View metadata, citation and similar papers at core.ac.uk

brought to you by CORE

provided by Elsevier - Publisher Connector

Available online at www.sciencedirect.com

ScienceDirect

Electronic Notes in Theoretical Computer Science

Electronic Notes in Theoretical Computer Science 221 (2008) 71–83

www.elsevier.com/locate/entcs

On Relativizations of the P =? NP Question for Several Structures

Christine Gaßner[1,2]

Institut für Mathematik und Informatik, Ernst-Moritz-Arndt-Universität, F.-L.-Jahn-Straße 15 a, 17487 Greifswald, Germany

Abstract

We consider the uniform model of computation over arbitrary structures with two constants. For several structures, including structures over the reals, we construct oracles which imply that the relativized versions of P and NP are equal or are not equal. Moreover we discuss some special features of these oracles resulting from the undecidability of halting problems in order to explain the difficulties to define structures of finite signature which satisfy P = NP. We show that there are oracles which lose their non-deterministic self-reducibility which is sufficient for a recursive definition if their elements are compressed to tuples of fixed length.

Keywords: BSS machines, oracle machines, relativizations, P-NP problem, Halting Problem

1 Introduction

The uniform model of computation over arbitrary algebraic structures $\mathbb{K}$ can be defined in analogy to the BSS model over the real numbers introduced by L. Blum, M. Shub, and S. Smale [5, 4]. For the structure $\mathbb{K}_{\{0,1\}} =_{\mathrm{df}} (\{0,1\}; 0, 1; ; =)$ which is also the basic structure for Turing machines (compare [2]) and for structures like the ordered ring of reals used in case of the BSS model, questions like $\mathrm{P} \stackrel{?}{=} \mathrm{NP}$ are open. For the classical setting, T. Baker, J. Gill, and R. Solovay [1] constructed relativized versions of P and NP which imply different relationships between these classes. There are oracles $\mathcal{O}$ such that the classes $\mathrm{P}^{\mathcal{O}}$ and $\mathrm{NP}^{\mathcal{O}}$ are equal and other oracles such that they are not equal. T. Emerson [10] transferred these results to the ring of reals and other ordered rings. In the classical setting, the proofs rely on the enumerability of the programs of oracle machines. Emerson introduced oracles of a new kind where he used the codes of BSS machines as specified in [5]. In this way the authors showed that, in both settings, for Turing machines as well as for

[1] Email: gassnerc@uni-greifswald.de

[2] I thank Robert Bialowons and Rainer Schimming for helpful hints and I thank the referee for his comments.

1571-0661 © 2008 Elsevier B.V. Open access under CC BY-NC-ND license.

doi:10.1016/j.entcs.2008.12.008

BSS machines, the extension of the machines by oracles is not very useful for solving the central problems like $\mathrm{P} \stackrel{?}{=} \mathrm{NP}$. This implies questions like the following for any structures $\mathbb{K}$: Which relationships between the relativized versions of $\mathrm{P}_{\mathbb{K}}$ and $\mathrm{NP}_{\mathbb{K}}$ will we obtain if we permit oracles for machines over $\mathbb{K}$? Can we provide evidence that the construction of new oracles is not really helpful for solving the $\mathrm{P} \stackrel{?}{=} \mathrm{NP}$ problem, by defining oracles $\mathcal{O}$ and $\mathcal{Q}$ satisfying $\mathrm{P}_{\mathbb{K}}^{\mathcal{O}} = \mathrm{NP}_{\mathbb{K}}^{\mathcal{O}}$ and $\mathrm{P}_{\mathbb{K}}^{\mathcal{Q}} \neq \mathrm{NP}_{\mathbb{K}}^{\mathcal{Q}}$ for structures for which the relation between $\mathrm{P}_{\mathbb{K}}$ and $\mathrm{NP}_{\mathbb{K}}$ is known? Is it possible to derive new relations of fixed arity from these oracles in order to get $\mathrm{P}_{\mathbb{M}} = \mathrm{NP}_{\mathbb{M}}$ for new structures $\mathbb{M}$?

2 The Model of Computation

Let $\mathsf{struc}(U)$ be the class of structures $\mathbb{K} = (U; (d_j)_{j\in J_0}; (f_j)_{j\in J_1}; (R_j)_{j\in J_2}, =)$ with the constants $d_j \in U$, the operations f_j, and the relations R_j. Any of these operations, f_j, has some fixed arity $n_{f_j} \geq 1$ and any relation R_j has some fixed arity n_{R_j}. For any $\mathbb{K} \in \mathrm{struc}(U)$, we define the $\mathbb{K}$-machines in analogy to [5, 24, 11] such that we get a natural format of abstract computers for this kind of structures, on the one hand, and such that one has to consider only a small number of kinds of instructions, on the other hand.

Every $\mathbb{K}$-machine $\mathcal{M}$ is equipped with registers $Z_1, Z_2, \ldots$ for the elements of U and with a fixed number of registers $I_1, I_2, \ldots, I_{k_{\mathcal{M}}}$ for indices in $\mathbb{N}^+ = \mathbb{N} \setminus \{0\}$. For an input $(x_1, \ldots, x_n) \in U^\infty =_{\mathrm{df}} \bigcup_{i=1}^{\infty} U^i$, the sequence $x_1, \ldots, x_n, x_n, x_n, \ldots$ is assigned to the registers $Z_1, Z_2, \ldots$. The index registers get the content n. After the input the machine executes its program defined by a finite sequence of labelled instructions until an output instruction is reached. The *computation*, *copy*, and *branching* instructions have the form $Z_j := f_k(Z_{j_1}, \ldots, Z_{j_{n_{f_k}}})$, $Z_j := d_k$, $Z_{I_j} := Z_{I_k}$, and *if cond then goto* l_1 *else goto* l_2 where *cond* can be of the form $Z_j = Z_k$ or $R_k(Z_{j_1}, \ldots, Z_{j_{n_{R_k}}})$. The $\mathbb{K}$-machines perform these instructions as a computer. Each function and each relation of $\mathbb{K}$ is processed within a fixed time. The index registers are used in the copy instructions. For useful copying, we also allow $I_j := 1$, $I_j := I_j + 1$, and *if* $I_j = I_k$ *then goto* l_1 *else goto* l_2. Moreover, *oracle machines* can execute *if* $(Z_1, \ldots, Z_{I_1}) \in \mathcal{O}$ *then goto* l_1 *else goto* l_2 for some oracle $\mathcal{O} \subseteq U^\infty$. The *non-deterministic* machines are able to guess an arbitrary number of arbitrary elements $y_1, \ldots, y_m \in U$ in one step after the input and to assign the guesses to $Z_{I_1+1}, \ldots, Z_{I_1+m}$. Note, that we do not restrict the domain for m to simplify matters. m is independent of n. However, a machine can use at most t guesses within t steps. In any case, the *size* of an input $(x_1, \ldots, x_n)$ is, by definition, its length n. If the *output* instruction is reached, then $(Z_1, \ldots, Z_{I_1})$ is the output and the machine halts.

Let $\mathsf{M}_{\mathbb{K}}$ and $\mathsf{M}_{\mathbb{K}}^{\mathrm{N}}$ be the sets of deterministic and non-deterministic $\mathbb{K}$-machines, respectively. Let, moreover, the machines in $\mathsf{M}_{\mathbb{K}}(\mathcal{O})$ and $\mathsf{M}_{\mathbb{K}}^{\mathrm{N}}(\mathcal{O})$ be able to use the oracle $\mathcal{O}$.

Let us assume in the following that the considered structures contain two constants $a = d_1$ and $b = d_2$. We denote the class of these structures by $\mathsf{struc}_{a,b}(U)$.

Then we say that a deterministic $\mathbb{K}$-machine *accepts* (or *rejects*, respectively) a tuple $\vec{x} \in U^\infty$ if the machine outputs a (or b, respectively) on input $\vec{x}$. A $\mathbb{K}$-machine $\mathcal{M}$ *accepts* an input $(x_1, \ldots, x_n) \in U^\infty$ *non-deterministically* if there is some finite sequence of guesses $(y_1, \ldots, y_m) \in U^\infty$ such that $\mathcal{M}$ outputs a on input $(x_1, \ldots, x_n)$ for the guesses $y_1, \ldots, y_m$. The execution of one instruction is one step of the computation process. That means that each step can be executed in a fixed time unit and that the cost of an instruction is 1. A $\mathbb{K}$-machine will come to a halt *in polynomial time* if there is a polynomial p such that, on every input $(x_1, \ldots, x_n) \in U^\infty$ (and for any guesses), the machine performs at most $p(n)$ instructions before the output is generated. The *decidability* and the *recognition* (or *semi-decidability*) of a problem $\mathcal{P}$ *over* $\mathbb{K}$ results from the computability of its (partial) characteristic function $f_\mathcal{P} : U^\infty \to \{a, b\}$ by some $\mathbb{K}$-machine.

For any structure $\mathbb{K}$, let $\mathrm{P}_\mathbb{K}$ and $\mathrm{NP}_\mathbb{K}$ denote the usual complexity classes of decision problems $\mathcal{P} \subseteq U^\infty$ decided or non-deterministically recognized by a machine in $\mathsf{M}_\mathbb{K}$ or in $\mathsf{M}^{\mathrm{N}}_\mathbb{K}$ in polynomial time (where an input is only accepted if and only if it is in $\mathcal{P}$). $\mathrm{DEC}_\mathbb{K}$ contains all problems decided by a machine in $\mathsf{M}_\mathbb{K}$. For any oracle $\mathcal{O}$, $\mathrm{P}^\mathcal{O}_\mathbb{K}$, $\mathrm{NP}^\mathcal{O}_\mathbb{K}$, and $\mathrm{DEC}^\mathcal{O}_\mathbb{K}$ denote the classes extended to machines which can also use $\mathcal{O}$.

Let $\mathsf{struc}^{\mathsf{fin}}_{a,b}(U)$ be the class of structures of finite signature of the form $(U; a, b, d_3, \ldots, d_{k_0}; f_1, \ldots, f_{k_1}; R_1, \ldots, R_{k_2}, =)$ for some $k_0 \geq 2$ and $k_1, k_2 \geq 0$. For any structure $\mathbb{K} \in \mathsf{struc}^{\mathsf{fin}}_{a,b}(U)$, we can define *universal* deterministic and non-deterministic $\mathbb{K}$-machines which are able to simulate the machines $\mathcal{M} \in \mathsf{M}_\mathbb{K}$ and $\mathcal{M} \in \mathsf{M}^{\mathrm{N}}_\mathbb{K}$, respectively, on any input $\vec{x}$ if they get $\vec{x}$ and a suitable code of $\mathcal{M}$ as input. In order to encode the programs of these machines by strings which can be transformed into tuples in U^∞, we consider strings over any alphabet U where U can also be infinite. The concatenation of any strings $s_1, s_2 \in U^*$ is denoted by $s_1 s_2$, and for $r \in U^*$ and $\mathcal{S}, \mathcal{S}_1, \mathcal{S}_2 \subseteq U^*$, we have $\mathcal{S}_1\mathcal{S}_2 = \{s_1 s_2 \mid s_1 \in \mathcal{S}_1 \;\&\; s_2 \in \mathcal{S}_2\}$, $r\mathcal{S} = \{r\}\mathcal{S}$, and $\mathcal{S}r = \mathcal{S}\{r\}$.

Definition 2.1 *Let* $\mathcal{S}_{\mathrm{code}} =_{\mathrm{df}} b^2(\{a, b\}^* \backslash (\{a, b\}^* b^2 \{a, b\}^*))$ *be a set of strings which are suitable to be codes and which contain the sub-string* b^2 *as prefix only. Let* $Code^*_\mathbb{K}$ *be an injective mapping of the set of all deterministic and non-deterministic oracle* $\mathbb{K}$*-machines into* $\mathcal{S}_{\mathrm{code}}$ *such that every character of the program is unambiguously translated into a string by this mapping where the oracle queries are encoded independent of the used oracle by taking the same characters as codes for all oracle queries.*

Note that we omit the index $\mathbb{K}$ since confusion is not to be expected. Since, in general, the strings over U are not elements of U, we use tuples as codes. Any $(c_1, \ldots, c_k) \in U^\infty$ can be stored in k registers.

Definition 2.2 *For every non-empty string* $s = c_1 \cdots c_k \in U^*$ *where* $|s| = k \geq 1$, *let* $\lceil s \rceil$ *be the* representation of s in the form of a tuple $(c_1, \ldots, c_k) \in U^k \subset U^\infty$, *that means that* $\lceil c_1 \cdots c_k \rceil = (c_1, \ldots, c_k)$.

To simplify matters, we use the vector notation for the tuples and for the parts

of tuples. $(\vec{x}, \lceil c_1 \cdots c_k \rceil)$ stands for $(x_1, \ldots, x_n, c_1, \ldots, c_k)$, and the like. Moreover, for any $t \geq 1$, $\tilde{t}$ stands for $\lceil b^t a \rceil$ and $Code$ is defined by $Code(\mathcal{M}) = \lceil Code^*(\mathcal{M}) \rceil$ for any machine $\mathcal{M}$.

Definition 2.3 *Let the* Universal $\mathrm{NP}_{\mathbb{K}}$-Problem*, the* Halting Problem*, and a* special halting problem with respect to $\mathbb{K} \in \mathsf{struc}^{\mathsf{fin}}_{a,b}(U)$ *be given by*

$$\mathrm{UNI}_{\mathbb{K}} = \{(\tilde{t}, \vec{x}, Code(\mathcal{M})) |\ \vec{x} \in U^\infty \ \&\ \mathcal{M} \in \mathsf{M}^{\mathrm{N}}_{\mathbb{K}} \ \&\ \mathcal{M} \text{ accepts } \vec{x} \text{ within } t \text{ steps}\},$$

$$\mathrm{H}_{\mathbb{K}} = \{(\vec{x}, Code(\mathcal{M})) \mid \vec{x} \in U^\infty \ \&\ \mathcal{M} \in \mathsf{M}_{\mathbb{K}} \ \&\ \mathcal{M} \text{ halts on } \vec{x}\},$$

$$\mathrm{H}^{\mathrm{spec}}_{\mathbb{K}} = \{Code(\mathcal{M}) \mid \mathcal{M} \in \mathsf{M}_{\mathbb{K}} \ \&\ \mathcal{M} \text{ halts on } Code(\mathcal{M})\}.$$

The first problem can be recognized by a universal non-deterministic machine in polynomial time. We can generalize some known results.

Proposition 2.4 *For each structure* $\mathbb{K} \in \mathsf{struc}^{\mathsf{fin}}_{a,b}(U)$*,* $\mathrm{UNI}_{\mathbb{K}}$ *is* $\mathrm{NP}_{\mathbb{K}}$*-complete.*

Corollary 2.5 *For each structure* $\mathbb{K} \in \mathsf{struc}^{\mathsf{fin}}_{a,b}(U)$*, we have*
(1) $\mathrm{P}_{\mathbb{K}} = \mathrm{NP}_{\mathbb{K}}$ *if and only if* $\mathrm{UNI}_{\mathbb{K}} \in \mathrm{P}_{\mathbb{K}}$*,*
(2) $\mathrm{P}_{\mathbb{K}} \neq \mathrm{NP}_{\mathbb{K}}$ *if* $\mathrm{UNI}_{\mathbb{K}} \notin \mathrm{DEC}_{\mathbb{K}}$*.*

Let us mention that the finite signature of the structure is a sufficient but not a necessary assumption for the definition of $\mathrm{NP}_{\mathbb{K}}$-complete problems. For example, for linear $\mathbb{R}_{\mathrm{lin}}$-machines over the reals and for scalar $\mathbb{Z}_{\mathrm{sc}}$-machines over the integers which can only execute the multiplication by constants, we can encode the constant factors by themselves, but there is not a universal machine (see [24, 13]). However, although there is not any $\mathrm{NP}_{\mathbb{Z}_{\mathrm{sc}}}$-complete problem, there are $\mathrm{NP}_{\mathbb{R}_{\mathrm{lin}}}$-complete problems (see [13]).

The undecidability of the Halting Problem is known for Turing machines, for BSS machines, for *While* programs on standard algebras [28], and so on. For these problems, the undecidability results from the enumerability of the codes of machines and the undecidability of halting sets investigated in [5, 4, 28], respectively. For BSS machines and restricted classes of BSS machines, further halting problems were considered, for instance, in [25] and in [12].

Proposition 2.6 *For each* $\mathbb{K} \in \mathsf{struc}_{a,b}(U)$*,* $\mathrm{H}_{\mathbb{K}} \in \mathrm{DEC}_{\mathbb{K}}$ *implies* $\mathrm{H}^{\mathrm{spec}}_{\mathbb{K}} \in \mathrm{DEC}_{\mathbb{K}}$*.*

Proposition 2.7 *For each* $\mathbb{K} \in \mathsf{struc}_{a,b}(U)$*,* $\mathrm{H}^{\mathrm{spec}}_{\mathbb{K}} \notin \mathrm{DEC}_{\mathbb{K}}$*.*

Proof. Assume that there is a $\mathbb{K}$-machine $\mathcal{M}_0$ which decides $\mathrm{H}^{\mathrm{spec}}_{\mathbb{K}}$. Let $\mathcal{M}_1$ be the following machine. $\mathcal{M}_1$ works as $\mathcal{M}_0$ until the output instruction of $\mathcal{M}_0$ is reached, $\mathcal{M}_1$ does not halt if the output of $\mathcal{M}_0$ is a, and $\mathcal{M}_1$ halts if the output of $\mathcal{M}_0$ is b. That means, that $\mathcal{M}_1$ executes instructions like

$$l\colon \quad Z_2 := a;\ \text{if } Z_1 = Z_2 \text{ then goto } l;\ \text{output } Z_1$$

iff $\mathcal{M}_0$ executes an output instruction of the form

$$l\colon \quad \text{output } Z_1.$$

Therefore, $\mathcal{M}_1$ halts on $Code(\mathcal{M}_1)$ iff the output of $\mathcal{M}_0$ on $Code(\mathcal{M}_1)$ is b, and consequently, iff $Code(\mathcal{M}_1)$ is not in $\mathrm{H}_{\mathbb{K}}^{\mathrm{spec}}$, and thus, iff $\mathcal{M}_1$ does not halt on $Code(\mathcal{M}_1)$. This is a contradiction. □

Corollary 2.8 *For each $\mathbb{K} \in \mathsf{struc}_{a,b}(U)$, $\mathrm{H}_{\mathbb{K}}$ is not decidable by a $\mathbb{K}$-machine.*

3 The Equality of Relativized Versions of P and NP

We shall define a universal oracle $\mathcal{O}$ with $\mathrm{P}_{\mathbb{K}}^{\mathcal{O}} = \mathrm{NP}_{\mathbb{K}}^{\mathcal{O}}$ for any structure $\mathbb{K}$ which permits to compute the codes of the programs of machines over $\mathbb{K}$. The first construction is restricted to structures of finite signature with two constants. Then, we can explicitly encode the programs of machines character-by-character similarly as in [10]. We transfer and modify the definitions given in [1] and [10]. The ideas for the definitions go also back to S. A. Cook, R. Karp, A. Meyer, M. Fischer, and H. B. Hunt. (For more details see [1].) The tuples which can occur in the oracles $\mathcal{O}_1 (= \mathcal{O}_1^{(\mathbb{K})})$ and $\mathcal{O}_2 (= \mathcal{O}_2^{(\mathbb{K})})$ (for a given $\mathbb{K} \in \mathsf{struc}(U)$, we omit the index $\mathbb{K}$) have the same form as the elements of a universal problem.

Definition 3.1 *For any $\mathbb{K} \in \mathsf{struc}_{a,b}^{\mathsf{fin}}(U)$, let*

$$\mathrm{UNI}_1^{(\mathbb{K})}(\mathcal{O}) = \{(\tilde{t}, \vec{x}, Code(\mathcal{M})) \mid \vec{x} \in U^\infty \;\&\; \mathcal{M} \in \mathsf{M}_{\mathbb{K}}^{\mathrm{N}}(\mathcal{O}) \;\&\; \mathcal{M}(\vec{x}) \downarrow^t\}$$

be the Universal $\mathrm{NP}_{\mathbb{K}}^{\mathcal{O}}$-Problem *where $\mathcal{M}(\vec{x}) \downarrow^t$ means that $\mathcal{M}$ accepts $\vec{x}$ for some guesses within t steps. Let $\mathcal{O}_1 (= \mathcal{O}_1^{(\mathbb{K})})$ be a* universal oracle *defined by $\mathcal{O}_1 = \bigcup_{i \geq 0} W_i$ where $W_0 = \emptyset$ and*

$$W_i = \{(\tilde{t}, \vec{x}, Code(\mathcal{M})) \in U^i \mid \mathcal{M} \in \mathsf{M}_{\mathbb{K}}^{\mathrm{N}}(\bigcup_{j<i} W_j) \;\&\; \mathcal{M}(\vec{x}) \downarrow^t\}.$$

For any oracle $\mathcal{O}$, $\mathrm{UNI}_1^{(\mathbb{K})}(\mathcal{O})$ is $\mathrm{NP}_{\mathbb{K}}^{\mathcal{O}}$-complete since the codes of machines allow to simulate the single steps of the oracle machines using the oracle $\mathcal{O}$ by only one universal oracle machine in polynomial time. Moreover, for any $i \geq 0$, we have $\mathrm{UNI}_1^{(\mathbb{K})}(\mathcal{O}_1) \cap U^i = \mathrm{UNI}_1^{(\mathbb{K})}(\bigcup_{j<i} W_j) \cap U^i = W_i$ since the length of a tuple in an oracle query, executed within the first t steps, is less than $t + n < i$ for any input $(x_1, \ldots, x_n)$. This implies $\mathrm{UNI}_1^{(\mathbb{K})}(\mathcal{O}_1) = \mathcal{O}_1$. Because of $\mathcal{O}_1 \in \mathrm{P}_{\mathbb{K}}^{\mathcal{O}_1}$ we get the following.

Proposition 3.2 *For any $\mathbb{K} \in \mathsf{struc}_{a,b}^{\mathsf{fin}}(U)$, there is some $\mathcal{O}$ such that $\mathrm{P}_{\mathbb{K}}^{\mathcal{O}} = \mathrm{NP}_{\mathbb{K}}^{\mathcal{O}}$.*

Remark 3.3 A further characterization of the power of the universal oracle $\mathcal{O}_1$ is possible by comparison of the classes $\mathrm{P}_{\mathbb{K}}^{\mathcal{O}_1}$ and $\mathrm{NP}_{\mathbb{K}}^{\mathcal{O}_1}$ with the classes of the polynomial hierarchy $\mathrm{PH}_{\mathbb{K}}$ and the class $\mathrm{PAT}_{\mathbb{K}}$ containing the problems recognized in polynomial alternating time (for the definitions of these classes see [2, 7–9]). For any $\mathbb{K} \in \mathsf{struc}_{a,b}^{\mathsf{fin}}(U)$, we know that $\mathrm{PH}_{\mathbb{K}} \subseteq \mathrm{PAT}_{\mathbb{K}}$ [9] and $\mathrm{PAT}_{\mathbb{K}} \subseteq \mathrm{P}_{\mathbb{K}}^{\mathcal{O}_1}$ [18].

The mentioned NP-completeness of $\mathrm{UNI}_1^{(\mathbb{K})}(\mathcal{O})$ is not a necessary assumption for the construction. Proposition 3.2 can be generalized to any structure $\mathbb{K}$ if every oracle machine can be encoded by a computable tuple $\vec{u} \in \mathcal{V} =_{\mathrm{df}} \{\lceil v \rceil \in U^\infty \mid v \in$

$b^2(U^* \setminus (U^* b^2 U^*))\}$. For structures of enumerable signature, the possible codes are the indices of a list of all programs as in the definition in [1], or they can have a form like the codes of the linear or scalar real machines, where the operations are encoded by real numbers, and so like. In this way we get the wished oracles also for many structures of infinite signature. Let, for any oracle $\mathcal{O}$,

$$\mathrm{UNI}_2^{(\mathbb{K})}(\mathcal{O}) = \{(\tilde{t}, \vec{x}, \vec{u}) \in U^\infty \mid \vec{u} \in \mathcal{V} \;\&\; (\exists \mathcal{M} \in \mathsf{M}_{\mathbb{K}}^{\mathrm{N}}(\mathcal{O}))(\vec{u} \text{ is the code of } \mathcal{M} \;\&\; \mathcal{M}(\vec{x}) \downarrow^t)\}$$

be a universal problem restricted to non-deterministic $\mathbb{K}$-machines which can use $\mathcal{O}$. $\mathrm{UNI}_2^{(\mathbb{K})}(\mathcal{O})$ is $\mathrm{NP}_{\mathbb{K}}^{\mathcal{O}}$-hard if every code of a machine $\mathcal{M}$ can be computed by a deterministic $\mathbb{K}$-machine $\mathcal{N}_{\mathcal{M}}$ on any input $\vec{x}$. For $\mathcal{O}_2 (= \mathcal{O}_2^{(\mathbb{K})}) = \bigcup_{i \geq 0} W_i$ defined by $W_0 = \emptyset$ and

$$W_i = \{(\tilde{t}, \vec{x}, \vec{u}) \in U^i \mid \vec{u} \in \mathcal{V} \;\&\; (\exists \mathcal{M} \in \mathsf{M}_{\mathbb{K}}^{\mathrm{N}}(\textstyle\bigcup_{j<i} W_j))(\vec{u} \text{ is the code of } \mathcal{M} \;\&\; \mathcal{M}(\vec{x}) \downarrow^t)\},$$

there holds $\mathrm{UNI}_2^{(\mathbb{K})}(\mathcal{O}_2) \cap U^i \subseteq W_i$ for any $i > 0$ if the codes (including the oracle queries) are independent of the used oracle. This implies the following.

Proposition 3.4 *For any $\mathbb{K} \in \mathsf{struc}_{a,b}(U)$, for which the oracle machines can be encoded by computable tuples in U^∞ independently of the used oracle, there is some oracle $\mathcal{O}$ such that $\mathrm{P}_{\mathbb{K}}^{\mathcal{O}} = \mathrm{NP}_{\mathbb{K}}^{\mathcal{O}}$.*

4 The Inequality of Relativized Versions of P and NP

We shall present three kinds of oracles $\mathcal{Q}_1 (= \mathcal{Q}_1^{(\mathbb{K})})$, $\mathcal{Q}_2 (= \mathcal{Q}_2^{(\mathbb{K})})$, and $\mathcal{Q}_3 (= \mathcal{Q}_3^{(\mathbb{K})})$ for several structures $\mathbb{K}$, in order to get the inequality between the corresponding relativized classes. The first two oracles are defined recursively by means of diagonalization techniques. These techniques were also used by Gill, Baker, Solovay, and R. Ladner (for details see [1]) and Emerson [10]. We simplify and generalize the construction for Archimedean rings given by Emerson and for special groups in [19].

4.1 The Classical Way to Define the First Kind of Oracles

If $\mathbb{K}$ is in the class $\mathsf{struc}_{a,b}^{\mathsf{enum}}(U)$ of structures of enumerable signature, then the wished oracle can be defined recursively on the numbers of programs as in [1]. We take positive integers in order to

- enumerate all programs of oracle machines whose form (including the oracle queries) is independent of the used oracle,
- encode all polynomials which can be used to define time bounds for the computation processes,
- encode all couples of polynomials and programs.

Let $i \in \mathbb{N}^+$ be the code of a pair (p_i, P_i) which determines a class of deterministic oracle $\mathbb{K}$-machines $\{\mathcal{N}_i^{\mathcal{B}} \mid \mathcal{B} \subseteq U^\infty\}$ by the following.

(a) The machine $\mathcal{N}_i^{\mathcal{B}}$ performs the instructions of the program P_i.

(b) If $\mathcal{N}_i^{\mathcal{B}}$ queries an oracle, then $\mathcal{N}_i^{\mathcal{B}}$ uses the oracle $\mathcal{B}$.

(c) The number of the instructions of P_i carried out by $\mathcal{N}_i^{\mathcal{B}}$ is simultaneously counted by $\mathcal{N}_i^{\mathcal{B}}$ by means of an additional index register.

(d) For any input in U^n, the machine $\mathcal{N}_i^{\mathcal{B}}$ halts after at most $p_i(n)$ steps of the execution of P_i. (The bound $p_i(n)$ can be computed by using index registers.)

(e) If the output of P_i is reached in this time, then $\mathcal{N}_i^{\mathcal{B}}$ outputs the value determined by P_i. If the output instruction of P_i is not reached in this time, then $\mathcal{N}_i^{\mathcal{B}}$ rejects the input.

Then, for any oracle $\mathcal{B}$ and any problem in $\mathrm{P}_{\mathbb{K}}^{\mathcal{B}}$ there is an $i \geq 1$ such that the machine $\mathcal{N}_i^{\mathcal{B}}$ decides this problem.

The Construction of $\mathcal{Q}_1$.

Let $V_0 = \emptyset$ and $m_0 = 0$. We construct the set $\mathcal{Q}_1$ in stages.

Stage $i \geq 1$: Let n_i be any integer such that $n_i > m_{i-1}$ and $p_i(n_i) + n_i < 2^{n_i}$. Moreover, let

$W_i = \bigcup_{j<i} V_j$,

$V_i = \{\vec{x} \in \{a,b\}^{n_i} \mid \mathcal{N}_i^{W_i}$ *rejects* $(a,\ldots,a) \in U^{n_i}$
$\quad \&\ \vec{x}$ *is not queried by* $\mathcal{N}_i^{W_i}$ *on input* $(a,\ldots,a) \in U^{n_i}\}$,

$m_i = 2^{n_i}$.

Finally, let $\mathcal{Q}_1 = \bigcup_{i\geq 1} W_i$ and $L_1 = \{\vec{y} \mid (\exists i \in \mathbb{N}^+)(\vec{y} \in U^{n_i}\ \&\ V_i \neq \emptyset)\}$.

Lemma 4.1 $L_1 \in \mathrm{NP}_{\mathbb{K}}^{\mathcal{Q}_1} \setminus \mathrm{P}_{\mathbb{K}}^{\mathcal{Q}_1}$.

Proposition 4.2 *For any structure $\mathbb{K} \in \mathsf{struc}_{a,b}^{\mathsf{enum}}(U)$ there is an oracle $\mathcal{Q}$ such that $\mathrm{P}_{\mathbb{K}}^{\mathcal{Q}} \neq \mathrm{NP}_{\mathbb{K}}^{\mathcal{Q}}$.*

4.2 The Second Kind of Oracles

Now, we want to consider mainly structures $\mathbb{K}$ whose signature and, consequently, the programs of oracle machines over $\mathbb{K}$ are not countable. Let us assume that, for any oracle $\mathcal{B}$, all machines in $\mathsf{M}_{\mathbb{K}}(\mathcal{B})$ can be encoded by tuples in a set $\mathcal{U} \subseteq U^\infty$ independently of the used oracle such that each $\vec{u} \in \mathcal{U}$ represents a pair $(p_{\vec{u}}, P_{\vec{u}})$ which determines a class of deterministic oracle $\mathbb{K}$-machines $\{\mathcal{N}_{\vec{u}}^{\mathcal{B}} \mid \mathcal{B} \subseteq U^\infty\}$ satisfying the properties analogously to (a), (b), (c), (d), and (e). Again this implies that, for any problem in $\mathrm{P}_{\mathbb{K}}^{\mathcal{B}}$, there is some $\vec{u} \in \mathcal{U}$ such that $\mathcal{N}_{\vec{u}}^{\mathcal{B}}$ decides this problem in polynomial time.

In order to get $\mathrm{P}_{\mathbb{K}_{\mathbb{R}}}^{\mathcal{Q}} \neq \mathrm{NP}_{\mathbb{K}_{\mathbb{R}}}^{\mathcal{Q}}$ for the structure $\mathbb{K}_{\mathbb{R}} = (\mathbb{R}; \mathbb{R}; +, -, \cdot; \leq, =)$ (where any real number can be a machine constant) Emerson constructed a new kind of oracles. For any program $P_{\vec{u}}$ and any polynomial $p_{\vec{u}}$, he considered the greatest absolute value of all numbers used in a query by one of the oracle machines in $\{\mathcal{N}_{\vec{u}}^{\mathcal{B}} \mid \mathcal{B} \subseteq U^\infty\}$ if these machines get their own code $\vec{u}$ as input. In order to define some oracle recursively, for any natural number $n > 0$, he summarized all codes of

$(p_{\vec{u}}, P_{\vec{u}})$ for which this greatest value is in the interval $]n-1, n]$. Emerson restricted his proofs to an Archimedean ring and he mentioned the possibility to transfer his results to other ordered rings if the Axiom of Choice (AC) and, consequently, the Well-Ordering Axiom are assumed. We can extend his investigation in two directions.

(i) We permit any structure $\mathbb{K}$ with an infinite universe U which allows to define the necessary codes by tuples in U^∞.

(ii) Since U is infinite, we shall assume that there is an element α_0 and an injective mapping $\sigma : U \to U$ satisfying $\sigma(\alpha_i) = \alpha_{i+1}$ and $\alpha_{i+1} \neq \alpha_0$ for all $i \in \mathbb{N}$. The mapping does not need to belong to the structure and it is not necessary that this mapping can be defined or computed over $\mathbb{K}$. We denote the infinite sequence of images by $\bar{1}, \bar{2}, \ldots$ where $\bar{n}(= \bar{n}_{\mathbb{K}}) =_{\mathrm{df}} \sigma(\alpha_{n-1})$ for any $n \in \mathbb{N}^+$.

Remark 4.3 In this way we also answer the three questions posed by Emerson in the last section of [10]. Our assumption is not equivalent to AC. If σ is computable, then neither any restrictions for the operations and the relations of the structure nor for the domain $U \setminus \{\alpha_0, \alpha_1, \ldots\}$ are necessary. The cardinality of the infinite universe U is not important for the construction.

For some other structures, the weaker Axiom of Depend Choice (DC) which was introduced by P. Bernays in his paper [3] and which is used instead of the general AC in the Analytical Topology can be sufficient. Let us consider an infinite abelian group which does not contain an element of infinite order. Then we can consider the inclusion relation on the set of all non-trivial subgroups. By DC there exists, for instance, an infinite sequence of subgroups $(G_i)_{i\geq 0}$ whose members include their predecessors properly. Moreover, this implies the existence of an injective mapping σ by DC where $\sigma(\alpha_i) \in G_{i+1} \setminus G_i$.

The Construction of $\mathcal{Q}_2$.

Let us assume that U contains an infinite sequence $\bar{1}, \bar{2}, \ldots$ given by an injective mapping σ described above. Let $V_0 = \emptyset$. We construct the set $\mathcal{Q}_2$ in stages.

Stage $i \geq 1$: Let

$K_i = \{\vec{u} \in \mathcal{U} \mid (\forall j > i)(\forall \mathcal{B} \subseteq U^\infty)$
$(\mathcal{N}^{\mathcal{B}}_{\vec{u}}$ *does not compute or use the value* $\bar{j}$ *on input* $\vec{u})\}$,

$W_i = \bigcup_{k<i} V_k$,

$V_i = \{(\overline{i+1}, \vec{u}) \mid \vec{u} \in K_i \ \&\ \mathcal{N}^{W_i}_{\vec{u}}$ *rejects* $\vec{u}\}$.

Finally, let $\mathcal{Q}_2 = \bigcup_{i\geq 1} W_i$ *and* $L_2 = \{\vec{y} \mid (\exists n \in \mathbb{N}^+)((\bar{n}, \vec{y}) \in \mathcal{Q}_2)\}$.

Lemma 4.4 $L_2 \in \mathrm{NP}^{\mathcal{Q}_2}_{\mathbb{K}} \setminus \mathrm{P}^{\mathcal{Q}_2}_{\mathbb{K}}$.

Proposition 4.5 *For any structure $\mathbb{K} \in \mathsf{struc}_{a,b}(U)$ with an infinite universe U which allows to encode the $\mathbb{K}$-machines by means of tuples in U^∞ independently of the used oracle, there is an oracle $\mathcal{Q}$ such that $\mathrm{P}^{\mathcal{Q}}_{\mathbb{K}} \neq \mathrm{NP}^{\mathcal{Q}}_{\mathbb{K}}$.*

Remark 4.6 Simpler constructions are possible if there is an element which is not computable from the codes in $\mathcal{U}$. For instance, the deterministic oracle machines

over $\mathbb{K}_{\mathbb{Q},\sqrt{2}} = (\mathbb{Q}\sqrt{2}+\mathbb{Q}; 0, 1; +, -, \cdot; =)$ can be encoded by integers $i \in \mathbb{N}$. Then, the inequalities $\mathrm{DEC}^{\mathcal{Q}}_{\mathbb{K}_{\mathbb{Q},\sqrt{2}}} \neq \mathrm{NP}^{\mathcal{Q}}_{\mathbb{K}_{\mathbb{Q},\sqrt{2}}}$ and thus $\mathrm{P}^{\mathcal{Q}}_{\mathbb{K}_{\mathbb{Q},\sqrt{2}}} \neq \mathrm{NP}^{\mathcal{Q}}_{\mathbb{K}_{\mathbb{Q},\sqrt{2}}}$ hold if $\mathcal{Q} = \{(\sqrt{2}, i) \mid \mathcal{N}_i^{\emptyset}$ rejects $i\}$.

Remark 4.7 The construction given by Emerson was simplified and generalized especially in order to show Proposition 4.5 for any structure of non-enumerable signature. However, for structures $\mathbb{K}$ like the ordered ring over the reals we can prove $\mathrm{P}^{\mathcal{Q}}_{\mathbb{K}} \neq \mathrm{NP}^{\mathcal{Q}}_{\mathbb{K}}$ for further oracles $\mathcal{Q}$. We will show that $\mathbb{Q} \in \mathrm{NP}^{\mathbb{Z}}_{\mathbb{K}_{\mathbb{R}}} \setminus \mathrm{P}^{\mathbb{Z}}_{\mathbb{K}_{\mathbb{R}}}$. Note that the proofs are the same for the unordered ring.

(1) *Proof for* $\mathbb{Q} \in \mathrm{NP}^{\mathbb{Z}}_{\mathbb{K}_{\mathbb{R}}}$. $\mathbb{Q}$ can be non-deterministically recognized by a machine in $\mathsf{M}^{\mathrm{N}}_{\mathbb{K}_{\mathbb{R}}}(\mathbb{Z})$ which queries the oracle whether the guesses y_1 and y_2 are integers and which checks $y_1 \neq 0$ and $y_1 x = y_2$ for any input x.

(2) *Proof for* $\mathbb{Q} \notin \mathrm{P}^{\mathbb{Z}}_{\mathbb{K}_{\mathbb{R}}}$. Assume that there is a machine $\mathcal{N}$ in $\mathsf{M}_{\mathbb{K}_{\mathbb{R}}}(\mathbb{Z})$ which decides $\mathbb{Q}$ in polynomial time. The decidability of a set of reals in polynomial time means that there is a number $t_0 \geq 1$ such that any input $x \in \mathbb{R}$ is accepted or rejected within t_0 steps. Consequently, the number of computation paths of $\mathcal{N}$ traverse by the inputs $x \in \mathbb{R}$ is finite. Thus, there is a finite set $M = \{p_1, \ldots, p_m\}$ containing polynomial functions of arity 1 and degree $d \geq 1$, such that each of these paths, P, can be described by a system S_P consisting of conditions of the form $p_k(x) \leq 0$, $p_k(x) > 0$, $p_k(x) \in \mathbb{Z}$, and $p_k(x) \notin \mathbb{Z}$ where $k \leq m$. An input x traverses a path P if and only if it satisfies S_P (for more details, compare also [12]). Moreover, $X = \{x \mid (\exists k \leq m)(p_k(x) \in \mathbb{Z})\}$ is countable. Therefore, the set $\mathbb{R} \setminus (\mathbb{Q} \cup X)$ is non-empty and it contains a real number r which is rejected by $\mathcal{N}$. Let P_r be the computation path of $\mathcal{N}$ traversed by r. Because of $r \notin X$, S_{P_r} does not contain conditions of the form $p_k(x) \in \mathbb{Z}$. If a condition of the form $p_k(x) \leq 0$ belongs to S_{P_r}, then $p_k(r) < 0$ holds. For any sequence of rational numbers $(q_i)_{i\in\mathbb{N}}$ with limit r there is an $i_0 \in \mathbb{N}$ such that, for all $i \geq i_0$, S_{P_r} is also satisfied by q_i. This is a contradiction to $q_i \in \mathbb{Q}$ since we suppose that any computation path is either an accepting path or a rejecting path.

4.3 The Third Kind of Oracles

The following oracle is not recursively defined and we can use the undecidability of the corresponding Halting Problem in the proof. We consider only the class $\mathsf{struc}^{\bar{\mathbb{N}}}_{a,b}(U)$ containing all structures $\mathbb{K} \in \mathsf{struc}^{\mathsf{fin}}_{a,b}(U)$ for which U includes an infinite set $\bar{\mathbb{N}} = \{\bar{0}, \bar{1}, \bar{2}, \ldots\}$ with the following properties.

- $\bar{\mathbb{N}}$ is defined by some injective mapping σ of U into U where $\overline{i+1} = \sigma(\bar{i}) \neq \bar{0}$.
- $\bar{\mathbb{N}}$ is decidable by a deterministic $\mathbb{K}$-machine.
- $\bar{\mathbb{N}}$ is enumerable by a deterministic $\mathbb{K}$-machine which can compute $\bar{0}$ independently of the input and which can compute $\overline{i+1}$ from $\bar{i}$.

The constructions given in Sections 4.1 and 4.2 are possible for any classes of time bounds limiting the work of the deterministic oracle machines. We can build, for instance, some oracle $\mathcal{Q}$ such that $\mathrm{EXP}^{\mathcal{Q}}_{\mathbb{K}} \neq \mathrm{NP}^{\mathcal{Q}}_{\mathbb{K}}$ holds if we use the exponential

functions as time bounds. The next oracle implies the corresponding inequalities for each class of time bounds.

The Definition of $\mathcal{Q}_3$.
For $\mathbb{K} \in \mathsf{struc}^{\bar{\mathbb{N}}}_{a,b}(U)$, let

$$\mathcal{Q}_3 = \{(\bar{t}, \vec{x}, Code(\mathcal{M})) \in U^\infty \mid \mathcal{M} \in \mathsf{M}_\mathbb{K} \;\&\; t \in \mathbb{N}^+ \;\&\; \mathcal{M}(\vec{x}) \downarrow^t\}.$$

Lemma 4.8 *For any $\mathbb{K} \in \mathsf{struc}^{\bar{\mathbb{N}}}_{a,b}(U)$, $\mathrm{H}_\mathbb{K} \in \mathrm{NP}^{\mathcal{Q}_3}_\mathbb{K}$ and $\mathrm{P}^{\mathcal{Q}_3}_\mathbb{K} \subseteq \mathrm{DEC}_\mathbb{K}$.*

By Corollary 2.8 we can conclude the following.

Proposition 4.9 *For any $\mathbb{K} \in \mathsf{struc}^{\bar{\mathbb{N}}}_{a,b}(U)$ there is some oracle $\mathcal{Q}$ such that $\mathrm{P}^{\mathcal{Q}}_\mathbb{K} \neq \mathrm{NP}^{\mathcal{Q}}_\mathbb{K}$.*

Remark 4.10 The results can be transferred to structures of infinite signature if they contain only finitely many relations and operations, for instance, to the structure $\mathbb{K}_\mathbb{R} = (\mathbb{R}; \mathbb{R}; +, -, \cdot; \leq, =)$.

5 Relations Instead of Oracles?

Since we do not know the answer for the classical problem $\mathrm{P} \stackrel{?}{=} \mathrm{NP}$, we should study the properties of all known structures $\mathbb{K}$ and the relationships between the classes $\mathrm{P}_\mathbb{K}$ and $\mathrm{NP}_\mathbb{K}$ (like, for instance, in [23, 24, 21, 26, 6, 11]) and we should investigate several possibilities to construct structures $\mathbb{K}$ with $\mathrm{P}_\mathbb{K} = \mathrm{NP}_\mathbb{K}$ (compare [26, 22, 14–18, 20, 27]). Inspired by a construction of a structure $\mathbb{K}$ of infinite signature with $\mathrm{P}_\mathbb{K} = \mathrm{NP}_\mathbb{K}$ given by G. Mainhardt [22] where an infinite number of relations was derived from a universal $\mathrm{NP}_\mathbb{K}$-problem, we want to discuss the following question. Is it possible to replace the oracle $\mathcal{O}_1^{(\mathbb{K})}$ for some $\mathbb{K}$ by one additional relation of fixed arity in order to get a structure $\mathbb{M}$ of finite signature with $\mathrm{P}_\mathbb{M} = \mathrm{NP}_\mathbb{M}$?

If we want to derive a new relation R (which can be satisfied only by tuples of a fixed length n_R) from the oracle $\mathcal{O}_1$ such that any oracle query $(Z_1, \ldots, Z_{I_1}) \in \mathcal{O}$ can be replaced by a condition of the form $R(Z_1, \ldots, Z_{n_R})$, then we have to compress the tuples in $\mathcal{O}_1$ to tuples of length n_R. Since, for many structures, it is not possible to compute a bijection of the set of the finite sequences of elements into a set of tuples of a fixed length, here we want to consider a class of structures over strings which allow to encode finite sequences of elements by single elements.

Definition 5.1 *For an arbitrary universe U, let $\mathcal{A} = U^*$ such that the elements of U are the characters of the strings in $\mathcal{A}$, and let $\mathsf{struc}^*(\mathcal{A})$ be the class of structures of the form $(\mathcal{A}; \mathcal{A}_0; f_1, \ldots, f_{k_1}, \mathrm{add}, \mathrm{sub_l}, \mathrm{sub_r}; R_1, \ldots, R_{k_2}, =)$ where $\mathcal{A}_0 \subseteq \mathcal{A}$ is a finite set of constants and $a, b, \varepsilon \in \mathcal{A}_0$.* add *is a binary operation for adding a character to a string.* $\mathrm{sub_r}$ *and* $\mathrm{sub_l}$ *are unary operations for computing the last character and the remainder of a string, respectively. That means that these functions are defined for the strings $s \in \mathcal{A}$, $r \in \mathcal{A} \setminus U$, and $c \in U$ by $\mathrm{add}(s, c) = sc$, $\mathrm{sub_l}(sc) = s$, $\mathrm{sub_r}(sc) = c$, $\mathrm{add}(s, r) = \varepsilon$, $\mathrm{sub_l}(\varepsilon) = \varepsilon$, and $\mathrm{sub_r}(\varepsilon) = \varepsilon$. Each f_i is an operation on $\mathcal{A}$. Each R_i is a relation on $\mathcal{A}$.*

Lemma 5.2 *$\{b^i \mid i \in \mathbb{N}\}$ is decidable and enumerable over $\mathbb{K} \in \mathsf{struc}^*(\mathcal{A})$.*

Moreover, in encoding the elements of oracles we can use that the tuples of strings can be encoded by strings in the following way.

Definition 5.3 *For every string $s \in \mathcal{A}$, let the value $\langle s\rangle$ be recursively defined by $\langle\varepsilon\rangle = a$ and $\langle rc\rangle = \langle r\rangle ca$ for all strings $r \in \mathcal{A}$ and all character $c \in U$. For every integer $n > 1$ and every tuple $\vec{s} = (s_1, \ldots, s_n) \in \mathcal{A}^n$, let $\langle s_1, \ldots, s_n\rangle$ be the string $\langle s_1\rangle b^2 \cdots \langle s_{n-1}\rangle b^2 \langle s_n\rangle$.*

Although the elements of the oracles $\mathcal{O} = \mathcal{O}_1^{(\mathbb{K})}$ and $\mathcal{Q} = \mathcal{Q}_3^{(\mathbb{K})}$ (for any $\mathbb{K} \in \mathsf{struc}^*(\mathcal{A})$) have a similar form, we have different relationships between the relativized versions of $\mathrm{P}_{\mathbb{K}}$ and $\mathrm{NP}_{\mathbb{K}}$. That implies, on the one hand, the conjecture that it could be easy to define oracles $\bar{\mathcal{O}}, \bar{\mathcal{Q}} \subseteq \mathcal{A}$ or new unary relations R by compressing the sequences of strings in $\mathcal{O}, \mathcal{Q} \subseteq \mathcal{A}^\infty$ to single strings in order to get $\mathrm{P}_{\mathbb{K}}^{\bar{\mathcal{O}}} = \mathrm{NP}_{\mathbb{K}}^{\bar{\mathcal{O}}}$ and $\mathrm{P}_{\mathbb{K}}^{\bar{\mathcal{Q}}} \neq \mathrm{NP}_{\mathbb{K}}^{\bar{\mathcal{Q}}}$ and $\mathrm{P}_{\mathbb{K}_R} = \mathrm{NP}_{\mathbb{K}_R}$ or $\mathrm{P}_{\mathbb{K}_R} \neq \mathrm{NP}_{\mathbb{K}_R}$ for new structures $\mathbb{K}_R$. On the other hand it implies the conjecture that it is not possible to define oracles $\bar{\mathcal{O}} \subseteq \mathcal{A}$ with $\mathrm{P}_{\mathbb{K}}^{\bar{\mathcal{O}}} = \mathrm{NP}_{\mathbb{K}}^{\bar{\mathcal{O}}}$ since the different relationships between the complexity classes, relativized by using the oracles $\mathcal{O}$ and $\mathcal{Q}$, respectively, mainly are the result of the different representation of the number of steps: In case of $\mathcal{O}$, the number of possible steps, t, is determined by the length of the tuple $\tilde{t}$. In case of $\mathcal{Q}$, the number of steps is given by only one element of the structure. To use only single strings as codes of the elements of $\mathcal{O}$ in defining a new oracle $\bar{\mathcal{O}}$ could be easier said than done. The following results bear out that. They follow from the undecidability of $\mathrm{H}_{\mathbb{K}}^{\mathrm{spec}}$ and

$$\mathrm{H}_{\mathbb{K}}^{\mathrm{spec}}(\bar{\mathcal{O}}) = \{Code(\mathcal{M}) \in \mathcal{A}^\infty \mid \mathcal{M} \in \mathsf{M}_{\mathbb{K}}(\bar{\mathcal{O}}) \;\&\; \mathcal{M} \text{ halts on } Code(\mathcal{M})\}.$$

Theorem 5.4 *For any $\mathbb{K} \in \mathsf{struc}^*(\mathcal{A})$, the oracle*

$$\bar{\mathcal{Q}} = \{b^t\langle\vec{x}\rangle Code^*(\mathcal{M}) \in \mathcal{A} \mid \mathcal{M} \in \mathsf{M}_{\mathbb{K}} \;\&\; t \in \mathbb{N}^+ \;\&\; \mathcal{M}(\vec{x}) \downarrow^t\}$$

implies $\mathrm{P}_{\mathbb{K}}^{\bar{\mathcal{Q}}} \neq \mathrm{NP}_{\mathbb{K}}^{\bar{\mathcal{Q}}}$.

Whereas it is easy to transfer the construction of oracles in order to again obtain inequalities between the relativized polynomial time complexity classes for structures over strings, the method does not work if we want to again get equations for the relativized classes as it is shown by the following theorem.

Theorem 5.5 *For any $\mathbb{K} \in \mathsf{struc}^*(\mathcal{A})$, there is not any oracle satisfying*

$$b^t\langle\vec{x}\rangle Code^*(\mathcal{M}) \in \bar{\mathcal{O}} \Leftrightarrow \mathcal{M} \in \mathsf{M}_{\mathbb{K}}^{\mathrm{N}}(\bar{\mathcal{O}}) \;\&\; t \in \mathbb{N}^+ \;\&\; \mathcal{M}(\vec{x}) \downarrow^t .$$

Each deterministic machine over $\mathbb{K}_{\{a,b\}^*} = (\{a,b\}^*; a, b, \varepsilon; \mathrm{add}, \mathrm{sub_l}, \mathrm{sub_r}; =)$ can be simulated by some Turing machine. Thus, the following statement follows from the undecidability of the Halting Problem for the set TM of Turing machines.

Proposition 5.6 *The set*

$$\bar{\mathcal{Q}}_{\mathrm{TM}} = \{b^t\langle\vec{x}\rangle Code^*(\mathcal{M}) \in \{a,b\}^* \mid \mathcal{M} \in \mathsf{TM} \;\&\; t \in \mathbb{N}^+ \;\&\; \mathcal{M}(\vec{x}) \downarrow^t\}$$

implies $\mathrm{DEC}_{\mathbb{K}_{\{a,b\}^*}}^{\bar{\mathcal{Q}}_{\mathrm{TM}}} \neq \mathrm{NP}_{\mathbb{K}_{\{a,b\}^*}}^{\bar{\mathcal{Q}}_{\mathrm{TM}}}$ and, hence, $\mathrm{P}_{\mathbb{K}_{\{a,b\}^*}}^{\bar{\mathcal{Q}}_{\mathrm{TM}}} \neq \mathrm{NP}_{\mathbb{K}_{\{a,b\}^*}}^{\bar{\mathcal{Q}}_{\mathrm{TM}}}$.

Remark 5.7 The last result remains true if we consider machines over the structure $(\{a,b\}^*; a, b, \varepsilon; \mathrm{add}, \mathrm{sub_l}, \mathrm{sub_r}; =a, =b)$ where any test has the form $Z_j = a$ or $Z_j = b$.

Remark 5.8 A possibility to define new relations R of arity 1 (or oracles containing only single elements of the universe) derived from $\mathcal{O}_1^{(\mathbb{K})}$ such that there holds $\mathrm{P}_{\mathbb{K}_R} = \mathrm{NP}_{\mathbb{K}_R}$ for the new structures $\mathbb{K}_R$, is presented in [14, 15, 18]. The crucial idea is to define new relations R satisfied by padded codes of the elements of an $\mathrm{NP}_{\mathbb{K}_R}$-complete problems. (For more details see [16, 17], too.) The subject of [14] is the construction of a new structure of binary trees for which the equality of trees cannot be decided in one step.

In this way we can once more substantiate the thesis that additional oracles are not very helpful for solving the $\mathrm{P}_{\mathbb{K}} \stackrel{?}{=} \mathrm{NP}_{\mathbb{K}}$ problem for any structure $\mathbb{K}$. On the one hand, we know structures $\mathbb{K}$ with $\mathrm{P}_{\mathbb{K}} \neq \mathrm{NP}_{\mathbb{K}}$ and we can define an oracle $\mathcal{O}$ which implies $\mathrm{P}_{\mathbb{K}}^{\mathcal{O}} = \mathrm{NP}_{\mathbb{K}}^{\mathcal{O}}$. On the other hand, we know structures $\mathbb{M}$ with $\mathrm{P}_{\mathbb{M}} = \mathrm{NP}_{\mathbb{M}}$ and we can define an oracle $\mathcal{Q}$ implying $\mathrm{P}_{\mathbb{M}}^{\mathcal{Q}} \neq \mathrm{NP}_{\mathbb{M}}^{\mathcal{Q}}$.

References

[1] Baker, T., J. Gill, and R. Solovay (1975). Relativizations of the P $\stackrel{?}{=}$ NP question. *SIAM J. Comput.* 4, 431–442.

[2] J. Balcázar, J. Díaz, and J. Gabarró (1988/90). *Structural Complexity I* and *Structural Complexity II*. Springer-Verlag.

[3] Bernays, P. (1942). A system of axiomatic set theory. *Journal of Symbolic Logic* 7, 65–89.

[4] Blum, L., F. Cucker, M. Shub, and S. Smale (1998). *Complexity and Real Computation*. Springer-Verlag.

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

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

[7] Cucker, F. (1993) On the complexity of quantifier elimination: the structural approach. *The Computer Journal* 36, 399–408.

[8] Bournez, O., F. Cucker, P. J. de Naurois, and J.-Y. Marion (2003). Safe recursion over an arbitrary structure. *ENTCS* 90, 3–14.

[9] Bournez, O., F. Cucker, P. J. de Naurois, and J.-Y. Marion (2006). Implicit complexity over an arbitrary structure: Quantifier alternations. *Information and Computation* 202, 2, 210–230.

[10] Emerson, T. (1994). Relativizations of the P $\stackrel{?}{=}$ NP question over the reals (and other ordered rings). *Theoretical Computer Science* 133, 15–22.

[11] Gaßner, C. (2001). The P-DNP problem for infinite abelian groups. *Journal of Complexity* 17, 574–583.

[12] Gaßner, C. (2007). A Hierarchy below the Halting Problem for Additive Machines. *Theory of Computer Systems* DOI 10.1007/s00224-007-9020-y.

[13] Gaßner, C. (1997). On NP-Completeness for Linear Machines. *Journal of Complexity* 13, 259–271.

[14] Gaßner, C. (2004). Über die Konstruktion von Strukturen endlicher Signatur mit P = NP. *Preprint* 1/2004. [3]

[3] *Preprint-Reihe Mathematik, E.-M.-Arndt-Universität Greifswald*

[15] Gaßner, C. (2004). NP $\subset$ DEC und P = NP für Expansionen von Erweiterungen von Strukturen endlicher Signatur mit Identitätsrelation. *Preprint* 13/2004.[3]

[16] Gaßner, C. (2006). A structure with P = NP. *CiE 2006. Computer Science Report Series of the University of Wales Swansea* CSR 7, 85–94.

[17] Gaßner, C. (2006). Expansions of structures with P = NP. *CiE 2006. Computer Science Report Series of the University of Wales Swansea* CSR 7, 95–104.

[18] Gaßner, C. (2007). P = NP for Expansions Derived from Some Oracles. *CiE 2007. Technical report* no. 487, June 2007, 161–169.

[19] Gaßner, C. (2008). Computation over a group. *CiE 2008. Technical report*, University of Athens, June 2007, 147–156.

[20] Hemmerling, A. (2005). P = NP for some structures over the binary words. *Journal of Complexity* 21, 557–578.

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

[22] Mainhardt, G. (2004). P versus NP and computability theoretic constructions in complexity theory over algebraic structures. *Journal of Symbolic Logic* 69, 39–64.

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

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

[25] Meer, K. and M. Ziegler (2005). An explicit solution to Post's problem over the reals. *15th International Symposium on Fundamentals of Computation Theory* LNCS 3623, 456–467.

[26] Poizat, B. (1995). *Les Petits Cailloux.* Aléas.

[27] Prunescu, M. (2006). Structure with fast elimination of quantifiers. *Journal of Symbolic Logic* 71, 321–328.

[28] Tucker, J. V. and J. I. Zucker (2000). Computable functions and semicomputable sets on many-sorted algebras. *Handbook of Logic in Computer Science* 5 (S. Abramskz, D. Gabbay, T. Maibaum Eds.), Oxford University Press, 317–523.