The Complexity of Membership Problems for Circuits over Sets of Integers

Stephen Travers

Report No. 334 Mai 2004

Preprint-Reihe
Fakultät für Mathematik und Informatik
Universität Würzburg

The Complexity of Membership Problems for Circuits over Sets of Integers

Stephen D. Travers

Theoretische Informatik
Bayerische Julius-Maximilians-Universität Würzburg
Am Hubland, D-97074 Würzburg, Germany
travers@informatik.uni-wuerzburg.de

Abstract. We investigate the complexity of membership problems for $\{\cup, \cap, ^-, +, \times\}$-circuits computing sets of integers. These problems are a natural modification of the membership problems for circuits computing sets of natural numbers studied by McKenzie and Wagner (2003). We show that there are several membership problems for which the complexity in the case of integers differs significantly from the case of the natural numbers: Testing membership in the subset of integers produced at the output of a $\{\cup, +, \times\}$-circuit is NEXPTIME-complete, whereas it is PSPACE-complete for the natural numbers. As another result, evaluating $\{^-, +\}$-circuits is shown P-complete for the integers and PSPACE-complete for the natural numbers. The latter result extends work by McKenzie and Wagner (2003) in nontrivial ways. Furthermore, evaluating $\{\times\}$-circuits is shown $\mathrm{NL} \wedge \oplus\mathrm{L}$-complete, and several other cases are resolved.

1 Introduction

In complexity theory, combinational circuits play an important role. There is a variety of different kinds of circuits, and those over the boolean semiring certainly belong to the best investigated. Circuits over more general algebraic structures can be considered as well. A circuit over a universe U with operations $o_1, \ldots, o_r$ defined on the elements of U describes a way to compute an element $a \in U$.

In this paper, we study circuits over the power set of the integers. Besides the arithmetical gates $+$ and $\times$, which compute set-theoretic addition and multiplication, we also allow gates computing the set operations $\cup, \cap$ and $^-$. The main question is the following: Given an integer b and a $\{\cup, \cap, ^-, +, \times\}$-circuit C with integer inputs, does b belong to the set computed by C?
This is the membership problem $\mathrm{MC}_{\mathbb{Z}}(\cup, \cap, ^-, +, \times)$ and $\mathrm{MF}_{\mathbb{Z}}(\cup, \cap, ^-, +, \times)$ is the same problem restricted to formulas. The notion for other restricted versions of these problems, for example $\mathrm{MC}_{\mathbb{Z}}(^-, +)$, is self-explanatory. The complexity of such membership problems varies considerably in dependence of the kinds of gates allowed in the circuit.

In 2003, McKenzie and Wagner investigated the complexity of analogously defined membership problems for circuits over sets of natural numbers, and presented their results at STACS 2003 [MW03]. We denote these problems with $\mathrm{MC}_{\mathbb{N}}(\ldots)$ and $\mathrm{MF}_{\mathbb{N}}(\ldots)$, respectively.

Changing the universe to the power set of the integers instead of the power set of the natural numbers, we investigate the complexity of these modified membership problems. As will be seen, for many of these new problems the complexity coincides with the corresponding problem in the $\mathbb{N}$-case. Interestingly, there also are several membership problems for which the complexity in the two cases differs significantly. We highlight here some of the differences.

Membership problem	$\mathbb{N}$-case ($X = \mathbb{N}$)	$\mathbb{Z}$-case ($X = \mathbb{Z}$)
$\mathrm{MC}_X(\cup, +, \times)$	PSPACE-complete	NEXPTIME-complete
$\mathrm{MC}_X(\times)$	NL-complete	NL$\wedge\oplus$L-complete
$\mathrm{MC}_X(^-, +)$	PSPACE-complete	P-complete
$\mathrm{MF}_X(^-, +)$	PSPACE-complete	L-complete

It turns out that it is not always possible to reduce the $\mathbb{N}$-case to the $\mathbb{Z}$-case, e.g. it holds that $\mathrm{MF}_{\mathbb{N}}(^-, +) \not\leq_m^{\log} \mathrm{MF}_{\mathbb{Z}}(^-, +)$ and $\mathrm{MC}_{\mathbb{N}}(^-, +) \leq_m^{\log} \mathrm{MC}_{\mathbb{Z}}(^-, +)$ iff P = PSPACE.

This paper is organized as follows: Section 2 gives the basic definitions and preliminaries. Section 3 is a digression to membership problems for circuits over sets of natural numbers. Since the problems $\mathrm{MF}_{\mathbb{N}}(^-, +)$ and $\mathrm{MC}_{\mathbb{N}}(^-, +)$ were omitted in [MW03], we provide a proof of their PSPACE-completeness. As an intermediate step, we introduce the problem *Quantified Sum of Subset* and prove that it is PSPACE-complete. In section 4, we analyze differences in complexity between membership problems in the $\mathbb{N}$-case and the $\mathbb{Z}$-case. Section 5 then presents several membership problems with the same complexity in both $\mathbb{N}$- and $\mathbb{Z}$-case. Finally, we conclude with several open problems. A summary of our results can be found in Table 1 at the end of the paper.

2 Preliminaries

We fix the alphabet $\Sigma = \{0, 1\}$. Σ^* is the set of words, and $|w|$ is the length of a word $w \in \Sigma^*$. We denote with L, NL, P, NP, coNP, PSPACE and NEXPTIME the standard complexity classes whose definitions can be found in any textbook on computational complexity (cf. [Pa94], for example).
Furthermore, we need the function class #L and the complexity classes $\oplus$L and $\mathrm{C_=L}$. For a nondeterministic logarithmic space machine M, define $\mathrm{acc}_M(x)$ as the number of accepting paths of M on input x. The class #L consists of precisely these functions. A set A is in $\oplus$L if there exists $f \in$ #L such that $x \in A \Leftrightarrow f(x) \equiv 1(2)$ for every $x \in \Sigma^*$. A set A is in $\mathrm{C_=L}$ if there exist

$f, g \in \#\mathrm{L}$ such that $x \in A \Leftrightarrow f(x) = g(x)$ for every $x \in \Sigma^*$. See [All97] for a survey on these counting classes. For complexity classes $\mathcal{K}$ and $\mathcal{M}$ we define $\mathcal{K} \wedge \mathcal{M} =_{\mathrm{def}} \{A \cap B : A \in \mathcal{K}, B \in \mathcal{M}\}$.For sets A and B we say that A is many-one logspace reducible to B and write $A \leq_m^{\log} B$ if there exists a logarithmic space computable function f such that $x \in A \Leftrightarrow f(x) \in B$ for every $x \in \Sigma^*$.

$\mathbb{N}$ denotes the set of the natural numbers including 0, $\mathbb{Z}$ denotes the set of the integers. We denote the absolute value of an integer z with $\mathrm{abs}(z)$. For any integer b, let $\mathrm{sgn}(b) =_{\mathrm{def}} 0$, if $b < 0$ and $\mathrm{sgn}(b) =_{\mathrm{def}} 1$, otherwise. Like natural numbers, integers are represented in binary but have an additional sign; 0 does not have a sign. Moreover, we will sometimes use a more general k-nary representation for integers: For $z > 0$ there exist numbers $m \geq 0$ and $a_0, a_1, \ldots, a_m \in \{0, \ldots, k-1\}$ such that $z = \sum_{i=0}^{m} a_i \cdot k^i$. The k-nary representation of z is $a_m a_{m-1} \ldots a_1 a_0$. This representation is –except for leading zeros— unique. Conversely, for $b_0, \ldots, b_m \in \{0, .., k-1\}$ $(b_m b_{m-1} \ldots b_0)_k$ denotes that integer, whose k-nary representation is $b_m b_{m-1} \ldots b_0$. For $z < 0$, the k-nary representation is defined analogously with an additional sign. The k-nary representation of $z = 0$ is 0.
We extend the arithmetical operations $+$ and $\cdot$ to subsets of $\mathbb{Z}$: For $M, N \subseteq \mathbb{Z}$ we define the sum of M and N as $M + N =_{\mathrm{def}} \{m + n : m \in M \text{ und } n \in N\}$. For $M, N \subseteq \mathbb{Z}$, we define the product of M and N as $M \odot N =_{\mathrm{def}} \{m \cdot n : m \in M \text{ und } n \in N\}$.

If the complement of a set M is finite, we call M *co-finite*.

A *circuit* $C = (G, E, g_C)$ is a finite, directed, acyclic graph (G, E), where G is the set of nodes and E is the set of vertices. The graph can contain multi-edges and does not necessarily have to be connected. As we consider circuits, we will call nodes *gates* from now on. C contains a specified gate g_C, the *output gate*. Gates with indegree 0 are called the *input gates*. Let $\mathcal{O} \subseteq \{\cup, \cap, ^-, +, \times\}$. An $\mathcal{O}$-circuit $C = (G, E, g_C, \alpha)$ is a circuit (C, E, g_C), whose gates have indegree 0, 1, or 2 and are labelled by the function $\alpha : G \to \mathcal{O} \cup \mathbb{Z}$ in the following way: Every input gate is labelled with an integer, every gate with indegree 1 is labelled with $^-$, and every gate with indegree 2 with $\cup, \cap, +,$ or $\times$. For each of its gates g, C computes a set $I(G) \subseteq \mathbb{Z}$, inductively defined as follows:

- If g is an input gate with $\alpha(g) = a$ then $I(g) =_{\mathrm{def}} \{a\}$.
- If g is a $+$-Gate with predecessors g_1, g_2 then $I(g) =_{\mathrm{def}} \{a+b : a \in I(g_1) \wedge b \in I(g_2)\}$
- If g is a $\times$-Gate with predecessors g_1, g_2 then $I(g) =_{\mathrm{def}} \{a \cdot b : a \in I(g_1) \wedge b \in I(g_2)\}$.
- If g is a $\cup$-Gate with predecessors g_1, g_2 then $I(g) =_{\mathrm{def}} I(g_1) \cup I(g_2)$.
- If g is a $\cap$-Gate with predecessors g_1, g_2 then $I(g) =_{\mathrm{def}} I(g_1) \cap I(g_2)$.
- If g is a $^-$-Gate with predecessor g_1 then $I(g) =_{\mathrm{def}} \mathbb{Z} \setminus I(g_1)$.

The set computed by C is $I(C) =_{\mathrm{def}} I(g_C)$. If a gate $g \in G$ computes a singelton $\{a\}$, we will sometimes write $I(g) = a$ for simplicity. An *$\mathcal{O}$-formula* is an $\mathcal{O}$-circuit with maximal outdegree 1.

For $\mathcal{O} \subseteq \{\cup, \cap, ^-, +, \times\}$ we define *membership problems* for $\mathcal{O}$-circuits and $\mathcal{O}$-formulae over sets of integers by

$$\mathrm{MC}_{\mathbb{Z}}(\mathcal{O}) =_{\mathrm{def}} \big\{(C, b) : C \text{ is an } \mathcal{O}\text{-circuit, } b \in \mathbb{Z} \text{ and } b \in I(C)\big\}$$

and

$$\mathrm{MF}_{\mathbb{Z}}(\mathcal{O}) =_{\mathrm{def}} \big\{(C, b) : C \text{ is an } \mathcal{O}\text{-formula, } b \in \mathbb{Z} \text{ and } b \in I(C)\big\}.$$

For simplicity, we write $\mathrm{MC}_{\mathbb{Z}}(o_1, \ldots, o_r)$ instead of $\mathrm{MC}_{\mathbb{Z}}(\{o_1, \ldots, o_r\})$ and $\mathrm{MF}_{\mathbb{Z}}(o_1, \ldots, o_r)$ instead of $\mathrm{MF}_{\mathbb{Z}}(\{o_1, \ldots, o_r\})$.

We denote membership problems for circuits over sets of natural numbers (see [MW03] for details) by $\mathrm{MC}_{\mathbb{N}}(\mathcal{O})$ and $\mathrm{MF}_{\mathbb{N}}(\mathcal{O})$, respectively. These problems are defined analogously. The only differences are that we solely allow input gates with non-negative labels and that $^-$-gates compute the complement with respect to $\mathbb{N}$. To avoid confusion, we denote the set computed by a circuit C over sets of natural numbers by $I_{\mathbb{N}}(C)$.

We assume any appropriate circuit and formula encoding, where gates are sorted topologically and neighborhoods are readily available. All completeness results are in terms of many-one logspace reducibility.

Example 1. Defining the circuits $\mathrm{Z} =_{\mathrm{def}} \overline{\{0\}} \cup \{0\}$ and $\mathrm{ODD} =_{\mathrm{def}} (\{2\} \times \mathrm{Z}) + \{1\}$, we obtain $I(\mathrm{Z}) = \mathbb{Z}$ and $I(\mathrm{ODD}) = \{z \in \mathbb{Z} : z \text{ odd}\}$.

Example 2. Consider the circuit $\mathbb{Z}$-POWER2 $=_{\mathrm{def}} \overline{(\mathrm{ODD} \cap \overline{-1 \cup 1}) \times \mathrm{Z}}$. It then holds that $I(\mathbb{Z}\text{-POWER2}) = \{2^k : k \geq 0\} \cup \{-2^k : k \geq 0\}$.

Example 3. The circuit $\mathbb{Z}$-PRIMES $=_{\mathrm{def}} C \cap \overline{C \times C}$, where C is a subcircuit defined as $\overline{-1 \cup 0 \cup 1}$, computes the set $\{p : p \text{ is prime}\} \cup \{-p : p \text{ is prime}\}$.

3 Digression: $\mathrm{MF}_{\mathbb{N}}(^-, +)$ is PSPACE-complete

Before we start our analysis of membership problems over sets of integers, we draw our attention to circuits over sets of natural numbers. In this section we extend work by McKenzie and Wagner [MW03] by proving the problem $\mathrm{MC}_{\mathbb{N}}(^-, +)$ to be PSPACE-complete.
As will be seen later, this result is of importance to us since the case $\{^-, +\}$ is one where the complexity of the $\mathbb{N}$- and $\mathbb{Z}$-membership problems diverges the most.

As an auxiliary tool, we define a new PSPACE-complete problem by introducing alternation into the well known NP-complete *sum of subset* problem: The problem *quantified sum of subset* is defined as

$$\mathrm{QSOS} =_{\mathrm{def}} \Big\{(a_1, \ldots, a_n, b) : a_1, \ldots, a_n, b \in \mathbb{N}, n \equiv 1(2) \text{ and } \underbrace{\exists_{c_1 \in \{0,1\}} \forall_{c_2 \in \{0,1\}} \cdots \exists_{c_n \in \{0,1\}}}_{\textit{strict alternation}} \Big(\sum_{i=1}^{n} c_i a_i = b\Big)\Big\}.$$

It is known that the problem of evaluating quantified boolean formulae (QBF) is PSPACE-complete [SM73]. QBF is defined as

$$\mathrm{QBF} =_{\mathrm{def}} \big\{H : H \textit{ is a boolean formula in } \text{3-CNF} \textit{ with variables } x_1, \ldots, x_n \text{ such that } \underbrace{\exists x_1 \forall x_2 \cdots \exists x_n}_{\textit{strict alternation}} \big(H(x_1, \ldots, x_n) = 1\big)\big\}.$$

Lemma 1. *It holds that* QBF $\leq_m^{\log}$ QSOS.

Proof. We define a logspace computable funtion f such that $H \in \mathrm{QBF} \Leftrightarrow f(H) \in$ QSOS. Let $H = \bigwedge_{i=1}^{m}(z_{i_1} \vee z_{i_2} \vee z_{i_3})$ with $z_{i_j} \in \{x_1, \ldots, x_n, \neg x_1, \ldots, \neg x_n\}$ be a boolean formula.

We then define f as

$$f(H) =_{\mathrm{def}} \big(v_1, v_2, \ldots, v_n, 0, v_1', 0, v_2', 0, \ldots, v_n', 0, c_1, 0, \ldots, c_m, 0, d_1, 0, \ldots, d_m, b\big),$$

where $b, v_1, \ldots, v_n, v_1', \ldots, v_n'$ are natural numbers which have the following decimal representations:

$$v_i =_{\mathrm{def}} \underbrace{k_1 k_2 \ldots k_m}_{m} \underbrace{0 \ldots 0\overset{(i)}{1}0 \ldots 0}_{n}, \text{ where}$$

k_j is the number of occurrences of literal x_i in the j-th clause of H,

$$v_i' =_{\mathrm{def}} \underbrace{k_1' k_2' \ldots k_m'}_{m} \underbrace{0 \ldots 0\overset{(i)}{1}0 \ldots 0}_{n}, \text{ where}$$

k_j' is the number of occurrences of literal $\neg x_i$ in the j-th clause of H,

$$c_i =_{\mathrm{def}} \underbrace{0 \ldots 0\overset{(i)}{1}0 \ldots 0}_{m} \underbrace{0 \ldots 00 \ldots 0}_{n}, \quad \text{(balancing vectors)}$$

$$d_i =_{\mathrm{def}} \underbrace{0 \ldots 0\overset{(i)}{2}0 \ldots 0}_{m} \underbrace{0 \ldots 00 \ldots 0}_{n}, \quad \text{(balancing vectors)}$$

$$b =_{\mathrm{def}} \underbrace{4 \ldots 444 \ldots 4}_{m} \underbrace{1 \ldots 11 \ldots 1}_{n} \quad \text{(target vector).}$$

By defining f in this way, we achieve that all vectors are appropriately quantified in the QSOS-instance: $v_1, \ldots, v_n$ are quantified strictly alternating and all other vectors are quantified existentially.

In the following, let $I_{a_1,\ldots,a_n}$ be the interpretation which, for $1 \leq i \leq n$, assigns truth-value $a_i \in \{0, 1\}$ to variable x_i in H.

The following equivalences now hold:

$$
\begin{aligned}
H \in \mathrm{QBF} &\Leftrightarrow \exists a_1 \forall a_2 \cdots \exists a_n (I_{a_1,\dots,a_n} \text{ satisfies } H)\\
&\Leftrightarrow \exists_{a_1\in\{0,1\}} \forall_{a_2\in\{0,1\}} \cdots \exists_{a_n\in\{0,1\}} (I_{a_1,\dots,a_n} \text{ satisfies each clause of } H)\\
&\Leftrightarrow \exists_{e_1\in\{0,1\}} \forall_{e_2\in\{0,1\}} \cdots \exists_{e_n\in\{0,1\}}\\
&\qquad \exists_{k_1,\dots,k_m\in\{1,2,3\}} \Big(\big((\sum_{i=1}^{n} e_i v_i + (1-e_i) v_i') \big) = k_1 k_2 \dots k_m \underbrace{1\dots1}_{n} \Big)\\
&\Leftrightarrow \exists_{e_1\in\{0,1\}} \forall_{e_2\in\{0,1\}} \cdots \exists_{e_n\in\{0,1\}} \exists_{l_1\in\{0,1\}} \exists_{l_2\in\{0,1\}} \cdots \exists_{l_n\in\{0,1\}}\\
&\qquad \exists_{k_1,\dots,k_m\in\{1,2,3\}} \Big(\big((\sum_{i=1}^{n} e_i v_i) + (\sum_{i=1}^{n} l_i v_i') \big) = k_1 k_2 \dots k_m \underbrace{1\dots1}_{n} \Big)\\
&\Leftrightarrow \exists_{e_1\in\{0,1\}} \forall_{e_2\in\{0,1\}} \cdots \exists_{e_n\in\{0,1\}} \exists_{l_1,\dots,l_n\in\{0,1\}} \exists_{r_1,\dots,r_m,r_1',\dots,r_m'\in\{0,1\}}\\
&\qquad \Big(\big((\sum_{i=1}^{n} e_i v_i + l_i v_i') + (\sum_{i=1}^{m} r_i c_i + r' d_i) \big) = \underbrace{44\dots4}_{m}\underbrace{1\dots1}_{n} \Big)\\
&\Leftrightarrow f(H) \in \mathrm{QSOS}.
\end{aligned}
$$

Since f is computable in logarithmic space, our reduction is complete. $\square$

Lemma 2. *It holds that* $\mathrm{QSOS} \leq_m^{\log} \mathrm{MF}_{\mathbb{N}}(^-, +)$.

Proof. Let $(a_1, \dots, a_n, b) \in \mathbb{N}^{n+1}$ for any odd n. Observe that $(a_1, \dots, a_n, b) \in \mathrm{QSOS} \Leftrightarrow (a_1, \dots, a_n, 0, (\sum_{i=1}^{n} a_i) + 1, b + (\sum_{i=1}^{n} a_i) + 1) \in \mathrm{QSOS}$. Hence, without loss of generality, we may assume that $b \geq \sum_{i=1}^{n-1} a_i$. For $i = 1, \dots, n$ we construct formulas A_i:

$$
A_i =_{\mathrm{def}} \begin{cases} \overline{1 + \overline{(a_i - 1)}}, & \text{if } a_i \geq 1\\ 0 & \text{otherwise.} \end{cases}
$$

Note that $I_{\mathbb{N}}(A_i) = \{0, a_i\}$. We now construct a $\{^-, +\}$-formula F as follows:

$F_n =_{\mathrm{def}} A_n$,
$F_{k-1} =_{\mathrm{def}} A_{k-1} + \overline{F_k}$ for $2 \leq k \leq n$ and
$F =_{\mathrm{def}} F_1$.

Claim: It holds that

$$
b \in I_{\mathbb{N}}(F) \Leftrightarrow \exists_{a_1' \in I_{\mathbb{N}}(A_1)} \forall_{a_2' \in I_{\mathbb{N}}(A_2)} \cdots \exists_{a_n' \in I_{\mathbb{N}}(A_n)} ((\textstyle\sum_{i=1}^{n} a_i') = b).
$$

Proof of the claim. We prove the claim for all $n = 2k + 1$, $k \in \mathbb{N}$.
(IB) $(k = 0) \Rightarrow n = 1$: Since $F = A_1$, we have $b \in I(F) \Leftrightarrow \exists_{a_1' \in I_{\mathbb{N}}(A_1)} (b = a_1')$.
(IS) $(k \to k + 1) \Rightarrow (n \to n + 2)$: $b \geq \sum_{i=1}^{n+1} a_i$. As argued above, we can assume that $b \geq \sum_{i=1}^{n}$. Consequently, $F = A_1 + \overline{A_2 + \overline{F'}}$, where F' is a $\{^-, +\}$-formula

for which the induction hypothesis is true. We obtain

$$
\begin{aligned}
b \in I_{\mathbb{N}}(F) &\Leftrightarrow \exists_{a_1' \in I_{\mathbb{N}}(A_1)}(b - a_1' \in I_{\mathbb{N}}(\overline{A_2 + \overline{F'}})) \\
&\Leftrightarrow \exists_{a_1' \in I_{\mathbb{N}}(A_1)}(b - a_1' \notin I_{\mathbb{N}}(A_2 + \overline{F'})) \\
&\Leftrightarrow \exists_{a_1' \in I_{\mathbb{N}}(A_1)} \forall_{a_2' \in I_{\mathbb{N}}(A_2)}(b - a_1' - a_2' \notin I_{\mathbb{N}}(\overline{F'})) \\
&\Leftrightarrow \exists_{a_1' \in I_{\mathbb{N}}(A_1)} \forall_{a_2' \in I_{\mathbb{N}}(A_2)}(b - a_1' - a_2' \in I_{\mathbb{N}}(F')) \\
&\overset{\text{(IH)}}{\Leftrightarrow} \exists_{a_1' \in I_{\mathbb{N}}(A_1)} \forall_{a_2' \in I_{\mathbb{N}}(A_2)} \\
&\qquad \exists_{a_3' \in I_{\mathbb{N}}(A_3)} \forall_{a_4' \in I_{\mathbb{N}}(A_4)} \cdots \exists_{a_{n+2}' \in I_{\mathbb{N}}(A_{n+2})}((\sum_{i=3}^{n+2} a_i' = (b - a_1' - a_2')) \\
&\Leftrightarrow \exists_{a_1' \in I_{\mathbb{N}}(A_1)} \forall_{a_2' \in I_{\mathbb{N}}(A_2)} \cdots \exists_{a_{n+2}' \in I_{\mathbb{N}}(A_{n+2})}((\sum_{i=1}^{n+2} a_i') = b),
\end{aligned}
$$

which proves the claim. So we conclude that

$$
\begin{aligned}
b \in I_{\mathbb{N}}(F) &\Leftrightarrow \exists_{a_1' \in I_{\mathbb{N}}(A_1)} \forall_{a_2' \in I_{\mathbb{N}}(A_2)} \cdots \exists_{a_n' \in I_{\mathbb{N}}(A_n)}((\sum_{i=1}^{n} a_i') = b) \\
&\Leftrightarrow \exists_{c_1 \in \{0,1\}} \forall_{c_2 \in \{0,1\}} \cdots \exists_{c_n \in \{0,1\}}((\sum_{i=1}^{n} c_i a_i) = b) \\
&\Leftrightarrow (a_1, \ldots, a_n, b) \in \text{SQSOS}.
\end{aligned}
$$

Obviously, formula F can be constructed in logarithmic space. Hence, $\mathrm{MF}_{\mathbb{N}}(^-, +)$ is PSPACE-hard. □

Theorem 1. *1.* $\mathrm{MF}_{\mathbb{N}}(^-, +)$ *is* PSPACE-*complete.*
2. $\mathrm{MC}_{\mathbb{N}}(^-, +)$ *is* PSPACE-*complete.*

Proof. This is a direct consequence of Lemma 2 and $\mathrm{MC}_{\mathbb{N}}(\cup, \cap, ^-, +) \in$ PSPACE [MW03]. □

4 Differences from the Case of the Natural Numbers

We start with some simple observations.

Lemma 3. *1. Let* $\mathcal{O} \subseteq \{\cup, \cap, +, \times\}$*. Then* $\mathrm{MC}_{\mathbb{N}}(\mathcal{O}) \leq_m^{\log} \mathrm{MC}_{\mathbb{Z}}(\mathcal{O})$ *and* $\mathrm{MF}_{\mathbb{N}}(\mathcal{O}) \leq_m^{\log} \mathrm{MF}_{\mathbb{Z}}(\mathcal{O})$*.*
2. Let $\mathcal{O} \subseteq \{\cup, \cap, ^-\}$*. Then* $\mathrm{MC}_{\mathbb{N}}(\mathcal{O}) \equiv_m^{\log} \mathrm{MC}_{\mathbb{Z}}(\mathcal{O})$ *and* $\mathrm{MF}_{\mathbb{N}}(\mathcal{O}) \equiv_m^{\log} \mathrm{MF}_{\mathbb{Z}}(\mathcal{O})$*.*

Proof. 1. Let C be an $\mathcal{O}$-circuit over the natural numbers. In the absence of $^-$-gates, all gates of C compute finite sets of non-negative integers, thus we have $(C, b) \in \mathrm{MC}_{\mathbb{N}}(\mathcal{O}) \Leftrightarrow (C, b) \in \mathrm{MC}_{\mathbb{Z}}(\mathcal{O})$ for $b \in \mathbb{N}$. Obviously, the same holds for formulas.

2. Let C be an $\mathcal{O}$-circuit with integer inputs $a_1, \ldots, a_k$, and $b \in \mathbb{Z}$. Let σ be any logspace-computable bijective mapping from $\mathbb{Z}$ to $\mathbb{N}$. We now construct a circuit C' by replacing C's inputs with $\sigma(a_1), \ldots, \sigma(a_k)$. It now holds that $(C, b) \in \mathrm{MC}_{\mathbb{Z}}(\mathcal{O}) \Leftrightarrow (C', \sigma(b)) \in \mathrm{MC}_{\mathbb{N}}(\mathcal{O})$, because C and C' do not contain any arithmetical gates. Since this construction preservers the circuit structure, the same holds for formulas. The other reduction is obvious. $\square$

Hence, we can omit $\mathrm{MC}_{\mathbb{Z}}(\mathcal{O})$ for $\mathcal{O} \subseteq \{\cup, \cap, {}^{-}\}$ from our study, as in these cases the complexity coincides with the complexity of the corresponding $\mathrm{MC}_{\mathbb{N}}(\mathcal{O})$-problem. Furthermore, we can take lower bounds for membership problems for circuits without complementation from [MW03].

The *graph accessibility problem* (GAP) for directed graphs, defined as $\{(G, s, t) :$ G is a directed graph with nodes s and t and there is a path from s to $t\}$ is NL-complete [Sa73].

Theorem 2. *The problem* $\mathrm{MC}_{\mathbb{Z}}(\times)$ *is* $\mathrm{NL} \wedge \oplus\mathrm{L}$*-complete.*

Proof. Containedness: Observe that $\mathrm{MC}_{\mathbb{Z}}(\times) = A \cap B$, where $A =_{\mathrm{def}} \{(C, b) :$ C is a $\{\times\}$-circuit, $b \in \mathbb{Z}$ and $\mathrm{abs}(b) = \mathrm{abs}(I(C))\}$ and $B =_{\mathrm{def}} \{(C, b) : C$ is a $\{\times\}$-circuit, $b \in \mathbb{Z}$ and $\mathrm{sgn}(b) = \mathrm{sgn}(I(C))\}$. Since A easily reduces to $\mathrm{MC}_{\mathbb{N}}(\times)$ which is in NL [MW03], A is in NL. It remains to show that B is in $\oplus$L. We construct a nondeterministic logarithmic space machine M working on input (C, b) as follows:

For every path in C from the output gate to an input gate, M produces a computation path. M accepts precisely on those paths, where the input gate of the corresponding path in C is labelled with a negative integer. If $b \geq 0$, M produces an additional accepting path. It now holds that $\mathrm{acc}_M((C, b)) \equiv 1(2) \Leftrightarrow$ $(C, b) \in B$. Hence, it follows that $B \in \oplus\mathrm{L}$ and we conclude $\mathrm{MC}_{\mathbb{Z}}(\times) \in \mathrm{NL} \wedge \oplus\mathrm{L}$.

Hardness: Let $A \in \mathrm{NL} \wedge \oplus\mathrm{L}$, thus there exist sets $B \in \mathrm{NL}$ and $C \in \oplus\mathrm{L}$ such that $A = B \cap C$. Let $x \in \Sigma^*$. GAP is NL-complete and so is $\overline{\mathrm{GAP}}$ due to [Im88] and [Sz88]. Consequently, we can reduce in logarithmic space B to $\overline{\mathrm{GAP}}$ via f. Let $(G, s, t) = f(x)$. It is easy to construct in logarithmic space a $\{\times\}$-circuit C_1 such that $(G, s, t) \in \overline{\mathrm{GAP}} \Rightarrow 1 \in I(C_1)$ and $(G, s, t) \notin \overline{\mathrm{GAP}} \Rightarrow 0 \in I(C_1)$ (see [MW03], Theorem 8.5). We obtain $x \in B \Leftrightarrow (G, s, t) \in \overline{\mathrm{GAP}} \Leftrightarrow (C_1, 1) \in$ $\mathrm{MC}_{\mathbb{Z}}(\times)$.

Due to $C \in \oplus\mathrm{L}$, there exists a nondeterministic logspace machine M such that $x \in C \Leftrightarrow \mathrm{acc}_M(x) \equiv 1(2)$. Let C_x be the transition graph of M on input x. We transform C_x into a $\{\times\}$-circuit C_2, where accepting and rejecting nodes in C_x become inputs with labels -1 and 1 respectively, and the node representing the starting configuration of M becomes the output gate of C_2. Observe that C_x and C_2 can be constructed in logarithmic space. We obtain $x \in C \Rightarrow I(C_2) = -1$ and $x \notin C \Rightarrow I(C_2) = 1$.

So we can construct in logarithmic space the circuit $C_3 =_{\mathrm{def}} C_1 \times C_2$. It now holds that $x \in A \Leftrightarrow (x \in B \wedge x \in C) \Leftrightarrow (I(C_1) = 1 \wedge I(C_2) = -1) \Leftrightarrow (C_3, -1) \in$ $\mathrm{MC}_{\mathbb{Z}}(\times)$, which yields the desired reduction. $\square$

Remark 1. Under the assumption $\oplus\mathrm{L} \not\subseteq \mathrm{NL}$, $\mathrm{MC}_{\mathbb{Z}}(\times)$ is harder than $\mathrm{MC}_{\mathbb{N}}(\times)$. That is in a way surprising, as at first glance it does not seem more complex to multiply integers than natural numbers. We can in fact spot the difficulty evaluating $\{\times\}$-circuits over the integers: If we forbid -1 as a label for the input gates of a $\{\times\}$-circuit, the membership problem for these altered circuits is again NL-complete. This is due to the fact that -1 is the only negative number that can be multiplied by itself many times without its absolute value becoming too large.

Similarily, we obtain:

Theorem 3. *1. The problem* $\mathrm{MC}_{\mathbb{Z}}(\cap,\times)$ *is hard for* $\mathrm{C}_{=}\mathrm{L} \wedge \oplus\mathrm{L}$*.*
2. The problem $\mathrm{MC}_{\mathbb{Z}}(\cap,\times)$ *is in* P*.*

In contrast to the $\mathbb{N}$-case, we do not have a completeness result for $\mathrm{MC}_{\mathbb{Z}}(+,\times)$. The following lemma states that $\mathrm{MC}_{\mathbb{Z}}(+,\times)$ is more related to $\mathrm{MC}_{\mathbb{N}}(\cap,+,\times)$, for which no completeness result is known either [MW03].

Lemma 4. *It holds that* $\mathrm{MC}_{\mathbb{N}}(\cap,+,\times) \leq_m^{\log} \mathrm{MC}_{\mathbb{Z}}(+,\times)$*.*

Proof. In [MW03], McKenzie and Wagner showed that the problem $\mathrm{EQ}_{\mathbb{N}}(+,\times)$, defined as $\{(C_1,C_2) : C_1$ and C_2 are $\{+,\times\}$-circuits over sets of natural numbers and $I_{\mathbb{N}}(C_1) = I_{\mathbb{N}}(C_2)\}$ is many-one logspace equivalent to $\mathrm{MC}_{\mathbb{N}}(\cap,+,\times)$. Let C_1, C_2 be $\{+,\times\}$-circuits over sets of natural numbers. We construct the $\{+,\times\}$-circuit $C =_{\mathrm{def}} C_1 + \big(C_2 \times (-1)\big)$ and obtain $(C_1,C_2) \in \mathrm{EQ}_{\mathbb{N}}(+,\times) \Leftrightarrow I(C_1) = I(C_2) \Leftrightarrow I(C_1) - I(C_2) = 0 \Leftrightarrow I(C) = 0 \Leftrightarrow (C,0) \in \mathrm{MC}_{\mathbb{Z}}(+,\times)$. $\square$

4.1 NEXPTIME-hard Membership Problems

It is known that $\mathrm{MC}_{\mathbb{N}}(\cup,+,\times)$ is PSPACE-complete [Wa84], [Ya00]. In this section we will show that the corresponding problem over the integers is complete for NEXPTIME. Intuitively, the difficulty when evaluating $\{\cup,+,\times\}$-circuits with integer labels is, that—unlike in the $\mathbb{N}$-case—we have to deal with very large (up to exponential in length) numbers: Our target number has polynomial length, but adding a very small and a very large integer can result in a number with a small absolute value again. In the $\mathbb{N}$-case, numbers can only become smaller when multiplied by 0, hence it suffices to compute numbers in the length of the target number when evaluating a circuit.

For $k > 2$ and $n \geq 1$ we define $A_{k,n} =_{\mathrm{def}} \{a \cdot k^l : 0 \leq l < n, 1 \leq a < k\}$. Hence, the k-nary representation of a vector $v = a \cdot k^l \in A_{k,n}$ is

$$\underbrace{00\ldots0a\overbrace{00\ldots0}^{l}}_{n}.$$

Lemma 5. *1. Let $w = (a_1a_2 \dots a_n)_k$ with $0 \le a_1, \dots, a_n < k$ be the sum of $(n-1)$ (not necessarily distinct) vectors from $A_{k,n}$. Then there exists $i \le n$ such that $a_i = 0$.*

2. Let $w = (a_1a_2 \dots a_n)_k$ be the sum of the (not necessarily distinct) vectors $v_1, \dots, v_n \in A_{k,n}$ such that $1 \le a_1, \dots, a_n < k$. Then there are no three vectors among the addends which have a value unequal to 0 on the same position.

Proof. In the following, when we talk of a digit with value i on position j in a vector v, we mean that the j-th digit of the k-nary representation of vector v has value i. When we say that a position in a vector is filled up, we mean that the digit on that position has a value unequal to 0 (because we added a vector which has a digit with a value unequal to 0 on the same position).

1. Assume that $w = (a_1a_2 \dots a_n)_k$ with $1 \le a_1, \dots, a_n < k$ is the sum of the vectors $v_1, \dots, v_{n-1}$.

Case 1: Summing up the vectors $v_1, \dots, v_{n-1}$ does not yield an overflow. Since each vectors contain precisely one digit with a value unequal to 0, there exists $1 \le i \le n$ such that $a_i = 0$. This is a contradiction.

Case 2: Summing up the vectors $v_1, \dots, v_{n-1}$ yields an overflow. Let $1 \le j \le n$ be the position in w where the rightmost overflow occurs. Hence, there are $m \ge 2$ vectors among the addends which have a digit with value unequal to 0 on position j. It follows that the overflow on position j is at most $\left\lfloor \frac{m \cdot (k-1)}{k} \right\rfloor < m$. Consequently, the overflow can fill up at most $\left\lceil \log_k(\frac{m \cdot (k-1)}{k}) \right\rceil =: f_k(m)$ positions in w. If some of the digits on positions $j-1, j-2, \dots$ already have values unequal to 0, the overflow can move further to the left, but the total number of positions which can be filled with the overflow cannot get larger than $\left\lceil \log_k(\frac{m \cdot (k-1)}{k}) \right\rceil$.
Since j is the rightmost overflow, $n - j$ vectors must have been used to fill up the positions $j+1, \dots, n-1, n$. Hence, $n-j+m$ vectors were used make the leftmost $n - j + 1 + f_k(m)$ positions in w unequal to 0. Observe that $n - j + 1 + f_k(2) = n - j + 2$ and $n - j + 1 + f_k(m') < n - j + m'$ for $m' \ge 3$. The argumentation for overflows occurring further left is the same. Therefore, the number of positions in w which can be made unequal to 0 using $n - 1$ vectors is at most $n - 1$ even if overflows occur.

2. This is a consequence of the last statement: Let $w = (a_1 \dots a_n)_k = \sum_{i=1}^{n} v_i$ such that $1 \le a_1, \dots, a_n < k$, and let j be the rightmost position in w, for which there exist $m \ge 3$ vectors among the addends which have a value unequal to 0 on position j. As argued above, it holds that $n - j + 1 + f_k(m') < n - j + m'$ for $m' \ge 3$, that means the number of positions which can be filled up with the overflow is smaller than the number of vectors used. That means we have wasted a vector, thus we cannot fill the remaining positions in w. ☐

Lemma 6. *The problem* $\mathrm{MC}_{\mathbb{Z}}(\cup, +, \times)$ *is* NEXPTIME-*hard.*

Proof. Let $L \in$ NEXPTIME and M be a nondeterministic Turing machine which decides L and has time bound $2^{p(|x|)}$ for a suitable polynomial p. For an input $x \in \Sigma^*$ we show how to construct in logarithmic space a $\{\cup, +, \times\}$-circuit C_{SOS} such that

$$x \in L \Leftrightarrow (C_{\mathrm{SOS}}, 0) \in \mathrm{MC}_{\mathbb{Z}}(\cup, +, \times).$$

Without loss of generality, we assume that M is normalized in the following way:

- M starts its work on input x in the initial state s_0. The machine head is on cell 0 of the working tape; this cell contains the first symbol of the input x.
- On input x, M works precisely $2^{p(|x|)}$ steps. M never moves the machine head on a cell left from cell 0.
- In each step, M branches and continues its work on precisely two computation paths.
- If M accepts x, it is in the sole accepting state s_1 and the computation halts with the machine head on cell 0. The rest of the tape is empty.

Let $x = x_1 \ldots x_n \in \Sigma^*$ and $N =_{\mathrm{def}} p(|x|)$. A configuration of M on input x at a certain time consists of the contents of the working tape and the number of the state the machine is currently in. We use this number to mark the position of the machine head on the working tape. Hence, we can encode a configuration of M on input x in a string with length $2^N + 1$.
For instance, the string $c_0 =_{\mathrm{def}} \overset{s_0}{x_1}\, x_2 \ldots x_n \square^{2^N - n} \square$ represents the initial configuration of M on input $x_1, \ldots, x_n$. Each computation path p of M on input x can be thought of as a $(2^N + 1) \cdot (2^N + 1)$ computation table $T(M, x, p)$. In this table, the (i, j)-th entry represents the j-th symbol of the configuration M is in after i steps (on path p). We call a computation table *accepting* if the last row of the computation table is the accepting configuration $\overset{s_1}{1}\, \square^{2^N - 1} \square$. Hence, an accepting computation table of M on input x corresponds to an accepting computation path of M on input x. Figure 1 depicts a computation table.

We will see that we can use the concept of a computation table to describe the structure of a highly regular formula F_x in conjunctive normal form (CNF) for which the following holds:

$$\begin{aligned} x \in L &\Leftrightarrow \text{on input } x,\ M \text{ has an accepting path } p \\ &\Leftrightarrow \text{there is an accepting computation table } T(M, x, p) \\ &\Leftrightarrow F_x \text{ is satisfiable.} \end{aligned}$$

For $i = 0, \ldots, 2^N, j = 0, \ldots, 2^N$, boolean variables $x_{i,j,1} \ldots x_{i,j,r} =: \overline{x}_{i,j}$ encode the j-th symbol of the i-th configuration. F_x is a formula in these variables.
On the structure of F_x:

For $i = 1, \ldots, 2^N - 1,\ j = 1, \ldots, 2^N - 1$ the following holds:

- The symbol encoded by $\overline{x}_{i,j}$ only depends on the program of M and the symbols encoded by $\overline{x}_{i-1,j-1}, \overline{x}_{i-1,j}$ and $\overline{x}_{i-1,j+1}$.

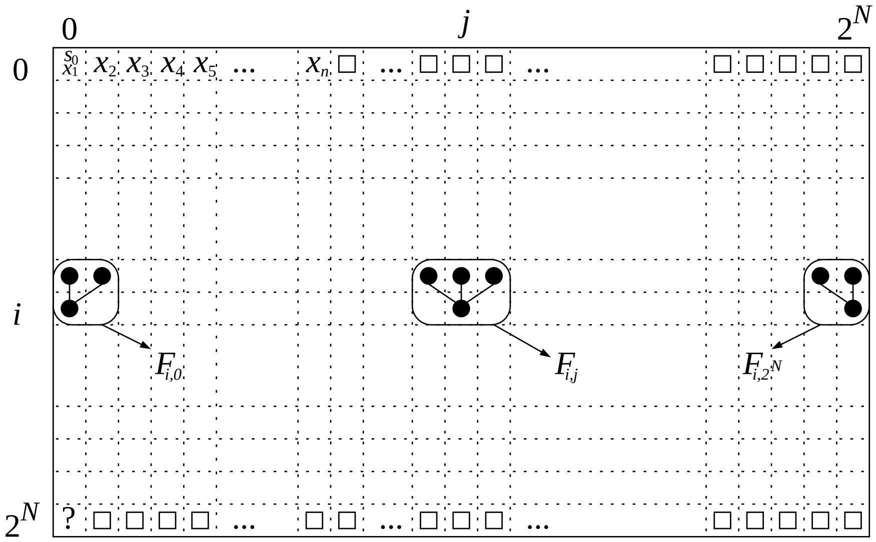

Fig. 1. Computation table $T(M, x, p)$

For a symbol on position (i, j), this is realized by a formula H in CNF such that

$$\underbrace{H(\overline{x}_{i,j}, \overline{x}_{i-1,j-1}, \overline{x}_{i-1,j}, \overline{x}_{i-1,j+1})}_{=:F_{i,j}} \equiv \bigwedge_{l=1}^{R} \underbrace{(x_{i,j,1}^{\alpha_{l,1}} \vee x_{i,j,2}^{\alpha_{l,2}} \vee \ldots \vee x_{i-1,j+1,r}^{\alpha_{l,4r}})}_{=:H_{i,j,l}}.$$

– It holds that $F_{i,j} =_{\text{def}} H(\overline{x}_{i,j}, \overline{x}_{i-1,j-1}, \overline{x}_{i-1,j}, \overline{x}_{i-1,j+1}) = 1 \Leftrightarrow \big($if M is in configuration c after $i-1$ steps and c contains the symbols encoded by $\overline{x}_{i-1,j-1}, \overline{x}_{i-1,j}$ and $\overline{x}_{i-1,j+1}$ on positions $j-1, j, j+1$, then there is a configuration c' which is an immediate successor to configuration c and contains the symbol encoded by $\overline{x}_{i,j}$ on position j.$\big)$.

Observe that $\alpha_{l,k}$ as well as r and R do not depend on i, j or the input x, but only on the program of M. In the following, we assume that R is a power of 2 and that $r \geq 2$ is a divisor of R.
Consequently, for $i, j \in \{1, \ldots, 2^N - 1\}$, formulas $F_{i,j}$ describe valid transitions in a computation table of M on input x. We need special formulas for the borders of a computation table (confer figure 1): The symbol encoded at the first position in each row only depends on two other symbols. The same holds for the symbol encoded at the last position in each row. Nevertheless, these formulas have a similarly regular structure. The formulas for the first row have to ensure that their variables encode the initial configuration. Since we are only interested in accepting computation tables, the formulas for the last row have to ensure that their variables encode the accepting configuration. Observe that for the first row,

only the formulas for positions $1, 2, \dots, |x|$ depend on x, and that the formulas in the last row do not depend on x at all. Let $F_{0,0}, \dots, F_{0,2^N}$ be the formulas for the first row of the computation table, and let $F_{2^N,0}, \dots, F_{2^N,2^N}$ be the formulas for the last row.
Without loss of generality, we assume that all special formulas are in CNF and consist of R clauses with $4r$ boolean variables each.

We now have

$$F_x \equiv \bigwedge_{i,j \in \{0,\dots,2^N\}} F_{i,j}$$

and

$$F_x \text{ is satisfiable} \Leftrightarrow x \in L\ .$$

Since both formulas are logically equivalent, we can assume that

$$\begin{aligned} F_x = \quad & F_{0,0} \wedge F_{0,1} \wedge \dots \wedge F_{0,2^N-1} \wedge F_{0,2^N} \\ & \wedge F_{1,0} \wedge F_{1,1} \wedge \dots \wedge F_{1,2^N-1} \wedge F_{1,2^N} \\ \vdots \\ & \wedge F_{2^N,0} \wedge F_{2^N,1} \wedge \dots \wedge F_{2^N,2^N-1} \wedge F_{2^N,2^N}. \end{aligned}$$

We now take a closer look on the structure of formulas $F_{i,j}$:
For $i \in \{1, \dots, 2^N - 2\}$ and $j \in \{2, \dots, 2^N - 2\}$ literals $x_{i,j,k}$ and $\neg x_{i,j,k}$ occur in formulas $H_{i,j,1}, \dots, H_{i,j,R}$, $H_{i+1,j-1,1}, \dots, H_{i+1,j-1,R}$, $H_{i+1,j,1}, \dots, H_{i+1,j,R}$ and $H_{i+1,j+1,1}, \dots, H_{i+1,j+1,R}$ and nowhere else. Observe that, in which of the clauses of a formula $F_{i,j}$ a literal $x_{i,j,k}$ occurs, only depends on k.

Hence, there exist $\beta_{k1}, \dots, \beta_{kR} \in \{0, 1\}$ for $k \in \{1, \dots, r\}$ such that for all $i \in \{1, \dots, 2^N - 2\}$, $j \in \{2, \dots, 2^N - 2\}$ and all $l \in \{1, \dots, R\}$ it holds that

$$\beta_{kl} = 1 \Leftrightarrow \text{Literal } x_{i,j,k} \text{ occurs in the } l\text{-th clause of } F_{i,j}.$$

In the same way, there exist $\delta_{k1}, \dots, \delta_{k(3R)} \in \{0, 1\}$ for $k \in \{1, \dots, r\}$, such that for all $i \in \{1, \dots, 2^N-2\}$, $j \in \{2, \dots, 2^N-2\}$, $l \in \{1, \dots, R\}$ and all $m \in \{0, 1, 2\}$ it holds that:

$$\delta_{k(l+m\cdot R)} = 1 \Leftrightarrow \text{Literal } x_{i,j,k} \text{ occurs in the } l\text{-th clause of } F_{i+1,j-1+m}.$$

Analogously, we define $\beta'_{k1}, \dots \beta'_{kR} \in \{0, 1\}$ and $\delta'_{k1}, \dots, \delta'_{k(3R)} \in \{0, 1\}$ for the occurrences of literals $\neg x_{i,j,k}$.

Quite similar to the proof of Lemma 1, we construct vectors v_i, v'_i for each boolean variable φ_i in F_x. These vectors describe occurrences of positive and negative literals of φ_i in clauses of F_x such that there is a subset of vectors whose sum has a certain value b if and only if F_x is satisfiable. Recall that F_x contains an exponential number of variables and clauses, thus we have to create exponentially many different vectors and each vector has exponential length. Due to the high-grade regularity of F_x, the vectors we have to construct have a very regular structure as well.

We now show how to construct vectors for the variables in formulas $F_{i,j}$ with $i \in \{1, \ldots, 2^N - 2\}$ and $j \in \{2, \ldots, 2^N - 2\}$:
For a literal $x_{i,j,k}$ we construct a vector $v_{i,j,k}$. Let $r' =_{\text{def}} 4r + 2$. Then, $v_{i,j,k}$ has the r'-nary representation

$$\underbrace{\underbrace{0\ldots0}_{l_1}\beta_{k1}\ldots\beta_{kR}\underbrace{0\ldots0}_{l_2}\delta_{k1}\ldots\delta_{k3R}\underbrace{0\ldots0}_{l_3}}_{\text{for clauses}}\underbrace{\underbrace{0\ldots0}_{l_4}\underbrace{(r'-1)\ldots(r'-1)}_{\frac{R}{r}}\underbrace{0\ldots0}_{l_5}}_{\text{for variables}},$$

with

$l_1 =_{\text{def}} R \cdot i \cdot (2^N + 1) + R \cdot j,$
$l_2 =_{\text{def}} R \cdot (2^N - 1),$
$l_3 =_{\text{def}} R \cdot (2^N - j - 1) + R \cdot (2^N - i - 1)(2^N + 1),$
$l_4 =_{\text{def}} R \cdot i \cdot (2^N + 1) + R \cdot j + (k - 1) \cdot \frac{R}{r}$ and
$l_5 =_{\text{def}} (r - k) \cdot \frac{R}{r} + R \cdot (2^N - j) + R \cdot (2^N - i)(2^N + 1).$

The first $l_1 + R + l_2 + 3R + l_3$ digits mark the clauses in F in which literal $x_{i,j,k}$ occurs. The remaining $l_4 + \frac{R}{r} + l_5$ digits encode the indices of the literal: The position of the $(r' - 1)$-block in the variable part of the vector encodes i, j and k. The vectors are constructed in such a way that $(r' - 1)$-blocks of distinct vectors do not overlap.

Observe that $\underbrace{R + l_2 + 3R + l_3 + l_4 + \frac{R}{r}}_{l_6} = R \cdot (2^{2N} + 2^{N+1} + 1) + k \cdot \frac{R}{r}$.
Obviously, we can ignore the l_1 preceding zeros when construction the vectors. Our construction ensures that the first l_6 digits of the vectors do not depend on i or j but only on k. Let M_k denote these l_6 most significant digits. Thus, l_5—the number of zeroes right to M_k—encodes i and j.
Since M_k only contains a constant number of digits unequal to 0, it is easy to construct circuits $C'_1 \ldots, C'_r$ in logarithmic space such that $I(C'_k) = (M_k)_{r'}$ holds for $k \in \{1, \ldots, r\}$.
It remains to annex zeros in order to complete the variable part. In logarithmic space, it is possible to construct $\{\cup, +, \times\}$-circuits $C_1, \ldots C_r$ such that the following holds for $k \in \{1, \ldots, r\}$:

$$I(C_k) = I(C'_k) \odot r'^{(r-k)\cdot\frac{R}{r}} \odot (r'^R)^{2\cdot(2^N+1)} \odot \left\{(r'^R)^l : l \in D\right\},$$

where

$$D =_{\text{def}} \left\{a + b \cdot (2^N + 1) : a \in \{2, \ldots, 2^N - 2\} \text{ and } b \in \{0, \ldots, 2^N - 3\}\right\}.$$

Consequently, each circuit C_k computes the set of SOS-vectors for literals $x_{i,j,k}$ with $i \in \{1, \ldots, 2^N - 2\}$ and $j \in \{2, \ldots, 2^N - 2\}$. The definition of D ensures that no vectors for variables in formulas at the borders of the computation table are created.

Analogously, we can construct circuits $C_1^{\neg} \dots C_r^{\neg}$ which compute the vectors $v'_{i,j,k}$ for the occurrences of literals $\neg x_{i,j,k}$. The only difference is that we use $\beta'_{k\,i}$ and $\delta'_{k\,i}$ instead of $\beta_{k\,i}$ and $\delta_{k\,i}$ for M'_k.

Hence, the $\{\cup, +, \times\}$-circuit $C_{\text{vectors'}} =_{\text{def}} \bigcup_{i=1}^{r} C_i \cup C_i^{\neg}$ can be constructed in logarithmic space. It computes vectors describing the distribution of positive and negative literals of variables which encode the symbols marked in figure 2.

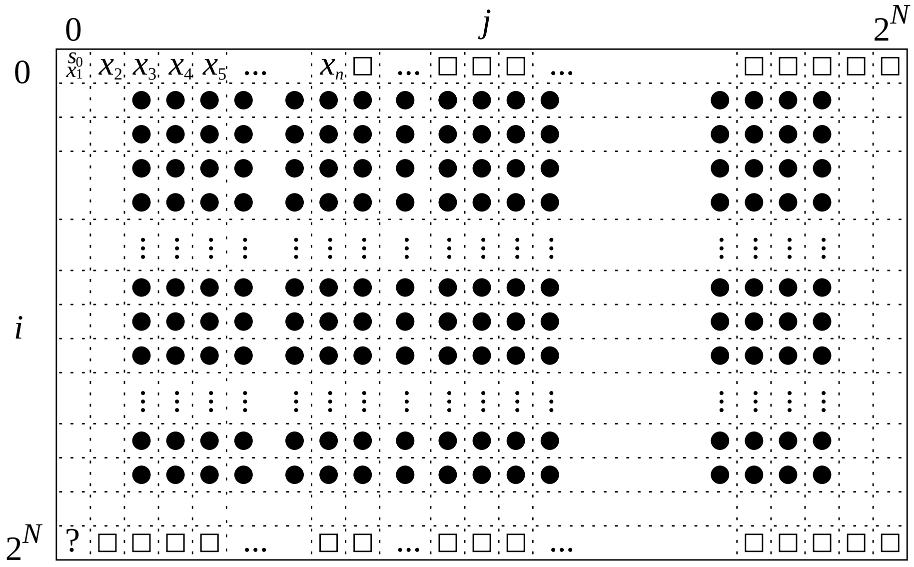

Fig. 2. Computation table $T(M, x, p)$

Similarly, we can construct circuits which compute vectors for the rest of the variables in F: The structure of the formulas for the first n symbols in the computation table and thus also that of the corresponding vectors depends on the input x. For each digit of x, it is easy to construct a circuit which computes the corresponding vectors. The structure of the remaining formulas is independent from the input and highly regular, thus we can create these vectors by using a similar construction as presented above.

Let C_{vectors} be the $\{\cup, +, \times\}$-circuit which computes the the set of all vectors for variables in F. Since there are two vectors for each variable, $I(C_{\text{vectors}})$ computes precisely $2 \cdot r \cdot (2^N + 1)^2$ vectors. Let $V =_{\text{def}} r \cdot (2^N + 1)^2$, hence $V = 2^{2N+\log r} + 2^{N+1+\log r} + 2^{\log r}$. Let $v \in I(C_{\text{vectors}})$. The r'-nary representation of v has (possibly with leading zeros) length $2 \cdot V \cdot \frac{R}{r}$. The first $V \cdot \frac{R}{r}$ digits form the clause part and the remaining $V \cdot \frac{R}{r}$ digits form the variable part. Observe that every two vectors which describe the positive and negative occurrences of literals of the same variable have identical clause parts. This means that the $(r'-1)$-blocks in such two vectors are located on the same positions.

We now construct—as can be seen in figure 3—in logarithmic space a $\{\cup, +, \times\}$-circuit C_{sums} which computes all possible sums of vectors from $I(C_{\text{vectors}})$ which have precisely V addends.

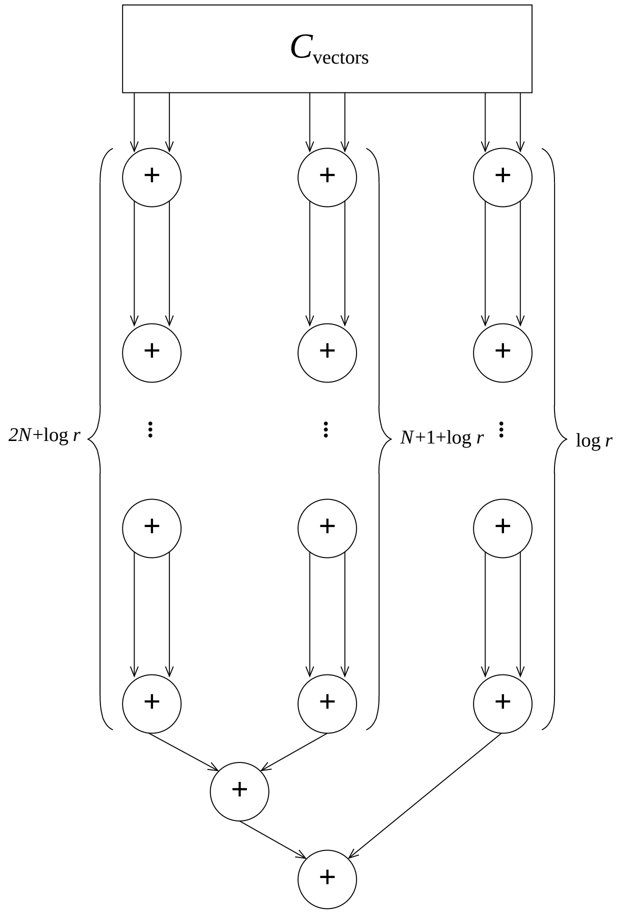

Fig. 3. Circuit C_{sums}

Let $v_1, \ldots, v_V$ and $v'_1, \ldots, v'_V$ be vectors such that, for $i = 1, \ldots, V$, vector v_i describes the positive occurrences and vector v'_i describes the negative occurrences of the i-th variable in F.

Claim: Let $v \in I(C_{\text{sums}})$ with $|v| \geq V \cdot \frac{R}{r}$ and $W =_{\text{def}} |v| - V \cdot \frac{R}{r}$. Then it holds:

$$v = \big(a_1 \dots a_W \underbrace{(r'-1)(r'-1)\dots(r'-1)}_{V\cdot\frac{R}{r}}\big)_{r'} \Leftrightarrow$$
$$\exists_{c_1,\dots,c_V\in\{0,1\}} \textstyle\sum_{i=1}^{V} \big(c_i v_i + (1-c_i)v_i'\big) = v.$$

Proof of the claim: The implication " $\Leftarrow$ " is a direct consequence of the construction of the vectors and thus evident. We obtain the implication " $\Rightarrow$ " from Lemma 5 and the following argument: If none of the last $V \cdot \frac{R}{r}$ digits of the r'-nary representation of v is unequal to $(r'-1)$, there cannot have been overflows when v was computed. Let us assume that this is false. Let then j be the rightmost position in the right part of v, for which more than one vector with value $(r'-1)$ on position j in the right part were added. Therefore, at least $(r'+1)$ of such vectors must have been added to obtain the value $(r'-1)$ on position j of v. By Lemma 5.2, the right part of v must contain at least one digit which is 0. This is a contradiction.

We now know that all vectors in $I(C_{\text{sums}})$, whose right parts only consist of digits $(r'-1)$, were created by a valid choice of vectors from $I(C_{\text{vectors}})$. A valid choice means that for each of the two vectors which belong to the same variable, precisely one was used in the sum. In other words, in the corresponding assignment of truth-values to the variables in F, each variable was assigned exactly one truth-value.

Since each clause of F_x contains exactly $4r$ variables, any assignment of truth-values to variables in F can make at most $4r$ literals true in each clause.

Hence, all vectors in in $I(C_{\text{sums}})$, which were created by a valid choice of vectors from $I(C_{\text{vectors}})$, have the r'-nary representation

$$a_1 a_2 \dots a_{(V\cdot\frac{R}{r})} \underbrace{(r'-1)(r'-1)\dots(r'-1)}_{V\cdot\frac{R}{r}},$$

with $0 \le a_1, a_2, \dots, a_{(V\cdot\frac{R}{r})} \le 4r = r'-2$. From now on, we will call such vectors *valid vectors*.

Defining $K =_{\text{def}} V \cdot \frac{R}{r}$, we obtain

$$\begin{aligned}
x \in L \Leftrightarrow\ & F_x \text{ is satisfiable} \\
& \Leftrightarrow \text{there exists an assignment of variables in } F_x\text{, which makes} \\
& \quad \text{at least 1 literal true in each clause} \\
& \Leftrightarrow \text{there exists a valid vector } v = (a_1 \dots a_K (r'-1)\dots(r'-1))_{r'} \text{ with} \\
& \quad 1 \le a_1, \dots, a_K \le 4r \text{ in } I(C_{\text{sums}})\ .
\end{aligned}$$

We now construct a circuit C_A which computes a set of balancing vectors such that the following holds:

$\Big($There exist $1 \leq a_1, \ldots, a_K \leq 4r = r' - 2$ such that

$$\Big(a_1 a_2 \ldots a_K \underbrace{(r'-1)(r'-1)\ldots(r'-1)}_{K}\Big)_{r'} \in I(C_{\text{sums}})\Big) \Leftrightarrow$$

$$\Big(\underbrace{(r'-1)(r'-1)\ldots(r'-1)}_{K+K}\Big)_{r'} \in I(C_{\text{sums}}) + I(C_A).$$

In logarithmic space we can construct a $\{\cup, +, \times\}$-circuit C'_A with $I(C'_A) = \{1, 2, \ldots, r'-2\} \odot \{(r')^{K+i} : 0 \leq i < K\}$. We use this circuit to construct a circuit C_A which computes all possible sums with precisely K addends from $I(C'_A)$. This can be done by a circuit which is very similar to that in figure 3.

Claim: For all $z \in I(C_A)$ with $z = (a_1 \ldots a_K \underbrace{0\ldots0}_{K})_{r'}$, it holds that

$$\exists_{i \in \{1,\ldots,K\}} a_i = (r'-1) \Rightarrow \exists_{i \in \{1,\ldots,K\}} a_i = (0). \qquad (\star)$$

Proof of the claim: Let $z = \big(a_1 \ldots a_{j-1}(r'-1)a_{j+1} \ldots a_K \underbrace{0\ldots0}_{K}\big)_{r'} \in I(C_A)$ with $0 < a_1, \ldots, a_K \leq r'-1$.
Let $v_1, \ldots, v_K \in I(C'_A)$ be vectors with $\sum_{i=1}^{K} v_i = z$.

The vectors in $I(C'_A)$ have the r'-nary representation $(\overbrace{0\ldots0a0\ldots0}^{K}\overbrace{0\ldots0}^{K})_{r'}$ with $1 \leq a \leq r'-2$. Since we assumed that the leftmost K digits from z are unequal to 0, K vectors must have been used to fill up these positions by Lemma 5.1. By this Lemma we also know that the following holds for the addends: For each of the leftmost K digits from z, there are at most two addends which have a value unequal to 0 on the appropriate position. Hence, there can be no overflows greater than 1.

We now consider the three different possibilities how digit $(r'-1)$ on position j of z was created:

1. There are vectors $u, v \in \{v_1, \ldots, v_K\}$ such that u has digit s on position j and v has digit t on position j with $1 \leq s, t \leq r'-2$ and $s + t = r'-1$.
2. There are vectors $u, v \in \{v_1, \ldots, v_K\}$ such that u has digit s on position j and v has digit t on position j with $1 \leq s, t \leq r'-2$, $s + t = r'-2$ and there is an overflow from the right.
3. There are vectors $u, v \in \{v_1, \ldots, v_K\}$ such that u has digit s on position $j-1$ and v has digit t on position $j-1$ with $1 \leq s, t \geq r'-1$. Hence, an overflow occurs. Furthermore, there is a vector $w \in \{v_1, \ldots, v_K\}$ which has digit $(r'-2)$ on position j.

In cases 1 and 2 $u' =_{\text{def}} \Big(\big(\sum_{i=1}^{K} v_i\big) - u\Big)$ is a vector, whose leftmost K digits are unequal to 0. Nevertheless, u' is the sum of $K-1$ vectors from $I(C'_A)$. This is a contradiction to Lemma 5.1. Similarly, in case 3 it also holds that the leftmost K digits of $u' =_{\text{def}} \Big(\big(\sum_{i=1}^{K} v_i\big) - w\Big)$ are unequal to 0. Again, this contradicts

Lemma 5.1. Hence, there has to be one digit with value 0 among the first K digits of z. This proves our claim.

This property of the balancing vectors ensures that every balancing vector which contains a digit with value $(r'-1)$ in the left part of its r'-nary representation does also contain a digit with value 0. Consequently, such a vector cannot balance all digits of a vector which has a digit with value 0 among the first K digits.

Obviously, for a valid vector $v \in I(C_{\text{sums}}) = \big(a_1 \dots a_K (r'-1)(r'-1) \dots (r'-1)\big)_{r'}$ with $1 \leq a_1, \dots, a_K \leq r'-2$ there exists a balancing vector $a \in I(C_A)$ such that $v + a = \big(\underbrace{(r'-1)(r'-1)\dots(r'-1)}_{2K}\big)_{r'}$.

For the other implicaton, let $w = \big(w_1 \dots w_{2K}\big)_{r'} \in I(C_{\text{sums}}) + I(C_A)$ with $w_i = (r'-1)$ for all $i \leq 2K$. Hence, $w = v + b$ with $b = (b_1 \dots b_K 00 \dots 0)_{r'} \in I(C_A)$ and $v = \big(a_1 \dots a_K (r'-1)(r'-1) \dots (r'-1)\big)_{r'}$ is a valid vector in $I(C_{\text{sums}})$, because the balancing vector b has no influence on the rightmost K digits of w. Hence, $0 \leq a_1, \dots, a_K \leq r'-2$. Let us assume that an overflow occurred while adding v and b. Then there exists a greatest $l \leq K$, such that $a_l + b_l > (r'-1)$. Since this the rightmost overflow, we have $w_l \neq (r'-1)$, which is a contradiction. Consequently, it holds that $a_i + b_i = (r'-1)$ for all $1 \leq i \leq K$. Assume that there exists $i \leq K$ with $a_i = 0$. Then $b_i = (r'-1)$, and due to $(\star)$ there exists $1 \leq j \leq K$ with $b_j = 0$. Since $a_j \leq r'-2$ and $w_j = (r'-1)$, this is a contradiction.

Since all circuits constructed so far can be constructed in logarithmic space, this also holds for the $\{\cup, +, \times\}$-circuit C_{SOS} which is defined by

$$C_{\text{SOS}} =_{\text{def}} \Big(C_{\text{sums}} + C_A\Big) + \Big((-1) \times \big(\underbrace{(r'-1)(r'-1)\dots(r'-1)}_{2K}\big)_{r'}\Big).$$

Altogether, we have

$$\begin{aligned}
x \in L &\Leftrightarrow \text{there exists an accepting computation table } T(M,x,p)\\
&\Leftrightarrow F_x \text{ is satisfiable}\\
&\Leftrightarrow \text{there exists a valid vector } v = \big(a_1 \dots a_K (r'-1) \dots (r'-1)\big)\\
&\ \text{with } 1 \leq a_1, \dots, a_K \leq 4r = r'-2 \text{ in } I(C_{\text{sums}})\\
&\Leftrightarrow \big(\underbrace{(r'-1)(r'-1)\dots(r'-1)}_{2K}\big)_{r'} \in I(C_{\text{sums}}) + I(C_A)\\
&\Leftrightarrow 0 \in I(C_{\text{SOS}})
\end{aligned}$$

This completes the proof. □

Theorem 4. *The problems* $\mathrm{MC}_{\mathbb{Z}}(\cup, +, \times)$ *and* $\mathrm{MC}_{\mathbb{Z}}(\cup, \cap, +, \times)$ *are complete for* NEXPTIME.

Proof. Hardness follows directly from Lemma 6. Let C be a $\{\cup, \cap, +, \times\}$-circuit, and let $b \in \mathbb{Z}$. In exponential time, we can unfold C into a (possibly exponentially larger) formula F. Let p be a polynomial such that $|F| \leq 2^{p(|C|)}$. Observe that for any integer i computed in a gate of F, it holds that $-2^{p(|C|)} \leq i \leq 2^{p(|C|)}$. Hence, we can guess nondeterministically an integer with absolute value $\leq 2^{p(|C|)}$ for every gate of F and then check whether these integers really prove that $b \in I(F) = I(C)$. $\square$

We do not have any decidable upper bound for the unrestricted versions of the membership-problems. As the next example shows, there is evidence suggesting that $\mathrm{MF}_{\mathbb{Z}}(\cup, \cap, ^-, +, \times)$—and thus also $\mathrm{MC}_{\mathbb{Z}}(\cup, \cap, ^-, +, \times)$—might be undecidable:

Example 4. A prime p is called a *Fermat-Prime* if there exists $k > 0$ such that $p = 2^{2^k} + 1$. The only known Fermat-Primes are $2^{2^k} + 1$ for $k = 0, 1, 2, 3, 4$. It is currently known that $2^{2^k} + 1$ is composite for $5 \leq k \leq 32$ [CMP03]. Moreover, it is conjectured that the number of Fermat-Primes is finite with $2^{2^4} + 1 = 65537$ being the largest Fermat-Prime (see, for example, [HW79]). There is a simple $\{\cup, \cap, ^-, +, \times\}$-formula FERMAT having the property that $0 \in I(F)$ iff there is no Fermat-Prime greater than $2^{2^4} + 1$.[1] Hence, a decision procedure for $\mathrm{MF}_{\mathbb{Z}}(\cup, \cap, ^-, +, \times)$ could test whether there exists a sixth Fermat-Prime. This would be surprising.
We construct FERMAT out of several subcircuits. Recall circuits Z, $\mathbb{Z}$-POWER2 and $\mathbb{Z}$-PRIMES from section 2. We now define the circuits $\mathbb{N}$-POWER8 $=_{\text{def}}$ $\mathbb{Z}$-POWER2 $\cap \big((\{7\} \times \mathrm{Z}) + \{1\}\big)$ and $\mathbb{N}$-POWER2 $=_{\text{def}} \{1, 2, 4\} \times$ $\mathbb{N}$-POWER8. Observe that $I(\mathbb{N}\text{-POWER8}) = \{8^k : k \geq 0\}$ and thus $I(\mathbb{N}\text{-POWER2}) = \{2^k : k \geq 0\}$. The circuit $\mathbb{Z}$-PRIMES$'$ $=_{\text{def}}$ $\mathbb{Z}$-PRIMES$\cap\overline{3 \cup 5 \cup 17 \cup 257 \cup 65537}$ computes the set of positive and negative primes minus the set of known Fermat-Primes. All circuits constructed so far can easily be unfolded into formulas.
Finally we define FERMAT as

$$\overline{\big(\mathbb{Z}\text{-PRIMES}' \cap (\mathbb{N}\text{-POWER2} + \{1\})\big) \times \{0\}}$$

and obtain:

$$\text{There is no Fermat-Prime greater than } 2^{2^4}+1 \Leftrightarrow 0 \in I(\text{FERMAT}).$$

The implication "$\Rightarrow$" follows directly from the construction. To see "$\Leftarrow$", note that the following is known from the literature:

Lemma 7. *Let $l \geq 0$. If $2^l + 1$ is prime, then there exists $k \geq 0$ such that $l = 2^k$.*

Proof. Assume that $2^k + 1$ is prime and k contains an odd factor b, that means $2^k + 1 = (2^a)^b + 1$. Then $2^a + 1$ is a factor of $2^k + 1$ because we can write $2^k + 1$ as $(2^a + 1) \cdot (2^{a(b-1)} - 2^{a(b-2)} + 2^{a(b-3)} - \ldots + 1)$. Hence $2^k + 1$ cannot be prime, which leads to a contradiction. $\square$

As a trivial consequence of Lemma 6, we obtain

Theorem 5. *The problem $\mathrm{MC}_{\mathbb{Z}}(\cup, \cap, ^-, +, \times)$ is NEXPTIME-hard.*

[1] In an earlier version, we used a weak variant of Goldbach's Conjecture as evidence. Holger Petersen (Stuttgart) observed that this can be improved to the statement about Fermat-Primes.

4.2 Complementation as the Only Set Operation

In this section, we analyze membership problems for circuits with addition, whose only set operation is complementation. We will see that sets computed by such circuits have a very simple structure.

Lemma 8. *Let $C = (G, E, g_C, \alpha)$ be a $\{^-, +\}$-circuit. For every $g \in G$, there exists an $a_g \in \mathbb{Z}$ such that $I(g) \in \big\{\{a_g\}, \mathbb{Z} \setminus \{a_g\}, \mathbb{Z}, \emptyset\big\}$.*

Proof. Observe that $\big\{\{a\}, \mathbb{Z} \setminus \{a\}, \mathbb{Z}, \emptyset\big\}$ is closed under complementation. For $a, b, c, d \in \mathbb{Z}$ we have $\{a\} + \{b\} = \{a + b\}$, $\big(\mathbb{Z} \setminus \{a\}\big) + \{b\} = \mathbb{Z} \setminus \{a + b\}$, $\big(\mathbb{Z} \setminus \{a\}\big) + \big(\mathbb{Z} \setminus \{b\}\big) = \mathbb{Z} + \big(\mathbb{Z} \setminus \{c\}\big) = \mathbb{Z} + \{d\} = \mathbb{Z}$. Adding $\emptyset$ always results in $\emptyset$. □

We say that $I(g)$ is of type I, II, III or IV if $I(g) = \{a\}$, $\mathbb{Z} \setminus \{a\}$, $\mathbb{Z}$, $\emptyset$, respectively.

Theorem 6. *1. The problem* $\mathrm{MC}_{\mathbb{Z}}(^-, +)$ *is* P*-complete.*
2. The problem $\mathrm{MF}_{\mathbb{Z}}(^-, +)$ *is in* L.

Proof. 1. Containedness: Let $C = (G, E, g_C, \alpha)$ be a $\{^-, +\}$-circuit and $b \in \mathbb{Z}$. Note that, by using the above calculation rule, we can determine in polynomial time what the type of $I(g_C)$ is. The following polynomial time algorithm works on input (C, b) and decides $\mathrm{MC}_{\mathbb{Z}}(^-, +)$. If $I(g_C)$ is of type III or IV we accept or reject, respectively. Otherwise, we eliminate all $^-$-gates from C and obtain a $\{+\}$-circuit C'. Since $\mathrm{MC}_{\mathbb{Z}}(+) \in C_{=}\mathrm{L} \subseteq \mathrm{P}$ (Theorem 7), we can decide $(C', b) \in \mathrm{MC}_{\mathbb{Z}}(+)$ in polynomial time. If $I(g_C)$ is of type I, we accept iff $(C', b) \in \mathrm{MC}_{\mathbb{Z}}(+)$. If $I(g_C)$ is of type II, we accept iff $(C', b) \notin \mathrm{MC}_{\mathbb{Z}}(+)$.
Hardness: We show that the P-complete *monotone boolean circuit value problem* [Go77] can be reduced to $\mathrm{MC}_{\mathbb{Z}}(^-, +)$. To do so, we transform a monotone boolean circuit C into a $\{^-, +\}$-circuit C' of basically the same structure. Every input gate in C with boolean value 0 is replaced by a sub-circuit computing $\mathbb{Z}$, e.g $\overline{0} + \overline{0}$, every input gate in C with boolean value 1 is replaced by a sub-circuit computing $\emptyset$, e.g $\overline{\overline{0} + \overline{0}}$. Every $\vee$-gate in C is replaced by a $+$-gate and every $\wedge$-gate in C with predecessors g_1, g_2 is replaced by a sub-circuit $\overline{\overline{g_1} + \overline{g_2}}$. For every gate g in C' we now have either $I(g) = \mathbb{Z}$ or $I(g) = \emptyset$. Observe that for every $+$-gate g in C' with predecessors g_1, g_2 it now holds that $I(g) = \emptyset \Leftrightarrow \big(I(g_1) = \emptyset \vee I(g_2) = \emptyset\big)$ and $I(g) = \mathbb{Z} \Leftrightarrow \big(I(g_1) = \mathbb{Z} \wedge I(g_2) = \mathbb{Z}\big)$. We obtain ($C$ evaluates to 0) $\Leftrightarrow I(C') = \mathbb{Z} \Leftrightarrow 0 \in I(C') \Leftrightarrow (C', 0) \in \mathrm{MC}_{\mathbb{Z}}(^-, +)$.

2. For a $\{^-, +\}$-formula F, determining the type of $I(F)$ can be done in logarithmic space, and $\mathrm{MC}_{\mathbb{Z}}(+)$ is in L, thus $\mathrm{MF}_{\mathbb{Z}}(^-, +)$ is in L. □

Contrary to these results, we already know that $\{^-, +\}$-circuits over sets of natural numbers can compute very complex sets (recall section 3).

5 Similarities between $\mathbb{Z}$-Case and $\mathbb{N}$-Case

In this section, we present several results where the complexity in the two cases coincides.

Theorem 7. *The problem* $\mathrm{MC}_{\mathbb{Z}}(+)$ *is complete for* $\mathrm{C}_{=}\mathrm{L}$.

Proof. By Lemma 3, it is sufficient to prove that $\mathrm{MC}_{\mathbb{Z}}(+)$ is in $\mathrm{C}_{=}\mathrm{L}$. Let $C = (G, E, g_C, \alpha)$ be a $\{+\}$-circuit and $b \in \mathbb{Z}$. Defining $C' =_{\mathrm{def}} C + (-b)$, we obtain $b \in I(C) \Leftrightarrow 0 \in I(C')$. We then further modify C':

- Add a new input gate g_- with label -1.
- Add a new input gate g_+ with label 1.
- Replace each input gate $g \in G$ with $\alpha(g) < 0$ with a $\{+\}$-circuit which computes $\alpha(g)$ solely from the input gate g_-.
- Replace each input gate $g \in G$ with $\alpha(g) \geq 0$ with a $\{+\}$-circuit which computes $\alpha(g)$ solely from the input gate g_+.

Let C'' be the circuit resulting from the above modifications. It clearly holds that $I(C) - b = I(C') = I(C'')$. We define $f_-(C'')$ as the number of paths in C'' from g_- to the output gate of C'' and $f_+(C'')$ as the number of paths in C'' from g_+ to the output gate of C''. It is easy to see that f_- and f_+ are #L-functions. We now have $(C, b) \in \mathrm{MC}_{\mathbb{Z}}(+) \Leftrightarrow I(C) = b \Leftrightarrow I(C'') = 0 \Leftrightarrow f_-(C'') = f_+(C'')$. Consequently, $\mathrm{MC}_{\mathbb{Z}}(+)$ is in $\mathrm{C}_{=}\mathrm{L}$. □

In contrast to $\{\times\}$-circuits, evaluating $\{+\}$-circuits is not harder in the $\mathbb{Z}$-case than it is in the $\mathbb{N}$-case.

Theorem 8. *The problems* $\mathrm{MF}_{\mathbb{Z}}(\cup, \cap, {}^-, +)$ *and* $\mathrm{MC}_{\mathbb{Z}}(\cup, \cap, {}^-, +)$ *are* PSPACE-*complete.*

Proof Sketch: Containedness can be shown by a simple modification of the proofs for the corresponding problems in the $\mathbb{N}$-case (cf. [MW03]). For PSPACE-hardness, observe that the PSPACE-complete problem QSOS can be reduced to $\mathrm{MF}_{\mathbb{Z}}(\cup, \cap, {}^-, +)$ by adapting the proof of Lemma 2: For $i = 1, \ldots, n$, define formulas A_i as $0 \cup a_i$. The rest of the proof is analog. □

Theorem 9. *The problems* $\mathrm{MF}_{\mathbb{Z}}(\cup, \cap, {}^-, \times)$ *and* $\mathrm{MC}_{\mathbb{Z}}(\cup, \cap, {}^-, \times)$ *are* PSPACE-*complete.*

Proof Sketch: With respect to containedness, the same as above holds. We here sketch a reduction of the PSPACE-complete problem $\mathrm{MF}_{\mathbb{N}}(\cup, \cap, {}^-, \times)$ [MW03] to $\mathrm{MF}_{\mathbb{Z}}(\cup, \cap, {}^-, \times)$. Let F be a $\{\cup, \cap, {}^-, \times\}$-formula over sets of natural numbers. A $\{\cup, \cap, {}^-, \times\}$-formula F' is obtained by replacing every input gate g of F with a formula $\big(\alpha(g) \cup (\alpha(g) \times \{-1\})\big)$. Arguing by induction on the number of gates in F, we obtain $I(F') = I_{\mathbb{N}}(F) \cup (I_{\mathbb{N}}(F) \times \{-1\})$. Therefore, it now holds for all $b \in \mathbb{N}$ that $b \in I_{\mathbb{N}}(F) \Leftrightarrow b \in I(F')$. This yields the desired reduction. □

For the complexity of the remaining membership problems in our analysis, we also have the same upper and lower bounds as McKenzie and Wagner have for the corresponding problems in the $\mathbb{N}$-case. The proofs for the $\mathbb{Z}$-case bounds only require straightforward modifications applied to those in the $\mathbb{N}$-case (see [MW03]). The results for these membership problems are also included in the table on the next page, but lack a reference to a theorem or lemma in this paper.

6 Conclusion and Open Problems

Table 1 summaries our results.

$\mathcal{O}$	$\mathrm{MC}_{\mathbb{Z}}(\mathcal{O})$ lower bound	$\mathrm{MC}_{\mathbb{Z}}(\mathcal{O})$ upper bound	Th. L.	$\mathrm{MF}_{\mathbb{Z}}(\mathcal{O})$ lower bound	$\mathrm{MF}_{\mathbb{Z}}(\mathcal{O})$ upper bound	Th. L.
$\cup,\ \cap,\ ^-,\ +,\ \times$	NEXPTIME	?	T5	PSPACE	?	
$\cup,\ \cap,\ +,\ \times$	NEXPTIME	NEXPTIME	T4	NP	NP	
$\cup,\ +,\ \times$	NEXPTIME	NEXPTIME	T4	NP	NP	
$\cap,\ +,\ \times$	P	coNP		L	DLOGCFL	
$+,\ \times$	P	coNP	L4	L	DLOGCFL	
$\cup,\ \cap,\ ^-,\ +$	PSPACE	PSPACE	T8	PSPACE	PSPACE	T8
$\cup,\ \cap,\ +$	PSPACE	PSPACE		NP	NP	
$\cup,\ +$	NP	NP		NP	NP	
$^-,\ +$	P	P	T6	L	L	T6
$\cap,\ +$	$\mathrm{C_{=}L}$	$\mathrm{C_{=}L}$		L	L	
$+$	$\mathrm{C_{=}L}$	$\mathrm{C_{=}L}$	T7	L	L	
$\cup,\ \cap,\ ^-,\ \times$	PSPACE	PSPACE	T9	PSPACE	PSPACE	T9
$\cup,\ \cap,\ \times$	PSPACE	PSPACE		NP	NP	
$\cup,\ \times$	NP	NP		NP	NP	
$\cap,\ \times$	$\mathrm{C_{=}L} \wedge \oplus\mathrm{L}$	P	T3	L	L	
$\times$	$\mathrm{NL} \wedge \oplus\mathrm{L}$	$\mathrm{NL} \wedge \oplus\mathrm{L}$	T2	L	L	
$\cup,\ \cap,\ ^-$	P	P	L3	L	L	L3
$\cup,\ \cap$	P	P	L3	L	L	L3
$\cup$	NL	NL	L3	L	L	L3
$\cap$	NL	NL	L3	L	L	L3

Table 1. Membership problems for circuits over subsets of $\mathbb{Z}$

Several open problems are apparent from table 1. If one could prove $\mathrm{MC}_{\mathbb{Z}}(+,\times) \in \mathrm{P}$, then it would follow that $\mathrm{MC}_{\mathbb{Z}}(+,\times)$, $\mathrm{MC}_{\mathbb{Z}}(\cap,+,\times)$, and $\mathrm{MC}_{\mathbb{N}}(\cap,+,\times)$ are all P-complete (cf. [MW03]).
Another open problem is finding a decidable upper bound for the complexity of $\mathrm{MC}_{\mathbb{Z}}(\cup,\cap,^-,+,\times)$, or proving it undecidable. As the corresponding problem in the $\mathbb{N}$-case shares that fate, we are interested in how the complexities of $\mathrm{MC}_{\mathbb{Z}}(\cup,\cap,^-,+,\times)$ and $\mathrm{MC}_{\mathbb{N}}(\cup,\cap,^-,+,\times)$ compare.
Related to this, there is a further open question: Is there a circuit over sets of integers computing precisely the set $\mathbb{N}$?
If such a circuit exists, $\mathrm{MC}_{\mathbb{N}}(\cup,\cap,^-,+,\times) \leq_m^{\log} \mathrm{MC}_{\mathbb{Z}}(\cup,\cap,^-,+,\times)$ would immediately follow. Holger Petersen (private communication) was the first to observe that there exists a circuit C having the property that $I(C) \cap \mathbb{N}$ is infinite while $I(C) \cap -\mathbb{N}$ is finite (see section 4.1).

Acknowledgments. The author is grateful to Klaus W. Wagner, Christian Glaßer, Daniel Meister, Bernhard Schwarz (Würzburg) and Holger Petersen (Stuttgart) for very useful discussions and important hints.

References

[All97] E. Allender, Making computation count: Arithmetic Circuits in the Nineties, in the Complexity Theory Column, *SIGACT NEWS 28 (4),* pp. 2–15, 1997.

[BLT92] J. L. Balcázar, A. Lozano, and J. Torán, The complexity of algorithmic problems in succinct instances, *Computer Science,* Plenum, New York, 1992.

[CMP03] R. E. Crandall, E. W. Mayer, and J. S. Papadopoulos, The Twenty-Fourth Fermat Number is Composite, *Math. Comput. 72,* 1555-1572, 2003.

[Go77] L. M. Goldschlager, The monotone and planar circuit value problems are logspace complete for P, *SIGACT NEWS 9,* pp. 25–29, 1977.

[HW79] G. H. Hardy and E. M. Wright, An introduction to the theory of numbers, *Oxford University Press*, 1979.

[Im88] N. Immerman, Nondeterministic space is closed under complementation, *SIAM Journal on Computing 17,* pp. 935–938, 1988.

[MW03] P. McKenzie and K. W. Wagner. The Complexity of Membership Problems for Circuits over Sets of Natural Numbers, *Lecture Notes in Computer Science 2607,* pp. 571–582, 2003.

[Pa94] C. H. Papadimitriou. Computational Complexity, *Addison-Wesley,* 1994.

[Pe04] H. Petersen, Bemerkungen zu ganzzahligen Ausdrücken, *Private Communication,* 2004.

[Sa73] W. J. Savitch, Maze recognizing automata and nondeterministic tape complexity, *Journal of Computer and System Sciences 7,* pp. 389–403, 1973.

[SM73] L. J. Stockmeyer, A. R. Meyer, Word Problems Requiring Exponential Time, *Proceedings of the 5th ACM Symposium on the Theory of Computing,* pp. 1–9, 1973.

[Sz88] R. Szelepcsényi, The method of forced enumeration for nondeterministic automata, *AI 26,* pp. 279–284, 1984.

[Wa84] K. W. Wagner, The complexity of problems concerning graphs with regularities, *Proceedings 11th Mathematical Foundations of Computer Science, Lecture Notes in Computer Science 176,* pp. 544–552, 1984. Full version as TR N/84/52, Friedrich-Schiller-Universität Jena, 1984.

[Ya00] K. Yang, Integer circuit evaluation is PSPACE-complete, *Proceedings 15th Conference on Computational Complexity,* pp. 204–211, 2000.