# Two situations with unit-cost: ordered abelian semi-groups and some commutative rings

Mihai Prunescu *

**Abstract**

The paper presents two situations where unit-cost complexity results are closely related with results from the classical computability.
- In the second section we study an important theorem by Pascal Koiran and Hervé Fournier from an axiomatic point of view. It is proved that the algebraic Knapsack problem belongs to P over some ordered abelian semi-group iff P = NP classically. In this case there would exist a unit-cost machine solving the algebraic Knapsack problem over all ordered abelian semi-groups in some uniform polynomial time.
- In the third section we apply the theorem of Matiyasevich in order to construct a ring with P $\neq$ NBP $\neq$ NP and such that its polynomial hierarchy does not collapse at any level.
**A.M.S.-Classification**: 03C60, 03D15.

## 1 Introduction

This paper presents two situations where unit-cost complexity results are closely related with results from the classical computability. In the second section we study an important theorem by Pascal Koiran and Hervé Fournier from an axiomatic point of view.

**Definition**: The classical problem Knapsack: given a finite list of natural numbers, it is asked if the last number in the list is a sum of some other numbers in the list. There are many other versions and specifications and many ways to encode the Knapsack problem. The present version is sometimes also called Subset Sub and denoted SSS. All variants of Knapsack are known to be NP-complete in the classical sense, see [7], [11], [15], [22].

**Definition**: The algebraic Knapsack problem (Knapsack with unit cost): given a finite list of real numbers (or, more generally, a list of elements in an algebraic structure with addition) it is asked if the last element in the list is a sum of some other elements in the list.

The following example should stress the differences between Knapsack as a classical (Turing) problem and Knapsack as an algebraic (unit-cost) problem: An instance of the classical problem Knapsack is for example a word of length $n$ with letters in the alphabet {0, 1, 2, 3, 4, 5, 6, 7, 8, 9, ♮} like for example:

$$15674\natural 08\natural 900\natural 666\natural 0002\natural 15\natural 010.$$

This instance has length 28 and is a solution of the problem. If we look at this instance as an instance of the algebraic unit-cost Knapsack problem, (over some structure with addition, like $(\mathbb{R}, +, <)$ or like $(\mathbb{N}, +, \cdot)$) then its length is only 7. The problem to algorithmically decide in a polynomial time in the digit-cost if an instance is a solution seems at the very first sight to be very different from the same problem to be solved in polynomial time according to the unit-cost. Also, the fact that in the first case just a recursive function in words must be computed and that in the

*Universität Freiburg, Germany; and I.M.A.R. Bucharest, Romania. This research was supported by the DFG over a post-doctoral grant in the *Graduiertenkolleg Mathematische Logik und Anwendungen*.

second case one must use only functions, relations and constants given in the structure contributes to this belief.

Nevertheless Hervé Fournier and Pascal Koiran proved in [4] the following surprising result:

**Theorem 1.1** *The ordered additive group of the reals* $(\mathbb{R}, +, -, <, 0, 1)$ *has* $\mathrm{P} = \mathrm{NP}$ *in the sense of unit-cost (algebraic) complexity for parameter-free computations if and only if* $\mathrm{P} = \mathrm{NP}$ *in the classical sense.*

For proving their result, Fournier and Koiran refined older geometric constructions by Friedhelm Meyer auf der Heide, see [9]. The most important notion in this inductive proof is maybe auf der Heide's invariant called *coarseness* of a hyperplane arrangement. The coarseness is the maximal radius of a hyperball with the following property: if the ball meets at least two hyperplanes in this arrangement, then they are not parallel and the ball intersects also their common intersection. In the proof of the theorem one can consider a problem in the sense of unit-cost (algebraic) complexity over the ordered reals with addition as a family of arrangements of polynomials in different dimensions. Without restricting the generality we can suppose the input to be in an unitary hypercube. This hypercube is partitioned in small hypercubes having a radius smaller than the coarseness of the hyperplane arrangement representing the problem in the given dimension. The coarseness is a rational number which can be computed with a method given by Meyer auf der Heide: to represent it binary takes a length bounded by a polynomial in the parameter $n$ (which is in the same time unit-cost of the input). Localising the input in some small box is done by binary search. The geometrical situation in the small box is projected on a $n-1$ dimensional side of the original hypercube, and the same procedure is repeated in dimension $n-1$. In order to guess a right face of the hypercube (a face, to project on) and to find points with rational coordinates on given linear varieties, only classical NP-oracles are used. This is why $\mathrm{P} = \mathrm{NP}$ classically implies $\mathrm{P} = \mathrm{NP}$ in the sense of unit-cost complexity.

For the other direction Fournier and Koiran looked at the boolean parts of the NP-problems according to the unit-cost and found out the classical NP-problems. This can lead to the opinion that it was essential to have constants 0 and 1 in the language. I give here a constant-free proof for this implication.

In the second section the theorem of Fournier and Koiran will be generalized in the following sense: over a wide class of structures only with addition and order the unit-cost version of Knapsack is equivalent with the classical P versus NP problem.

In the third section we work with unit-cost only. According to [1] and [17], given a ring $(R, +, -, \cdot, 0, 1)$, an input over this ring is a finite string of ring-elements. A problem over the ring is a set of inputs.

The class of problems $\mathrm{P}(R)$ decidable in polynomial time according to the unit-cost over the ring $R$ is defined as follows: there is a deterministic machine able to read inputs from $R$ of arbitrary length, to carry out operations $+, -, \cdot$, and to compare already computed elements according to equality tests, which stops in a number of steps bounded by some polynomial in the length of the input $n$ and gives the correct answer to the question if the input belongs to the given problem. Carrying out an operation or equality test can be always done in one unit of time (one step). Machines must have a finite description and may contain finitely many elements of the ring as constants in their description.

A problem $B$ over $R$ belongs to the class $\mathrm{NBP}(R)$ (first defined in [8], see also [17], and called boolean or digital NP) if and only if there is a polynomial $p(n)$ and a problem $A \in \mathrm{P}(R)$ such that for all inputs $\vec{x}$ in $R$:

$$\vec{x} \in B \Leftrightarrow \exists\, \vec{y} \in \{0,1\}^{p(|\,\vec{x}\,|)}\ (\vec{x}, \vec{y}) \in A.$$

Here $|\,\vec{x}\,|$ denotes the length of the vector $\vec{x}$, which is also its complexity measure as an input.

Analogously a problem $B$ over $R$ belongs to the class $\mathrm{NP}(R)$ (existential NP) if and only if there

is a polynomial $p(n)$ and a problem $A \in \mathrm{P}(R)$ such that for all inputs $\vec{x}$ in $R$:

$$\vec{x} \in B \Leftrightarrow \exists\, \vec{y} \in R^{p(|\,\vec{x}\,|)}\ (\vec{x}, \vec{y}) \in A.$$

Both definitions can be generalized to corresponding polynomial hierarchies. I write down the definition for the existential polynomial hierarchy PH $(R)$ because this is the hierarchy used in the third section. A problem $B$ over $R$ belongs to the class $\Sigma_k\mathrm{P}(R)$ if and only if there are polynomials $p_1(n)$, $p_2(n)$, ..., $p_k$ and a problem $A \in \mathrm{P}(R)$ such that for all inputs $\vec{x}$ in $R$:

$$\vec{x} \in B \Leftrightarrow \exists\, \vec{y}_1 \in R^{p_1(|\,\vec{x}\,|)}\ \forall\, \vec{y}_2 \in R^{p_2(|\,\vec{x}\,|)} \ldots \ldots Q_k\ \vec{y}_k \in R^{p_k(|\,\vec{x}\,|)}\ (\vec{x}, \vec{y}_1, \vec{y}_2, \ldots, \vec{y}_k) \in A.$$

The last quantifier $Q_k$ is an $\exists$ if $k$ is odd and a $\forall$ if $k$ is even.

The class $\Pi_k\mathrm{P}(R)$ consists of the complements of all problems in $\Sigma_k\mathrm{P}(R)$ in the sets of all inputs. It is easy to check that $\Sigma_k\mathrm{P}(R) \subseteq \Sigma_{k+1}\mathrm{P}(R)$ and $\Sigma_k\mathrm{P}(R) \subseteq \Pi_{k+1}\mathrm{P}(R)$. Also that $\Sigma_0\mathrm{P} = \Pi_0\mathrm{P} = \mathrm{P}$ and that $\Sigma_1\mathrm{P} = \mathrm{NP}$, respectively that $\Pi_1\mathrm{P} = \text{co-NP}$. The set $\bigcup_{k\geq 0} \Sigma_k\mathrm{P}(R)$ is denoted by $\mathrm{PH}(R)$ and is called the polynomial hierarchy of $R$.

It is known that a ring with P = NP (according to the formal language of rings) must be an algebraically closed field, and it is known that the answer to the still open question P = NP over an algebraically closed field of characteritic zero is independent of the choice of the algebraically closed field, see [2]. Also, for rings which are not algebraically closed fields (and so we know that there holds P $\neq$ NP) it is interesting to know if P $\neq$ NBP and if NBP $\neq$ NP. The first inequality, supposed to be true for all infinite rings, has been proved so far only for the infinite boolean rings, see [17] and [19]. In the case of the ring of integers $(\mathbb{Z}, +, -, \cdot, 0, 1)$ the second inequality follows easily from the theorem of Matiyasevich. This fact looks like folklore but has not been found however by the author in the literature, so it shall be discussed in the third section. The inequality P $\neq$ NBP is on the other side very difficult for the integers. Such a proof would imply P $\neq$ NBP for all domains of characteristic zero, in particular for the field of complex numbers $\mathbb{C}$.

In the third section we construct a ring $R$ with P $(R)$ $\neq$ NBP $(R)$ $\neq$ NP $(R)$ and such that its polynomial hierarchy does not collapse at any level. This is the only kind of example known by the author in the present that enjoys all these properties.

Occasionally we will write P, NBP, NP instead of $\mathrm{P}(R)$, $\mathrm{NBP}(R)$, $\mathrm{NP}(R)$ if there is no danger of confusion between classical and unit-cost complexity classes.

## 2 Ordered abelian semi-groups

We fix the formal language $L = (+, <)$ without constants, containing only a binary operation symbol and a binary relation symbol. We call T the theory generated by the following universal axioms.

1. The additive operation + is associative and commutative.

$$(x + y) + z = x + (y + z)\ \wedge\ x + y = y + x.$$

2. The relation $<$ is an irreflexive, transitive and total order.

$$(\neg\, x < x)\ \wedge\ (x < y \wedge y < z \longrightarrow x < z)\ \wedge\ (x < y \vee\ x = y\ \vee y < x).$$

3. The order is strongly compatible with the addition.

$$x < y\ \longleftrightarrow\ x + z < y + z.$$

All the axioms can be carried together in one formal statement starting with a block of three universal quantifiers followed by a disjunctive normal form. All axioms are standard excepting the last one. Note that the last two axioms imply that $x + z = y + z \to x = y$. So T is the theory of ordered abelian semi-groups not necessarily with 0.

The theory T enjoys a large variety of models. To give some examples, we notice $\mathbb{Z}$, the sets $\mathbb{N}_k$ of all naturals $n \geq k$; $\mathbb{Q}$, $\mathbb{R}$; all their non-negative half-lines with or without first element (negative half-lines with or without last element); any direct product of other models with the lexicographic ordering and many other non-standard models. Even $\mathbb{N}_1$ with + interpreted as the multiplication of natural numbers is a model of T.

We will consider machines without any constants working in the language of T. The machine can only copy elements from cell in cell (copy-operation), perform additions and check inequalities. Consequently, the machine can also check equalities: $\neg\ x < y\ \wedge\ \neg\ y < x\ \longleftrightarrow\ x = y$.

If any unit-cost machine decides Knapsack over a model of T, we can write down for all $n \geq 1$ an universal statement $\psi_n$ of the form:

$$\forall\ x_1, \ldots, x_{n+1} \quad \vec{x} \in \text{Knapsack} \ \longleftrightarrow\ \bigvee \bigwedge (\neg) \text{ left sum} < \text{right sum}.$$

Here on the left hand side the formula $\vec{x} \in$ Knapsack is a disjunction of $2^n - 1$ equalities in the $n$ variables. On the right hand side we have the disjunction over all the accepting paths in machine's computation tree. Any disjunct is a conjunction along the path taken over all tests performed by the machine in this path. A computation path is exactly determined by the tests performed along the path itself and by their results. According to the accepting computation path in question only some tests may occur negated. The statement $\psi_n$ says that the machine recognizes Knapsack for inputs of length $n$ and moreover describes its behavior in all occurring positive cases. We similarly define formal statements $\nu_n$, where the conjunctive formula of exponential length $\vec{x} \notin$ Knapsack is for all inputs equivalent with the disjunction taken over all rejecting paths of the conjunctions of (sometimes negated) tests defining the respective rejecting path. The statement $\nu_n$ says that the machine rejects the complement of Knapsack in dimension $n$ and describes its behavior in all occurring negative cases.

**Definition**: The first-order theory of a Knapsack machine is the set $\{\psi_n \wedge \nu_n | \, n \geq 1\}$.

We stress the fact that these conjunctions are not contradictory: statements say "for all inputs, they belong to Knapsack if and only if they are treated along an accepting path and they do not belong to Knapsack if and only if they are treated along some rejecting path". Belongness to Knapsack is itself a first order formula. Such translations in first order formulae have been written down for any unit-cost machines with given time-complexity. It has been remarked by many authors before, see for example [14], [10], or the survey [13].

In the following statement of Theorem 2.1 occurring machines are parameter-free, hence can be used over different structures of the same signature $\{+, <\}$, containing a binary operation and a binary relation. In this context we call *special* machine a machine able to solve the problem Knapsack in polynomial time only over a given structure with this signature. The machine occurring at the end of the theorem is called *general* because it can be started with inputs in arbitrary models of the theory T. Of course, if started for an input in a given ordered abelian semigroup $S$, all computations are carried out in this semigroup $S$.

**Theorem 2.1** *The following statements are equivalent:*

1. *There is some ordered abelian semi-group $(S, +, <) \neq 0$ and a parameter-free special machine only in $(+, <)$ that decides the algebraic Knapsack problem in polynomial time over $S$.*
2. P = NP.
3. *There is a parameter-free general machine in the same language $(+, <)$ solving the algebraic Knapsack problem in uniform polynomial time over all ordered abelian semi-groups.*

According to Gödel's Completeness Theorem for Predicate Calculus if some formal statement (set of formal statements) is true for all the models in some elementary class, then the given statement (set of statements) is provable in the theory defining the given elementary class. By applying this for the elementary class of ordered abelian semigroups defined by the given axiomatisation T one gets:

**Corollary 2.2** *If* P = NP *classically, then the first-order theory of a general machine (in the sense of theorem 2.1, point 3) is provable in the theory* T.

We recall that for $(\mathbb{R}, +, -, =, 0, 1)$ Knapsack cannot be solved in polynomial time. See [12].

**Definition**: Let T' be the theory obtained from T by replacing the axiom for transitivity of order with the weaker axiom $x < y \wedge y < z \longrightarrow x \leq z$. Recall that $x \leq y$ means $x < y \vee x = y$.

**Remark 2.3** T' *does not prove the first-order theory of any general machine (in the sense of theorem 2.1, point 3).*

**Proof**: Indeed, if we interpret over the field of the reals $\mathbb{R}$ the relation symbol $<$ as the relation $\neq$, we get a model of T' where the algebraic Knapsack problem is known to be undecidable in polynomial time by Meer's Theorem. □

**Proof** of the theorem 2.1:

**Lemma 2.4** *Let* $(S, +, <)$ *be an ordered abelian semi-group with at least two elements such that a special parameter-free unit-cost machine decides Knapsack in a polynomial time* $p(n)$ *over* $S$. *In this situation there is a machine over* $(\mathbb{N}_1, +, <)$ *deciding Knapsack in the same polynomial time* $p(n)$.

**Proof**: Let $x \neq y$ be two elements of $S$. If we assume that $x + x = x$ and $y + y = y$, then the identity $x + y = x + y$ implies $2x + y = x + 2y$. We get $x = y$, contradiction. This means that, say, $x + x \neq x$. If $x + x > x$, then the substructure of $S$ generated by $x$ is isomorphic with $\mathbb{N}_1$, in the other case it is isomorphic with $-\mathbb{N}_1$, so also isomorphic with $\mathbb{N}_1$ modulo an order reversing isomorphism. We restrict the machine to this sub-model of T, reversing the order-tests if necessary. □

**Lemma 2.5** *If the algebraic Knapsack problem can be decided over* $(\mathbb{N}_1, +, <)$ *by a parameter-free machine in polynomial time, then there is a classical deterministic Turing machine solving the decision problem* 3-SAT *in polynomial time, hence* P = NP.

**Proof**: An instance of 3-SAT is a set of clauses of the form:

$$F = (z_{11} \vee z_{12} \vee z_{13}) \wedge \ldots \wedge (z_{m1} \vee z_{m2} \vee z_{m3})$$

where $z_{ij} \in \{x_1, \ldots, x_n\} \cup \{\neg\, x_1, \ldots, \neg\, x_n\}$ and $m \leq (2n)^3$. We take $n$ as complexity cost.

In the following lines we will use the polynomial-time uniform codification of 3SAT in Knapsack given in [22]. One can use as well the similar codification of the NP-complete problem Exact Cover by 3-Sets given in [15], but I prefered the codification given below because it codifies directly 3SAT in Knapsack. 3SAT is seen by many authors as the most robust NP-complete problem.

Following [22] we define now $2n + 2m + 1$ natural numbers, whose decimal representations are bounded in length by $n + m$. Every number consists of a block of $m$ decimal digits followed by a block of $n$ decimal digits.

- The number $b$ consists of an $m$-block $44\ldots4$, followed by an $n$-block $11\ldots1$.

- The $n$ numbers $\text{pos}_i$ have the digit $s \in \{0,1,2,3\}$ on the place $k \leq m$ of the first block iff $x_i$ occurs in the clause $k$ exactly $s$ times non-negated. The $n$-block of $\text{pos}_i$ has an 1 at position $i$ and 0 in rest.

- The $n$ numbers $\text{neg}_i$ have the digit $s \in \{0,1,2,3\}$ on the place $k \leq m$ iff $x_i$ occurs in the clause $k$ exactly $s$ times negated. The $n$-block of $\text{neg}_i$ has an 1 at position $i$ and 0 in rest.

- The $m$ numbers $c_k$ have the digit 1 on the place $k \leq m$ of the $m$-block and 0 for the rest. Their $n$-blocks consist only in zeros.

- The $m$ numbers $d_k$ have the digit 2 on the place $k \leq m$ of the $m$-block and 0 for the rest. Their $n$-blocks consist also only in zeros.

The word $(\text{pos}_i)$, $(\text{neg}_i)$, $(c_k)$, $(d_k)$, $b$ (with separators between successive numbers) is a solution to the classical Knapsack problem iff the given 3-SAT instance is satisfiable. See [22] for the whole proof of this.

The first step of the deterministic Turing machine is to construct the classical Knapsack instance $(\text{pos}_i)$, $(\text{neg}_i)$, $(c_k)$, $(d_k)$, $b$ with separators. This takes a polynomial time. The biggest number computed by the unit-cost $\mathbb{N}$-machine for this input is $2^{p(n)}b$ consisting of $\leq p(n) + m + n \leq p(n) + 8n^3 + n$ many digits.

Now we simulate classically the unit-cost program over $\mathbb{N}_1$. Additions and checking inequalities do not cost a *constant* unit of time anymore, but a time which is also uniformly bounded by a polynomial in $n$. □

Using the Fournier-Koiran Theorem for the case that P = NP classically we get that there is a parameter-free machine solving Knapsack in polynomial time in the structure $(\mathbb{R}, +, -, <, 0, 1)$. This machine does not need even the constants 0 and 1. Getting any input, the machine can calculate 0 by $x_1 - x_1$. After this, if the whole input consists only in zeros, the machine accepts and stops. If not, by the first $x_i \neq 0$ the machine defines $1 := x_i$ if $x_i > 0$, respectively $1 := 0 - x_i$ if $x_i < 0$.

**Lemma 2.6** *The theory of ordered divisible abelian groups is complete, model-complete and decidable. This theory is the model-completion of the theory of ordered abelian groups.*

**Proof**: See [16], [18] or any other classical book of model theory. □

**Corollary 2.7** *If a machine decides the algebraic* Knapsack *in polynomial time over* $(\mathbb{R}, +, -, <)$*, then the same machine will decide the algebraic* Knapsack *in polynomial time over all ordered abelian groups.*

**Proof**: All ordered divisible abelian groups $(G, +, -, <)$ are elementarily equivalent with $\mathbb{R}$ in the given language, and hence model the first-order theory of the machine. Thus the machine will decide Knapsack over any such group. An arbitrary ordered group is embeddable in its divisible closure. The machine deciding Knapsack in polynomial time over some divisible ordered group, computes in fact only in the substructure generated by the input. So it will decide Knapsack in all ordered groups. □

**Lemma 2.8** *Let $S$ be some ordered abelian semi-group (model of* T*). The relation $\cong \subset S \times S$ given by $(x,y) \cong (x',y') \Leftrightarrow x + y' = x' + y$ is an equivalence relation. We denote the equivalence class of $(x,y)$ by $[x,y]$, and let $G$ be the set of equivalence classes. The following definitions:*

$$\begin{aligned} [a,b] + [c,d] &:= [a+c, b+d], \\ -[a,b] &:= [b,a], \\ [a,b] < [c,d] &:\Leftrightarrow a + d < c + b, \\ 0 &:= [a,a], \end{aligned}$$

*endow $G$ with a structure of ordered abelian group. The application $\iota : S \hookrightarrow G$ given by $\iota(x) := [2x, x]$ is a well defined embedding of* T*-models of $(S, +, <)$ in $(G, +, <)$. If $S$ is already an ordered group, then $(G, +, <) = (S, +, <)$ and $\iota = id$.*

$G$ is in general a set strictly containing $S \cup (-S)$ (example: if $S = \{n \in \mathbb{N} \mid n \geq 15\}$ then $G = \mathbb{Z}$).

**Corollary 2.9** *If there is a $(+, -, <)$ machine deciding Knapsack in polynomial time over all ordered abelian groups, then there is a $(+, <)$ general machine deciding Knapsack in polynomial time over all models of* T.

**Proof**: We substitute all registers (cells) $x$ of the group machine with pairs of registers $(x', x'')$. Instructions **input** $x_i$ are replaced by $x'_i := 2x_i$ ; $x''_i := x_i$. Instructions $x := y + z$ are replaced by $x' := y' + z'$ ; $x'' := y'' + z''$. Instructions $x := -y$ are replaced by $x' := y''$ ; $x'' := y'$. Tests **if** $x < y$ **then** ... are replaced by **if** $x' + y'' < y' + x''$ **then** .... □

The Corollary 2.9 closes our proof of theorem 2.1. Of course, starting with some lucky particular semigroup accepting a polynomial time unit-cost Knapsack machine, we showed that the semigroup $(\mathbb{N}\backslash\{0\}, +, <)$ accepts such a parameter-free machine. This implied P = NP classically by observing the polynomial time deterministic 3SAT's embedding in Knapsack. By the Theorem of Fournier and Koiran it implies that Knapsack is decidable with parameter-free polynomial time unit-cost machines in the ordered group of the reals with 1. Because of the absence of multiplication we can substitute 1 with some positive element ad-hoc produced given any particular input that is not identically zero. Using the elementary equivalence of divisible ordered abelian groups and the completeness of the elementary class of all such non-trivial structures we extend the result with one machine for the whole class. By considering the canonical embedding of some ordered abelian group in its divisible hull and again the fact that the machine is parameter-free, we get that the same machine solves Knapsack in a given polynomial time over all ordered abelian groups. Finally, every ordered abelian semi-group can be embedded in a smallest ordered abelian group. By exactly using this embedding and the fact that operations with equivalence classes does not depend of the choice of the reprezentant, we show how the machine can be modified to a machine working over all models of T (ordered abelian semi-groups) and why we further eliminate the constant 0 and the substraction. Now, if there is a parameter-free machine working over all ordered abelian semi-groups, then there is a particular non-trivial semi-group with such a machine, so all implications are in fact equivalences. □

Recall that the multiplicative structure $(\mathbb{N}\backslash\{0\}, \cdot, <)$ is also a model of T. Unit-cost Knapsack over this structure means to get a deterministic algorithm in polynomial time $p(n)$ able to recognize if for $n$ many natural numbers $x_1, x_2, \ldots, x_n$ is true, that $x_n$ is a product of some other $x_j$ with $j < n$. One immediately gets the following corollary:

**Corollary 2.10** *The unit-cost multiplicative Knapsack problem can be solved by a unit-cost polynomial time deterministic algorithm if and only if* P = NP *classically.*

The analogous bit-cost problem Subset Product is known to be NP-complete, see [7] and [23].

# 3 Commutative rings

In this section we provide some examples about the behavior of commutative rings concerning unit-cost algebraic complexity. As proved by Bruno Poizat in [17], if a ring does not allow quantifier elimination then it has P $\neq$ NP.

Now we look closer to such rings without quantifier elimination. Yuri Matiyasevich proved for the ring $(\mathbb{Z}, +, -, \cdot, 0, 1)$ the following:

**Theorem 3.1** *Let $R$ be a relation over $\mathbb{Z}$ ($E \subset \mathbb{Z}^k$). $E$ is a recursively enumerable set according to the classical computability theory if and only if $E$ is existentially definable in the ring $\mathbb{Z}$.*

It follows a translation in the context of the algebraic unit-cost complexity. This lemma seems to be folklore, and I would be surprized if it wasn't already noticed somewhere. However, I didn't see it stated in this form so far:

**Lemma 3.2** *The ring of rational integers $(\mathbb{Z}, +, -, \cdot, 0, 1)$ considered as a model of computation with unit-cost has* NBP $\neq$ NP *and its polynomial hierarchy does not collapse at any level.*

**Proof**: We recall that a problem over $\mathbb{Z}$ is a subset of the disjoint union $\coprod_{k\in\mathbb{N}} \mathbb{Z}^k$. For a complexity class $C$ we denote by $C \cap \mathbb{Z}$ the set of all $A \cap \mathbb{Z}$ for problems $A \in C$. We see that $\mathrm{NP}(\mathbb{Z}) \cap \mathbb{Z}$ is the class $\Sigma_1$ of all recursively enumerable predicates in the sense of classical computability. On the other hand $\mathrm{NBP}(\mathbb{Z}) \cap \mathbb{Z}$ consists only in recursive predicates. Because there are recursively enumerable predicates which are not recursive we get $\mathrm{NBP}(\mathbb{Z}) \neq \mathrm{NP}(\mathbb{Z})$. Moreover, for all classes $\Sigma_i\mathrm{P}(\mathbb{Z})$ in the polynomial hierarchy holds $\Sigma_i\mathrm{P}(\mathbb{Z}) \cap \mathbb{Z} = \Sigma_i$. The arithmetical hierarchy does not collapse at any level, so the polynomial hierarchy of $\mathbb{Z}$ according to unit-cost algebraic complexity doesn't collapse at any level either. □

**Lemma 3.3** *There is a problem $A \in \mathrm{NP}(\mathbb{Z})$ such that for all $k \geq 1$ the set $A_k = A \cap \mathbb{Z}^k$ is (classically) recursively enumerable but not recursive.*

**Proof**: Let $P(u, \vec{a})$ be a polynomial in one parameter that defines a recursively enumerable non-recursive subset of $\mathbb{N}$ by $\exists \vec{a} \in \mathbb{Z}\ P(u, \vec{a}) = 0$. We recall that over $\mathbb{Z}$ positive elements are sums of four squares and $z = \text{Cantor}\ (x, y)$ if $2z = (x + y)(x + y + 1) + 2y$. For input $x_1, \ldots, x_n$ the NP-machine verifies if all $x_i$ are positive, guesses $y_1, \ldots, y_{n-1}$, for $i = 1$ verifies that $y_1 = \text{Cantor}\ (x_1, x_2)$ and for $i = 2$ to $n - 1$ verifies that $y_i = \text{Cantor}\ (y_{i-1}, x_{i+1})$. Finally, the machine guesses $a_1, \ldots, a_t$ and verifies that $P(y_{n-1}, \vec{a}) = 0$. The non-deterministic time is linear. □

**Definition**: A boolean ring is a ring with 1 modelling the axiom $\forall x\ \ x^2 = x$. Given any set $S$ we call ring of sets any subring of the power set $2^S$ with $0 := \emptyset$, $1 := S$, $x + y :=$ the symmetric difference $x \,\Delta\, y$ and $xy := x \cap y$. □

Boolean rings are commutative and have characteristic 2: $\forall x\ x + x = 0$. Every boolean ring is isomorphic with a ring of sets.

**Definition**: For some ring $R$ let $D(R)$ be the following problem:

$$D(R) := \{(x_1, \ldots, x_n) \mid n \in \mathbb{N} \text{ and there are } \varepsilon_1, \ldots, \varepsilon_n \in \{0, 1\}$$
$$\text{such that } (x_1 + \varepsilon_1)(x_2 + \varepsilon_2) \ldots (x_n + \varepsilon_n) = 0\}.$$

Recall that in a boolean ring $x + 1$ is the complement of $x$. The following fact is proved by Bruno Poizat in [17] for infinite atomless boolean rings and by the present author in [19] in the following form:

**Lemma 3.4** *Let $R$ be an infinite boolean ring. Then the problem $D(R)$ belongs to* NBP$(R)$ *but not to* P$(R)$. *Hence infinite boolean rings have* P $\neq$ NBP.

Now we patch the rings together.

**Definition**: Let us fix an infinite boolean ring $B$. We denote by $\mathcal{R}$ the direct product $B \times \mathbb{Z}$ with $0 := (0, 0)$, $1 := (1, 1)$ and operations defined componentwise.

**Theorem 3.5** *The ring* $(\mathcal{R}, +, -, \cdot, 0, 1)$ *considered as a model of computation with unit-cost has* $\mathrm{P} \neq \mathrm{NBP} \neq \mathrm{NP}$ *and its existential polynomial hierarchy does not collapse at any level.*

**Proof**: We first show that $\mathrm{P} \neq \mathrm{NBP}$. The proof is similar with the proof for atomless boolean rings, as it was given in [17].

Let M be a machine containing constants $c_1, \ldots, c_k \in \mathcal{R}$ where all $c_i$ are pairs $(b_i, z_i)$ deciding $D(\mathcal{R})$ in polynomial time. The elements $b_1, \ldots, b_k$ define a finite subring of $\mathcal{R}$ generated by atoms $\alpha_1, \ldots, \alpha_l$ with $l \leq 2^k$. At least one restriction $B \,|\, \alpha_i$ as boolean algebra must be infinite, say $B_1 := B \,|\, \alpha_1$. Let $C_1$ be the finite subring of $B$ generated by $\alpha_2, \ldots, \alpha_l$. We consider inputs of the form $(x_1, 0)$, $\ldots$, $(x_n, 0)$ with $x_i \in B_1$. Considering this input to be $(x_i, 0, 0) \in B_1 \times C_1 \times \mathbb{Z}$, all computed elements do not quit $B_1 \times C_1 \times \mathbb{Z}$. Take $n$ big enough such that $2^n > p(n) + 1$, where $p(n)$ is the polynomial time used by the machine for inputs of length $n$. Independent elements $x_1, \ldots, x_n$ go along some rejecting path in machine's computation tree. The path contains $\leq p(n)$ equality tests.

This negative input generates a free boolean subalgebra $X$ of $B_1$ with $2^{2^n}$ elements. Only $\leq p(n)$ many atoms $(x_1 + \varepsilon_1) \ldots (x_n + \varepsilon_n)$ with $\vec{\varepsilon} \neq$ the constant vector $\vec{1}$ of $2^n$ have been checked to be $\neq 0$. By removing a well chosen atom from $X$ in all $x_i$ we get a new input $y_i$ such that:

- The $\leq p(n)$ checked products $(y_1 + \varepsilon_1) \ldots (y_n + \varepsilon_n) \neq 0$.
- $(y_1 + 1) \ldots (y_n + 1) \neq 0$.
- There is some $\vec{\varepsilon} \neq \vec{1}$ such that $(y_1 + \varepsilon_1) \ldots (y_n + \varepsilon_n) = 0$.

The machine gives the same answer for positive and negative inputs. Contradiction.

Now we prove that $\mathrm{NBP}(\mathcal{R}) \neq \mathrm{NP}(\mathcal{R})$. Let $A$ be a problem in $\mathrm{NBP}(\mathcal{R})$ and $k > 0$ in $\mathbb{N}$. Let $\pi_2$ be the projection onto $\mathbb{Z}$. The set $\pi_2(A \cap \mathcal{R}^k)$ is always a classically recursive subset of $\mathbb{Z}^k$. To see this, note that $A_k := A \cap \mathcal{R}^k$ has a definition like $\exists \varepsilon_1, \ldots, \varepsilon_k \in \{0,1\}\ \varphi(\vec{x}, \vec{\varepsilon})$. This is equivalent with a disjunction over $2^k$ many $0,1$-tuples, so is a quantifier free formula $\psi(\vec{x})$ with parameters in $\mathcal{R}$. For atomic formulas with parameters we observe that:

$$\begin{aligned} P(\vec{x}) = 0 \quad &\leftrightarrow \quad P_B = 0 \wedge P_{\mathbb{Z}} = 0, \\ P(\vec{x}) \neq 0 \quad &\leftrightarrow \quad P_B \neq 0 \vee P_{\mathbb{Z}} \neq 0, \end{aligned}$$

where $P_R = 0$ is a short form for $R \models \pi_i(P)(\pi_i(\vec{x})) = 0$ for $R = B, \mathbb{Z}$ and $i = 1, 2$. By grouping together atomic formulas that refer to the same ring we find a definition of the form:

$$\vec{x} \in A_k \ \Leftrightarrow \ \bigvee_{i=1}^{s} (\sigma_i^B(\vec{x}) \wedge \tau_i^{\mathbb{Z}}(\vec{x})),$$

where for example $\sigma_i^B(\vec{x})$ means $B \models \sigma_i(\pi_1(x))$, with $\sigma_i$ quantifier-free formula with parameters in $B$. It follows that:

$$\pi_2(A_k) = \bigcup_{\{i \,|\, \exists \vec{x}\ \sigma_i^B(\vec{x})\}} \{\vec{x} \,|\, \tau_i^{\mathbb{Z}}(\vec{x})\},$$

which is a classically recursive set, as a finite union of recursive sets.

But in $\mathrm{NP}(\mathcal{R})$ there are problems such that $\pi_2(A \cap \mathcal{R}^k)$ is recursively enumerable but not recursive. For this, we observe that for all polynomial $P \in \mathbb{Z}[\vec{x}, \vec{y}]$:

$$\{\vec{x} \in \mathcal{R}^k \,|\, \exists\, \vec{y} \in \mathcal{R}^k \quad 2 \cdot P(\vec{x}, \vec{y}) = 0\ \} = B \times A_k$$

where $A_k \subset \mathbb{Z}^k$ is the diophantine set defined by $P$. Here we use the fact that the boolean ring $B$ has characteristic 2. According to the theorem of Matiyasevich there are such sets which are classically recursively enumerable but are not recursive.

Finally we prove that $\mathcal{R}$ has a polynomial hierarchy that does not collapse at any level. We recall that the arithmetical hierarchy is built up as the closure of the class of recursive sets to

projections and complementations. The levels $\Sigma_i$ and $\Pi_i$ with $i \geq 1$ can be got if we start with the recursively enumerable sets instead of the recursive sets. In the following lines let $\vec{u}$ be tuples of fixed length $k \geq 1$. Applying the theorem of Matiyasevich and the properties of the ring $\mathcal{R}$ we get that definitions of the form:

$\exists \vec{x}\ \ 2P(\vec{u},\vec{x}) = 0$ define $\{B \times A \,|\, A \in \Sigma_1\}$,

$\forall \vec{x}\ \ 2P(\vec{u},\vec{x}) \neq 0$ define $\{B \times A \,|\, A \in \Pi_1\}$,

$\exists \vec{x}_1\ \forall \vec{x}_2\ \ 2P(\vec{u},\vec{x}) \neq 0$ define $\{B \times A \,|\, A \in \Sigma_2\}$,

$\forall \vec{x}_1\ \exists \vec{x}_2\ \ 2P(\vec{u},\vec{x}) = 0$ define $\{B \times A \,|\, A \in \Pi_2\}$,

$[\ldots]$

$\exists \vec{x}_1\ \forall \vec{x}_2\ \ldots\ \exists \vec{x}_{2i+1}\ \ 2P(\vec{u},\vec{x}) = 0$ define $\{B \times A \,|\, A \in \Sigma_{2i+1}\}$,

$[\ldots]$

$\exists \vec{x}_1\ \forall \vec{x}_2\ \ldots\ \forall \vec{x}_{2i+2}\ \ 2P(\vec{u},\vec{x}) \neq 0$ define $\{B \times A \,|\, A \in \Sigma_{2i+2}\}$, and so on.

**Definition**: In the formal language of rings, $\Sigma_0 = \Pi_0$ formulas are the quantifier-free formulas, in our case with parameters in the ring. $\Sigma_i$, respectively $\Pi_i$ formulas are exactely the formulas consisting in a prefix of type $\Sigma_i$, respectively $\Pi_i$, preceding a quantifier-free formula. For some ring $R$ we say that some relation $W \subset R^k$ belongs to $\Sigma_i(R)$ if $W$ is defined by a $\Sigma_i$ formula. The same for $\Pi_i$.

We observe that $\Sigma_0(\mathbb{Z})$ does not coincide with the class $\Sigma_0$ of all recursive sets, but for all $i \geq 1$ one has $\Sigma_i(\mathbb{Z}) = \Sigma_i$. Also, we observe that for every ring $R$ the polynomial hierarchy class $\Sigma_i \mathrm{P}(R) \cap R^k = \Sigma_i^k(R)$.

In order to end the proof it is enough to show that:

$$A \in \Sigma_{i+1} \setminus \Sigma_i \quad \Rightarrow \quad B \times A \in \Sigma_{i+1}(\mathcal{R}) \setminus \Sigma_i(\mathcal{R}).$$

The last follows from:

$$B \times A \in \Sigma_i(\mathcal{R}) \quad \Rightarrow \quad A \in \Sigma_i,$$

and the same for $\Pi_i(\mathcal{R})$ and $\Pi_i$. For proving them we need the following fact of wide generality:

**Lemma 3.6** *Let $L$ be a formal language and $\varphi(\vec{x}) \in \Sigma_n(L)$ be some $L$-formula containing free variables. Then there is a formula $\psi(\vec{u},\vec{v})$ which is a positive boolean combination of formulas $\sigma_s(\vec{u})$ and $\tau_s(\vec{v})$ with $\sigma_s, \tau_s \in \Sigma_n(L)$ and $\vec{u},\vec{v}$ both tuples of free variables with the same length as $\vec{x}$ such that for all $L$-structures $\mathfrak{A}$ and $\mathfrak{B}$ and for all $\vec{a} \in \mathfrak{A}$ and $\vec{b} \in \mathfrak{B}$, if $c_1 := (a_1, b_1), \ldots, c_k := (a_k, b_k)$ then:*

$$\mathfrak{A} \times \mathfrak{B} \ \models\ \varphi(\vec{c}) \quad \Leftrightarrow \quad (\mathfrak{A},\mathfrak{B}) \ \models\ \psi(\vec{a},\vec{b}) = \bigvee (\sigma_s^{\mathfrak{A}}(\vec{a}) \wedge \tau_s^{\mathfrak{B}}(\vec{b})).$$

*The same is true for all $\Pi_n$-formulas $\varphi$ with corresponding $\sigma_s$ and $\tau_s$ also in $\Pi_n$.*

**Proof**: We make induction over $n$. For $n = 0$ we have quantifier-free formulas, $\Sigma_0(L) = \Pi_0(L)$. Atomic formulas over a direct product split in conjunctions if they are not negated and in disjuctions of negated formulas if they are negated. We bring the formula in disjunctive normal form and we group the conjuncts in $\sigma^{\mathfrak{A}} \wedge \tau^{\mathfrak{B}}$. Step from $\Sigma_n$ to $\Pi_n$: by the negation of $\psi$ we get a positive boolean combination of $\neg\sigma$ and $\neg\tau$ which are now in $\Pi_n$. Step from $\Pi_n$ to $\Sigma_{n+1}$: if we add a new existential quantifier block, it commutes with the disjunction and splits in the conjunctions:

$$\exists\, (\vec{u}', \vec{v}')\ \sigma^{\mathfrak{A}}(\vec{u},\vec{u}') \wedge \tau^{\mathfrak{B}}(\vec{v},\vec{v}') \quad \leftrightarrow \quad \exists \vec{u}'\, \sigma^{\mathfrak{A}}(\vec{u},\vec{u}') \wedge \exists \vec{v}'\, \tau^{\mathfrak{B}}(\vec{v},\vec{v}').$$

The new $\sigma := \exists \vec{u}'\, \sigma^{\mathfrak{A}}(\vec{u},\vec{u}')$ and $\tau := \exists \vec{v}'\, \tau^{\mathfrak{B}}(\vec{v},\vec{v}')$ belong now to $\Sigma_{n+1}$. $\square$

Now, let $B \times A$ be element in $\Sigma_i(\mathcal{R})$. According to the lemma is:

$$B \times A = \bigcup_{s=1}^{m} B_s \times A_s,$$

with $B_s \in \Sigma_i(B)$ and $A_s \in \Sigma_i(\mathbb{Z}) = \Sigma_i$ for $i \geq 1$. It follows:

$$A = \bigcup_{B_s \neq \emptyset} A_s,$$

so $A \in \Sigma_i$ as a finite union of $\Sigma_i$ sets. Similarly if $B \times A \in \Pi_i(\mathcal{R})$ then $A \in \Pi_i$. □

We observe also that $\mathcal{R}$ is a non-boolean ring with P $\neq$ NBP. A non-boolean ring models the axiom $\exists x\ x^2 \neq x$. Moreover, the ring $B \times \mathbb{C}$ (where $B$ atom-free boolean and $\mathbb{C}$ the field of complex numbers) has P $\neq$ NBP (hence also P $\neq$ NP, with the same proof), is non-boolean, and *admits quantifier-elimination* in the language expanded with the constants $\alpha := (1,0)$ and $\beta := (0,1)$.

**Aknowledgments**: The results about ordered abelian semi-groups have been done in the time when the author was an assistant at the University of Greifswald. The idea to consider the ring $\mathcal{R} = B \times \mathbb{Z}$ appeared in Freiburg in discussions with Martin Ziegler, Mark Weyer and Olivier Roche. Martin Ziegler suggested also to formulate lemma 3.6 in its present generality.

The author is grateful to all three anonymous referees who did very worthy suggestions for improving both comprehension and readability. During this process the editors of these Dagstuhl Seminar Proceedings have given the proof of their patience and helpfulness.

# References

[1] **Lenore Blum, Felipe Cucker, Michael Shub, Steven Smale**: *Complexity and Real Computation.* Springer-Verlag, New-York, 1998.

[2] **Lenore Blum, Felipe Cucker, Michael Shub, Steven Smale**: *Algebraic settings for the problem "* P $\neq$ NP *?"* in *The mathematics of numerical analysis* (Park City, UT, 1995) 125 - 144, Lectures in Applied Mathematics 32, A. M. S., Providence, RI, 1996.

[3] **Lenore Blum, Michael Shub, Steven Smale**: *On a theory of computation and complexity over the real numbers.* Bulletin A.M.S. 21, 1989.

[4] **Hervé Fournier, Pascal Koiran**: *Lower bounds are not easier over the reals: inside PH.* Montanari, Ugo (ed.) et al., Automata, languages and programming. 27th international colloquium, ICALP 2000, Geneva, Switzerland, July 9-15, 2000. Proceedings. Berlin: Springer. Lecture Notes Comput. Sci. 1853, 832 - 843 (2000).

[5] **Chrisine Gaßner**: *The P-DNP Problem for Infinite Abelian Groups.* Journal of Complexity 17, 2001.

[6] **Yuri V. Matiyasevich**: *Hilbert's Tenth Problem.* The MIT Press, 1993.

[7] **Michael R. Garey, David S. Johnson**: *Computers and intractability.* W. H. Freeman and Company, New York, 1979.

[8] **John B. Goode**: *Accessible telephone directories.* Journal of Symbolic Logic, 59, 1, 92 - 105, 1994.

[9] **Friedhelm Meyer auf der Heide**: *A polynomial linear search algorithm for the n-dimensional Knapsack problem.* Journal of the Association for Computing Machinery, Vol. 31, No. 3, 668 - 676, 1984.

[10] **Armin Hemmerling**: *On P versus NP for parameter-free programs over algebraic structures.* Mathematical Logic Quarterly 47, 1, 67-92, 2001.

[11] **R. M. Karp**: *Reducibility among combinatorial problems.* in Complexity and Computer Computations by R. E. Miller and J. W. Thatcher (eds), Plenum Press, 1972.

[12] **Klaus Meer**: *Real Number Models under Various Sets of Operations.* Journal of Complexity 9, 1993.

[13] **Klaus Meer, Christian Michaux**: *A survey on real structural complexity theory.* Bull. Belg. Math. Soc. – Simon Stevin 4, 1, 113 - 148, 1997.

[14] **Christian Michaux**: P $\neq$ NP *over the non-standard reals implies* P $\neq$ NP *over* $\mathbb{R}$. Theoretical Computer Science 133, 95 - 104, 1994.

[15] **Christos H. Papadimitriou**: *Computational Complexity.* Addison Wesley, 1994.

[16] **Alexander Prestel**: *Einführung in die Mathematische Logik und Modelltheorie.* Vieweg Studium, 1986.

[17] **Bruno Poizat**: *Les petits cailloux.* ALEAS, Lyon, 1995.

[18] **Bruno Poizat**: *A course in model theory. An introduction to contemporary mathematical logic.* Universitex, Springer Verlag, N.Y. 2001.

[19] **Mihai Prunescu**: *$P \neq NP$ for all infinite Boolean algebras.* Mathematical Logic Quarterly 49, 2, 210 - 213, 2003.

[20] **Mihai Prunescu**: *A model-theoretical proof for* P $\neq NP$ *over all infinite Abelian groups.* Journal of Symbolic Logic 67, 1, 235 - 238, 2002.

[21] **Hartley Rogers, Jr.**: *Theory of Recursive Functions and Effektive Computability*, The MIT Press, 1987 (5. Edition 2002).

[22] **Uwe Schöning**: *Theoretische Informatik - kurz gefaßt.* Spektrum Verlag, 1997.

[23] **A. C. Yao**: *private communication to M. R. Garey and D. S. Johnson*, 1978.