, Jour. of Symb. Logic, **59**, 1 (1994), 92- 105

[42] E. Grädel, K. Meer, *Descriptive Complexity Theory over the Real Numbers*, in : Proceedings of the 27^{th} Symposium on Theory of Computing (1995), 315-324; full paper version to appear in: Proceedings of the AMS Summer Seminar on "Mathematics of Numerical Analysis: Real Number Algorithms", Park City 1995, Lectures in Applied Mathematics, ed.: J. Renegar, M. Shub, S. Smale

[43] D.Y. Grigor'ev, *Complexity of Deciding Tarski Algebra*, Journal of Symbolic Computation **5** (1988), 65 - 108

[44] Y. Gurevich, *Logic and the challenge of computer science*, in: E. Boerger (ed.), Trends in Theoretical Computer Science, Computer Science Press (1988), 1-57

[45] J. Heintz, J. Morgenstern, *On the intrinsic Complexity of Elimination Theory*, Journal of Complexity **9** (1993), 471-498

[46] J. Heintz, M.F. Roy, P. Solerno, *On the complexity of semialgebraic sets*, Proceedings IFIP 1989, San Fracisco, North-Holland (1989), 293 - 298

[47] A. Hemmerling, *On Genuine Complexity and Kinds of Nondeterminism*, J. Inform. Process. Cybernet. EIK **30 (2)** (1994), 77-96

[48] A. Hemmerling, *Computability and Complexity over Structures of Finite Type*, E.M.Arndt-University Greifswald, Preprint **2** (1995)

[49] M.D. Hirsch, *Applications of Topology to Lower bound estimates*, in : From Topology to Computation, Proceedings of the Smalefest, Springer-Verlag, New-York, Berlin (1993), 395-418

[50] G. Hotz, G. Vierke, B. Schieffer, *Analytic Machines*, Electronic Colloquium on Computational Complexity **95-025** (1995), available via ftp.eccc.uni-trier.de:/pub/eccc/

[51] N. Karmarkar, *A new polynomial time algorithm for linear programming*, Combinatorica **4** (1984), 373-395

[52] L.G. Khachyan, *A polynomial algorithm in linear programming*, Dokl. Akad. Nauk USSR **244** (1979), 1093-1096; english translation in: Soviet Math. Dokl. **20** (1979), 191-194

[53] Ker-I Ko, *Complexity of real functions*, Birkhäuser, Basel-Berlin-Boston (1991)

[54] P. Koiran, *A weak version of the Blum-Shub-Smale model*, FOCS'93 (1993), 486-495 and NeuroCOLT TR Series **NC-TR-94-5** (1994) [6]

[55] P.Koiran, *Puissance de calcul de réseux de neurones artificiels*, PhD-thesis, ENS Lyon (1993)

[56] P. Koiran, *Computing over the reals with addition and order*, Theoretical Computer Science **133** (1994), 35-47

[57] H. Levine,*A lower bound for the topological complexity of Poly(D,n)*, Journal of Complexity **5** (1991), 34 - 44

[58] T. Lickteig, *On semialgebraic decision complexity, TR-90-052 ICSI, Berkeley und Univesität Tübingen, Habilitationsschrift* (1990)

[59] T. Lickteig, *Semi-algebraic Decision Complexity, the Real Spectrum and Degree*, to appear in : Journal of pure and applied algebra (1994)

[60] T. Lickteig, K. Meer, *A note on testing the resultant*, Journal of Complexity **11** 3 (1995), 344-351

[61] W. Maass, *On the use of inaccessible numbers and order indiscernibles in lower bound arguments for random access machines*, Jour. of Symb. Logic **53**, 4 (1988), 1098–1109

[62] W. Maass, *Computing on Analog Neural Nets with Arbitrary Real Weights*, in : Theoretical Advances in Neural Computation and Learning, V.P.Roychowdhury, K.Y.Siu, A.Orlitsky, editors, Kluwer Academic Publishers (Boston) (1994), 153-172

[63] G. Malajovich, *On generalized Newton algorithms : quadratic convergence, path-following and error analysis*, Theoretical Computer Science **133** (1994), 65-84

[64] G. Malajovich, K. Meer, *On the structure of* $NP_{\mathbb{C}}$, preprint 1995

[65] R. Mansfield, *A complete axiomatization of computer arithmetic*, Math. Comp. **42** (1984), 623-635

[6] both references are cited because the NeuroCOLT version contains some further results.

[66] R. Mansfield, *The irrationals are not recursively enumerable*, Proc. Amer. Math. Soc. **110** (1990), 495-497

[67] A.R. Mathias, Comparison of the projective and non-deterministic hierarchies, preprint 1994

[68] K. Meer, *Computations over $\mathbb{Z}$ and $\mathbb{R}$: a comparison*, Journal of Complexity **6** (1990) , 256-263

[69] K. Meer, *A note on a $P \neq NP-$ result for a restricted class of real machines*, Journal of Complexity **8** (1991), 451-453

[70] K. Meer, *Komplexitätsbetrachtungen für reelle Maschinenmodelle*, PhD-thesis, RWTH Aachen, Verlag Shaker-Aachen, ISBN 3-86111-385-6 (1993)

[71] K. Meer, *Real number models under various sets of operations*, Journal of Complexity **9** (1993), 366-372

[72] K. Meer, *On the complexity of Quadratic Programming in real number models of computations*, Theoretical Computer Science **133** (1994), 85 - 94

[73] K. Meer, *Real number models : On the use of information*, Journal of Symbolic Computation **18** (1994), 199-206

[74] K. Meer, *Eine Einführung in die reelle Komplexitätstheorie*, Lecture note manuscript, RWTH Aachen (1995)

[75] K. Meer, *On the relations between discrete and continuous complexity theory*, Mathematical Logic Quaterly **41** (1995), 281-286

[76] N. Megiddo, *A general NP-completeness Theorem*, in : From Topology to Computation, Proceedings of the Smalefest, Springer-Verlag, New-York, Berlin (1993), 432-442

[77] W. de Melo, B. Fux-Svaiter, *The cost of computing integers*, preprint 1995

[78] F. Meyer auf der Heide, *A polynomial search algorithm for the n-dimensional Knapsack problem*, Journal of the ACM **31(3)** (1984), 668-676

[79] D. Mercier, *Les machines sur $\mathbb{R}$ (d'après Blum,Shub and Smale)*, mémoire de licence, Université de Mons-Hainaut, 1989

[80] C. Michaux, *Une remarque à propos des machines sur $\mathbb{R}$ introduites par Blum, Shub et Smale*, C.R. Acad. Sci. Paris, **t. 309** série I (1989), 435 - 437

[81] C. Michaux, *Machines sur les réels et problèmes NP-complets (d'après L. Blum, M. Shub, S. Smale et al.)*, Séminaire des structures algébriques ordonnées 1988-1989, prépublication de l'équipe de Logique Mathématique de Paris 7, **2** (1990), 30 pages

[82] C. Michaux, *Ordered rings over which output sets are recursively enumerable*, Proc. Amer. Math. Soc. **112** (1991), 569-575

[83] C. Michaux, *Differential fields , Machines over the real numbers and Automata*, PhD-thesis, Université de Mons Hainaut, Faculte des Sciences, 1991

[84] C. Michaux, *$P \neq NP$ over the nonstandard reals implies $P \neq NP$ over $\mathbb{R}$*, Theoretical Computer Science **133** (1994), 95-104

[85] E. Novak, *Complexity of continuous problems - an introduction*, Lecture note manuscript, University of Erlangen (1993)

[86] E. Novak, *The real number model in numerical analysis*, Journal of Complexity **11** (1995), 57-73

[87] E. Novak, H. Woźniakowski, *Topological complexity of zero finding*, preprint 1996

[88] B. Poizat, *Les Petits Cailloux, une approche modèle-théorique de l'Algorithmie*, Aléas 1995

[89] J. Renegar, *Computational Complexity of Solving Real Algebraic Formulae*, Proceedings of the International Congress of Mathematicians Kyoto 1990 **I,II**, Math. Soc. Japan, Tokyo (1991), 1595 - 1606

[90] J. Renegar, *On the computational Complexity and Geometry of the first-order Theory of the Reals , I - III*, Journal of Symbolic Computation **13** (1992), 255 - 352

[91] R. Saint Jones, *Theory of Computation for the Real Numbers and Subrings of the Real Numbers Following Blum/Shub/Smale*, Dissertation, University of Berkeley, 1995

[92] A. Schönhage, *Equations solving in terms of computational complexity*, Proceedings of the International Congress of Mathematicians Berkeley 1986 **I**, 131-153

[93] M. Shub, *Some Remarks on Bezout's Theorem and Complexity Theory*, in : From Topology to Computation, Proceedings of the Smalefest, Springer-Verlag, New-York, Berlin (1993), 443-455

[94] M. Shub, *On the Work of Steve Smale on the Theory of Computation*, in : From Topology to Computation, Proceedings of the Smalefest, Springer-Verlag, New-York, Berlin (1993), 281-301

[95] M. Shub, *Mysteries of Mathematics and Computation*, Mathematical Intelligencer **16(1)** (1994), 10-15

[96] K. Skandalis, *Programmable real numbers and functions*, Fundamenta Informaticae **7**,1 (1984), 27-56

[97] K. Skandalis, *On the characterization of recursively enumerable sets as pseudo-diophantine*, Proceedings of the American Mathematical Society **123** 2 (1995), 555-558

[98] M. Shub, S. Smale, *Complexity of Bezout's Theorem I : Geometric aspects*, Journal of the AMS **6** (1993), 459-501

[99] M. Shub, S. Smale, *On the Intractability of Hilbert's Nullstellensatz and an algebraic version of* $''NP \neq P?''$, to appear in : Duke Journal

[100] S. Smale, *On the Topology of Algorithms I*, Journal of Complexity **3** (1987), 81 - 89

[101] S. Smale, *Some Remarks on the Foundations of Numerical Analysis*, SIAM Review **32** No.2 (1990), 211 - 220

[102] S. Smale, *Theory of computation*, in : Mathematical Research Today and Tomorrow, ed.: M.Castellet et al. , LN in Mathematics **1525** (1991), 59-69

[103] E.D. Sontag, *Real addition and the polynomial hierarchy*, Information Processing Letters **20** (1985), 115-120

[104] E.D. Sontag, H. Siegelmann, *Neural Networks with real weights : analog computational complexity*, Report SYCON-92-05, Rutgers University (1992)

[105] L. Stockmeyer, *The polynomial hierarchy*, Theoretical Computer Science **3** (1976), 1-22

[106] L. Stockmeyer, *Classifying the computational complexity of problems*, Jour. of Symb. Logic **52**,1 (1987), 1-43

[107] J.M. Steele , A.C. Yao, *Lower Bounds for Algebraic Decision Trees*, Journal of Algorithms **3** (1982), 1 - 8

[108] V.Strassen, *Algebraische Berechnungskomplexität*, in : Perspectives in Mathematics , Anniversary of Oberwolfach (1984), 509 - 550

[109] A. Tarski, *A decision method for elementary algebra and geometry*, 2nd edition, Univ. Calif. Press, Berkeley (1951)

[110] J. Traub, *On Smale's work in the theory of Computation : From polynomial zeros to Continuous Complexity*, in : From Topology to Computation, Proceedings of the Smalefest, Springer-Verlag, New-York, Berlin (1993), 317-320

[111] J. Traub, H. Woźniakowski, *Complexity of linear programming*, Operations Research Letters **1**,2 (1982), 59-62

[112] E. Triesch, *A note on a Theorem of Blum, Shub, and Smale*, Journal of Complexity **6** (1990), 166 - 169

[113] V.A. Vassil'ev, *Cohomology of Braid Groups and Complexity*, in : From Topology to Computation, Proceedings of the Smalefest, Springer-Verlag, New-York, Berlin (1993), 359-367

[114] S.A. Vavasis, *Nonlinear Optimization, Complexity Issues*, International Series of Monographs on Computer Science **8**, Oxford University Press (1991), New York - Oxford

[115] L. van den Dries, *Alfred Tarski's elimination theory for real closed fields*, Jour. of Symb. Logic **53**, 1 (1988), 7–19

[116] B.L. van der Waerden, *Moderne Algebra I / II*, Springer Verlag Berlin (1931)

[117] S. Vigna, *On the relations between distributive computability and the BSS-model*, to appear in : Theoretical Computer Science

[118] J. von zur Gathen, *Algebraic Complexity Theory*, Ann. Review of Computer Science **3** (1988), 317 - 347

[119] K. Weihrauch, *A simple Introduction to Computable Analysis*, Informatik-Berichte FernUniversität Hagen 171-2 (1995)