Membership problems for recurrent systems over the power set of the natural numbers

Daniel Meister

Report No. 336 Juni 2004

Preprint-Reihe
Fakultät für Mathematik und Informatik
Universität Würzburg

Membership problems for recurrent systems over the power set of the natural numbers

Daniel Meister
Theoretische Informatik
Bayerische Julius-Maximilians-Universität Würzburg
97074 Würzburg, Germany
meister@informatik.uni-wuerzburg.de

Abstract. A finite recurrent system over the power set of the natural numbers of dimension n is a pair composed of n n-ary functions over the power set of the natural numbers and an n-tuple of singleton sets of natural numbers. Every function is applied to the components of the tuple and computes a set of natural numbers, that might also be empty. The results are composed into another tuple, and the process is started anew. Thus, a finite recurrent system defines an infinite sequence of n-tuples containing sets of natural numbers. The last component of a generated n-tuple is called the output of one step, and the union of all outputs is the set defined by the finite recurrent system. We study the membership problem for special finite recurrent systems, whose functions are built from the set operations union, intersection, complementation and the arithmetic operations addition and multiplication. Sum and product of two sets of natural numbers are defined elementwise. We will show undecidability results as well as completeness results for complexity classes like NP and PSPACE.

1. Introduction. Sets of natural numbers can be represented by a variety of mathematical objects. Finite sets or co-finite sets, the complements of finite sets, can be represented by words over $\{0,1\}$, i.e., by natural numbers, with a canonical interpretation. However, large sets require large numbers in this model. If these sets possess regularities a more efficient representation would be desirable. In case of sets that are neither finite nor co-finite such a simple representation does not work at all. Stockmeyer and Meyer defined integer expressions, which are expressions built from naturals, the set operations union, intersection and complementation (relative to the set of naturals) and an addition operation [10]. Wagner studied an hierarchical model of a similar flavour that can nowadays be understood as an arithmetic circuit [14], [15]. Such concise representations however make it difficult to derive information about the set from its representation. The *membership problem* for natural numbers in general can be understood as the problem, given a set M of natural numbers represented in a certain model and a number b, to decide whether b belongs to set M. The complexity of the membership problem heavily depends on the representation and can generally be described by the formula: the more concise the representation the more complex the membership problem.

McKenzie and Wagner recently studied a large number of membership problems [6]. Given an arithmetic circuit over the power set of the natural numbers involving the standard set operations union, intersection, complementation and the arithmetic operations addition and multiplication (both operations are defined on sets, and sum and product of two sets are defined elementwise) and a natural number b, does the circuit represent a set that contains b? It was shown that restricting the set of possible operations as well as restricting circuits to formulas cover a wide range of complexity classes. Here, a formula is an arithmetic circuit where every vertex has at most one successor. Their work extends past works by Stockmeyer and Meyer [10], Wagner [14] and Yang [16].

The standard approach to circuits is via functions, and circuits represent these functions efficiently. In this sense, all problems above concern such circuits but applied only to fixed inputs. Circuits of various types have been studied deeply, and they are an interesting model to obtain lower bounds complexity results. A lot of information on this subject can be found in the book by Vollmer [13]. In this paper we combine ideas that have been sketched above to obtain set representations by special recurrent system.

Recurrences are well-known. The sequence $1, 1, 2, 3, 5, 8, \ldots$ of numbers—the Fibonacci numbers—is generated by the simple formula $F(n+2) =_{\text{def}} F(n+1) + F(n)$ where $F(0) =_{\text{def}} F(1) =_{\text{def}} 1$. Numerical simulations of one particle or multi-particle systems in physics use systems of recurrences instead of differential equations. Recurrences play an important role in mathematics, computer and other sciences. Though recurrences normally involve only basic arithmetic operations such as addition and multiplication over the natural or the real numbers, operations do not have to be limited to this small collection. A recurrent system over the power set of the natural numbers of dimension n is a pair consisting of a set of n n-ary functions $f_1, \ldots, f_n$ over the power set of the natural numbers and an n-tuple of naturals. Starting from singleton sets defined by the n-tuple the result of function f_i in one step is used as the i-th input in the next step (the precise definition is provided in Section 3). So, a recurrent system generates iteratively an infinite sequence of tuples of sets of natural numbers. The last component of each tuple is the output of the system in the corresponding evaluation step. Then, the union of all outputs defines a set that may be finite or infinite. The existential membership problem M_{ex} for recurrent systems then asks whether there is an evaluation step such that the corresponding output contains a given number, and the exact membership problem M_{tm} asks whether a given number is contained in the result of a specified evaluation step. We require that functions can be represented by arithmetic circuits.

A recurrent system in the sense defined here can be regarded equal to a number of other models. For instance, each vertex of a sink-free directed graph is attached a function whose arity corresponds to the number of incoming arcs, and in each step the value of a vertex is the result of the function applied to the values of the last step of its preceding vertices. Each vertex then has a kind of memory. We can also require that only some vertices are memory vertices, but then there must be further restrictions on the structure of the underlying graph. The idea of memory elements can be found in the field of Boolean circuits describing processes.

We examine membership problems for recurrent systems for a restricted set of functions over the power set of the natural numbers. Equal to the work of McKenzie and Wagner, our functions are built from the three known set operations and addition and multiplication. The general problems in this restricted sense are denoted by $\mathrm{M}_{ex}(\cup, \cap, ^-, \oplus, \otimes)$ and $\mathrm{M}_{tm}(\cup, \cap, ^-, \oplus, \otimes)$. Reducing the set of allowed operations leads to problems like $\mathrm{M}_{ex}(\cup, \oplus)$, where functions are built only from $\cup$ and $\oplus$, $\mathrm{M}_{tm}(\oplus, \otimes)$ or $\mathrm{M}_{tm}(\cup)$. We study the complexity of such membership problems with respect to the set of allowed operations. We will see that such problems are complete for a number of well-known complexity classes such as NP, PSPACE and RE. However, undecidability of some of our problems does not answer an open question from [6]. The authors expected that the general exact membership problem $\mathrm{M}_{tm}(\cup, \cap, {}^{-}, \oplus, \otimes)$ was undecidable. Some evidence for undecidability was given by showing that a decision algorithm would prove or disprove Goldbach's conjecture about sums of primes.

The paper is composed as follows. In Section 3, finite recurrent systems are introduced. Section 4 presents the undecidability proofs of $\mathrm{M}_{ex}(\cup, \cap, \oplus, \otimes)$ and $\mathrm{M}_{ex}({}^{-}, \oplus, \otimes)$. This proof is achieved by reducing DIOPHANTINE, the problem about the satisfiability of Diophantine equations, which is known to be undecidable [5], to both problems. The following sections classify a range of membership problems for recurrent systems. For example, $\mathrm{M}_{ex}(\cup)$ is NL-complete (Section 5), $\mathrm{M}_{ex}(\cap)$ and $\mathrm{M}_{ex}(\cap, \oplus)$ are NP-complete (Section 7), and $\mathrm{M}_{ex}(\cup, \cap)$, $\mathrm{M}_{ex}(\cup, \oplus)$ and $\mathrm{M}_{ex}(\cup, \otimes)$ are PSPACE-complete (Section 8).

2. Preliminaries.

We fix the alphabet $\Sigma =_{\mathrm{def}} \{0, 1\}$. The set of all words over Σ is denoted by Σ^*. All inputs are assumed to be given as words over Σ. By L, NL, P, NP and PSPACE we denote the known complexity classes; for an exact definition and further information we refer to the book by Papadimitriou in that subject [8]. If the computation mode is not mentioned we mean deterministic computations. Nondeterminism is always indicated. The class FL contains all functions that can be computed deterministically by a Turing machine with output tape using logarithmic working space. By $\mathrm{RE} = \Sigma_1$, Δ_2, Σ_2 and Π_2 we denote classes of the arithmetical hierarchy, where Σ_1 is the set of recursively enumerable sets and $\Delta_2 =_{\mathrm{def}} \Sigma_2 \cap \Pi_2$ [4], [7]. A set A is *log-space reducible* to some set B, $A \le_m^{\mathrm{L}} B$, if there is $f \in \mathrm{FL}$ such that, for all $x \in \Sigma^*$, $x \in A \leftrightarrow f(x) \in B$. Since this is the only reducibility that we employ, we will shortly say that A *reduces* to B. For some complexity class $\mathcal{C}$, set A is $\le_m^{\mathrm{L}}$*-complete* for $\mathcal{C}$, if $A \in \mathcal{C}$ and $B \le_m^{\mathrm{L}} A$ for all sets $B \in \mathcal{C}$. We will shortly say that A is $\mathcal{C}$-complete.

Numbers. The set of the natural numbers is denoted by $\mathbf{N}$ and certainly contains 0. If we talk of numbers, we always mean natural numbers. Numbers are represented in binary form. The power set of $\mathbf{N}$ is the set of all subsets of $\mathbf{N}$. For natural numbers a, b, $a \le b$, $[a, b] =_{\mathrm{def}} \{a, a{+}1, \ldots, b\}$. For every number $n \ge 2$, there are n prime numbers smaller than n^2. Two numbers are *relatively prime*, if their greatest common devisor is 1.

Theorem 1. (Chinese Remainder Theorem)
Let $b_1, \ldots, b_k$ be pairwise relatively prime numbers, and let $n_1, n_2 \in \mathbf{N}$. Let $b =_{\mathrm{def}} b_1 \cdot \ldots \cdot b_k$. Then, $n_1 \equiv n_2 \pmod{b}$ if and only if $n_1 \equiv n_2 \pmod{b_i}$ for every $i \in [1, k]$.

For set A and two binary operations $\diamond$ and $\circ$ over A, the triple $(A, \diamond, \circ)$ is a *semiring* if $(A, \diamond)$ and $(A, \circ)$ are commutative monoids and the two distributive laws hold. For $+$ and $\cdot$ denoting addition and multiplication over $\mathbf{N}$, $(\mathbf{N}, +, \cdot)$ is a semiring. By $\mathrm{SR}(b)$ we denote the semiring $([0, b{+}1], \mathrm{sum}_b, \mathrm{prod}_b)$ where the binary operations sum_b and prod_b are defined as follows. Let $a_1, a_2 \in \mathbf{N}$.

$$\mathrm{sum}_b(a_1, a_2) =_{\mathrm{def}} \begin{cases} a_1 + a_2 & \text{, if } a_1 + a_2 \le b \\ b{+}1 & \text{otherwise} \end{cases}$$

Similarly for prod_b. We can define matrices over $\mathrm{SR}(b)$, and matrix multiplication is well-defined. We define relation $\simeq$. For $k \ge 1$ and $a_1, \ldots, a_k, a'_1, \ldots, a'_k \in \mathbf{N}$ let $(a_1, \ldots, a_k) \simeq (\{a'_1\}, \ldots, \{a'_k\})$ if and only if $a_i = a'_i$ for all $i \in [1, k]$. The reader may assume symmetry.

Graphs. A simple, finite, directed graph is a pair $G = (V, A)$ where V is a finite set and $A \subseteq V \times V$. For two vertices $u, v \in V$ there is a u, v*-path* in G, if there is a sequence $(x_1, \ldots, x_k)$ such that $x_1 = u$, $x_k = v$ and $(x_i, x_{i+1}) \in A$ for all $i \in [1, k{-}1]$. The *graph accessibility problem for directed graphs*, denoted by diGAP, is the set of all triples (G, u, v) where G is a

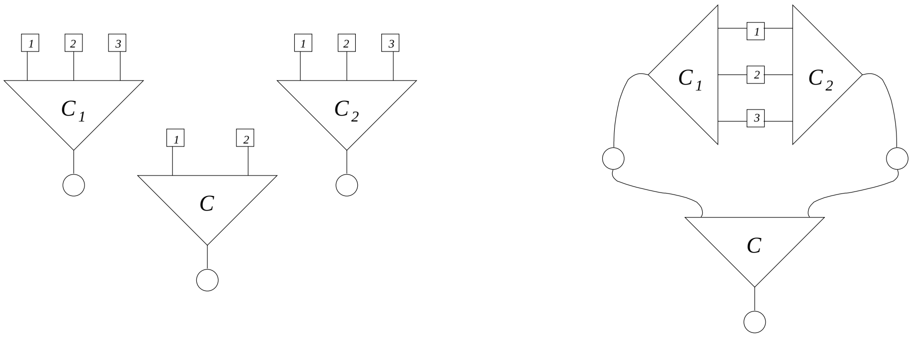

Figure 1. Circuit representation of the superposition of functions.

directed graph, u and v are vertices of G and there is a u,v-path in G. The problem diGAP is NL-complete [9]. G is *acyclic*, if there is no sequence $P = (x_0, \ldots, x_n)$ for n the number of vertices of G such that P is an x_0, x_n-path in G. The problem ACYC is the set of all directed acyclic graphs. Since diGAP restricted even to acyclic graphs is NL-complete, ACYC is NL-complete. For vertices u and v of G, u is a *predecessor* of v, if $(u,v) \in A$. A vertex that is not the predecessor of any vertex is a *sink*.

Circuits. Let $\mathcal{O}$ be a set of commutative operations over set M. $C = (G, g_c, \alpha)$ is an n-ary *arithmetic $\mathcal{O}$-circuit* over M for $n \geq 0$, if $G = (V, A)$ is a (simple, finite) acyclic graph, $g_C \in V$ is a specified vertex of G, the *output vertex*, and $\alpha : V \to \mathcal{O} \cup [1, n]$ such that α establishes a 1-1 correspondence between n vertices without in-coming arcs and $[1, n]$ and all other vertices are assigned operations from $\mathcal{O}$ whose arity correspond with the number of in-coming arcs of the vertices. Vertices assigned a number are called the *input vertices* of C. The arithmetic $\mathcal{O}$-circuit C over M represents a function f_C over M in the following way. Let $(a_1, \ldots, a_n) \in M^n$. The value of the input vertex assigned number i is a_i, the value of vertex u where u is assigned an operation from $\mathcal{O}$ is the result of $\alpha(u)$ applied to the values of the predecessors of u. Then, $f_C(a_1, \ldots, a_n)$ is the value of the output vertex g_C. Let f_C be an n-ary function represented by circuit C, and let $f_{C_1}, \ldots, f_{C_n}$ be n'-ary functions represented by circuits $C_1, \ldots, C_n$. A circuit representation of function $f(x_1, \ldots, x_{n'}) =_{\text{def}} f_C(f_{C_1}(x_1, \ldots, x_{n'}), \ldots, f_{C_n}(x_1, \ldots, x_{n'}))$ is obtained from $C, C_1, \ldots, C_n$ by indentifying the input vertices of $C_1, \ldots, C_n$ assigned the same numbers and identifying the vertices of C assigned numbers with the output vertex of the corresponding circuit C_i. Figure 1 gives a schematic drawing. Input vertices are represented by squares containing the assigned numbers. The example represents $f(\mathbf{x}) =_{\text{def}} f_C(f_{C_1}(\mathbf{x}), f_{C_2}(\mathbf{x}))$ for $\mathbf{x} =_{\text{def}} (x_1, x_2, x_3)$.

3. Recurrent systems. A recurrence is a pair composed of a function and a sequence of elements. From recurrences one can generate infinite sequences by applying the function to certain of already generated elements. Usual recurrences are defined over the set of the real or complex numbers and involve only basic arithmetical operations like addition and multiplication. We extend this notion to recurrent systems over power sets of numbers.

Definition 1. *Let $n \geq 1$. A **finite recurrent system** over the power set of the natural numbers of dimension n is a pair $S = (\mathcal{F}, A)$ where $\mathcal{F} =_{\text{def}} \langle f_1, \ldots, f_n \rangle$ for $f_1, \ldots, f_n$ n-ary functions over the power set of $\mathbf{N}$ and $A \in \mathbf{N}^n$. The dimension of S is denoted by* $\dim S$.

Let $S = (\mathcal{F}, A)$ be a finite recurrent system over the power set of the natural numbers where $\mathcal{F} =_{\mathrm{def}} \langle f_1, \ldots, f_n \rangle$ and $A = (a_1, \ldots, a_n)$. We define for every $t \in \mathbf{N}$:

$$f_i(0) =_{\mathrm{def}} \{a_i\} \text{ and } f_i(t+1) =_{\mathrm{def}} f_i(f_1(t), \ldots, f_n(t)), \ \ i \in [1, n]$$
$$\mathcal{F}(t) =_{\mathrm{def}} (f_1(t), \ldots, f_n(t)) \, .$$

Let $S(t) =_{\mathrm{def}} f_n(t)$. Note that $f_i(t)$ is rather a symbolic denotation of the result of f_i in the t-th evaluation step. We can say that a finite recurrent system over the power set of the natural numbers defines or represents an infinite sequence of sets of natural numbers. By $[S]$ we denote the union of these sets, i.e., $[S] =_{\mathrm{def}} \bigcup_{t \geq 0} S(t)$. We are interested in two problems that arise immediately from our definitions. One can ask whether a number b is generated in step t or whether b is generated in some step at all.

Several authors studied membership problems of sets of natural numbers that can be built from singleton sets of natural numbers by applying the set operations union, intersection, complementation and the two arithmetic set operations addition and multiplication, denoted by $\oplus$ and $\otimes$ [10], [14], [16], [6]. Addition and multiplication on sets are defined elementwise. Let $A, B \subseteq \mathbf{N}$. Then, $A \oplus B =_{\mathrm{def}} \{r + s : r \in A \text{ and } s \in B\}$ and $A \otimes B =_{\mathrm{def}} \{r \cdot s : r \in A \text{ and } s \in B\}$. Let $\mathcal{O} \subseteq \{\cup, \cap, ^-, \oplus, \otimes\}$. An n-ary $\mathcal{O}$-function $f = f(x_1, \ldots, x_n)$ is a function over the variables $x_1, \ldots, x_n$ defined by using only operations from $\mathcal{O}$. In this paper we will consider only finite recurrent systems on $\{\cup, \cap, ^-, \oplus, \otimes\}$-functions.

Definition 2. *Let $\mathcal{O} \subseteq \{\cup, \cap, ^-, \oplus, \otimes\}$. A **finite recurrent $\mathcal{O}$-system** $S = (\mathcal{F}, A)$ over the power set of the natural numbers is a finite recurrent system over the power set of the natural numbers where every function in $\mathcal{F}$ is an $\mathcal{O}$-function.*

Since we will only deal with finite recurrent $\{\cup, \cap, ^-, \oplus, \otimes\}$-systems over the power set of the natural numbers we will henceforth call them *recurrent systems* for short. (In Section 4, *recurrent system* may also mean another type of a finite recurrent system, but this will be restricted to that section and always be clear.) Every recurrent system S defines a possibly infinite set $[S]$ of natural numbers. The *existential membership problem* M_{ex} for recurrent systems asks whether a given number is contained in the defined set, and the *exact membership problem* M_{tm} asks whether a given number is contained in the result of a specified evaluation step. We want to study the complexities of these membership problems with respect to the involved functions. Let $\mathcal{O} \subseteq \{\cup, \cap, ^-, \oplus, \otimes\}$.

$$\mathrm{M}_{ex}(\mathcal{O}) =_{\mathrm{def}} \{(S, b) : S \text{ a recurrent } \mathcal{O}\text{-system and } b \in [S]\}$$
$$\mathrm{M}_{tm}(\mathcal{O}) =_{\mathrm{def}} \{(S, t, b) : S \text{ a recurrent } \mathcal{O}\text{-system and } b \in S(t)\}$$

Instead of writing $\mathrm{M}_{ex}(\{\cup, \cap, \oplus\})$ we will write $\mathrm{M}_{ex}(\cup, \cap, \oplus)$ for short; similarly for the other problems. The complexities of our problems strongly depend on the input representation. We assume that natural numbers are given in binary form and functions are represented by arithmetic circuits with appropriate labels. For circuits we require any encoding that permits adjacency tests of two vertices and detection of labels of vertices in deterministic logarithmic space. (A label list and neighbourhood representation by adjacency lists can be assumed.) So, the size of the representation of a circuit is of order the number of its vertices and edges. It can be verified in nondeterministic logarithmic space whether an input represents an $\mathcal{O}$-function for $\mathcal{O} \subseteq \{\cup, \cap, ^-, \oplus, \otimes\}$. Using our notations McKenzie and Wagner studied the complexity of the question for given recurrent $\mathcal{O}$-system S and number $b \geq 0$ whether $(S, 1, b) \in \mathrm{M}_{tm}(\mathcal{O})$ [6]. Their input representation additionally required a topological ordering of the vertices of the circuits, but this is only of importance for problems that are contained in NL. We will denote the problems investigated by McKenzie and Wagner by MC($\mathcal{O}$). It follows for every $\mathcal{O} \subseteq \{\cup, \cap, ^-, \oplus, \otimes\}$ that $\mathrm{M}_{tm}(\mathcal{O})$ is decidable if and only if MC($\mathcal{O}$) is decidable. The only problems that have not yet been proved decidable are $\mathrm{MC}(\cup, \cap, ^-, \oplus, \otimes)$ and $\mathrm{MC}(^-, \oplus, \otimes)$, and they are not believed to be so (see also [6]).

The results that we will present in the following sections can be classified as decidability and undecidability results. We will show that in case of decidability of $\mathrm{M}_{ex}(\mathcal{O})$ for some $\mathcal{O} \subseteq \{\cup, \cap, ^-, \oplus, \otimes\}$ there is an easily computable function f such that $(S, b) \in \mathrm{M}_{ex}(\mathcal{O})$ for S a recurrent $\mathcal{O}$-system and $b \in \mathbf{N}$ if and only if $(S, t, b) \in \mathrm{M}_{tm}(\mathcal{O})$ for some $t \leq f(S, b)$. We will show $\mathrm{M}_{ex}(\cup, \cap, \oplus, \otimes)$ and $\mathrm{M}_{ex}(^-, \oplus, \otimes)$ to be undecidable. For both problems there is no computable function that bounds number t. Otherwise, $\mathrm{M}_{ex}(\cup, \cap, \oplus, \otimes)$ would be decidable due to the decidability of $\mathrm{M}_{tm}(\cup, \cap, \oplus, \otimes)$. It is still an open question whether $\mathrm{M}_{tm}(^-, \oplus, \otimes)$ is decidable. The following statements are easy observations that link decidability of membership problems for recurrent systems and circuits.

Fact 2. *i.* $\mathrm{M}_{tm}(\cup, \cap, ^-, \oplus, \otimes)$ *is either decidable or not recursively enumerable.*

ii. $\mathrm{M}_{ex}(\cup, \cap, ^-, \oplus, \otimes)$ *is recursively enumerable if and only if* $\mathrm{MC}(\cup, \cap, ^-, \oplus, \otimes)$ *is decidable.*

iii. $\mathrm{M}_{ex}(^-, \oplus, \otimes)$ *is recursively enumerable if and only if* $\mathrm{MC}(^-, \oplus, \otimes)$ *is decidable.*

Glaßer showed that $\mathrm{MC}(\cup, \cap, ^-, \oplus, \otimes)$ is contained in $\Delta_2 = \Sigma_2 \cap \Pi_2$, where Σ_2 and Π_2 are the complexity classes forming the second level of the arithmetic hierarchy [1]. This entails the following theorem.

Theorem 3. $\mathrm{M}_{tm}(\cup, \cap, ^-, \oplus, \otimes) \in \Delta_2$ *and* $\mathrm{M}_{ex}(\cup, \cap, ^-, \oplus, \otimes) \in \Sigma_2$.

4. Undecidability of membership problems. We begin our study of the computational complexity of membership problems for recurrent systems by showing that the most general problem that we consider is hard. In fact, we will show that $\mathrm{M}_{ex}(\cup, \cap, ^-, \oplus, \otimes)$ is not decidable. We also expect this problem to be not even contained in RE, which we are not able to prove at the moment. However, there are some hints that lead to such an assumption some of which will be given in the conclusions section of this article.

Besides this hardness result we are also able to show a similar result for two other membership problems. It turns out that $\mathrm{M}_{ex}(\cup, \cap, \oplus, \otimes)$ and $\mathrm{M}_{ex}(^-, \oplus, \otimes)$ are undeciable, too. In case of the latter problem undecidability is already expected for the corresponding problem $\mathrm{MC}(^-, \oplus, \otimes)$, but has not yet been proved. The former problem, however, in its non-recurrent form is contained in NEXP, and undecidability of $\mathrm{M}_{ex}(\cup, \cap, \oplus, \otimes)$, therefore, is somehow surprising. Undecidability relies on the fact that we can express disjointness of two sets using restricted sets of operations, and then, we are able to decrement numbers.

We introduce a new problem $\mathrm{M}_{ex}(\mathrm{gr}, \oplus, \otimes, 1)$, and we will prove it to be undecidable. This proof heavily relies on the ability to enumerate all k-tuples over $\mathbf{N}$ for k any constant. Using this result we can reduce DIOPHANTINE to $\mathrm{M}_{ex}(\mathrm{gr}, \oplus, \otimes, 1)$. Finally, we will show that $\mathrm{M}_{ex}(\mathrm{gr}, \oplus, \otimes, 1)$ reduces to both problems $\mathrm{M}_{ex}(\cup, \cap, \oplus, \otimes)$ and $\mathrm{M}_{ex}(^-, \oplus, \otimes)$.

The function 1 merely represents the constant set $\{1\}$. Using this denotation, one might expect confusions between the meanings 1 and $\{1\}$. But the two meanings will always be distinguishable from each other. Let $A, B \subseteq \mathbf{N}$. We define function gr, which should be understood as *greater than*, as follows: if $B \subseteq A \oplus \mathbf{N}$ then $\mathrm{gr}(A, B) =_{\mathrm{def}} \{0\}$, otherwise $\mathrm{gr}(A, B) =_{\mathrm{def}} \{1\}$. Of course, gr is more a predicate than an operation or a

```
Enumerate_k:
1   begin
2      n := -1;
3      b_k := 0;
4      m_1 := 0; ...; m_k := 0;
5      n := n + 1;
6      i := min{i ∈ {1,...,k} : m_i ≥ 1} ∪ {k+1};
7      b_1 := 1; ...; b_{i-2} := 1;
8      m_{i-1} := b_{i-1};
9      b_{i-1} := b_{i-1} + 1;
10     m_i := m_i - 1;
11     goto 5
12  end.
```

Figure 2. An algorithm for enumeration of $\mathbf{N}^k$.

function. It is obvious that $\{\mathrm{gr}, \oplus, \otimes, 1\}$-functions always compute singleton sets, if they are applied to singleton sets only. Problem $\mathrm{M}_{ex}(\mathrm{gr}, \oplus, \otimes, 1)$ is the set of all pairs (S, b) where S is a recurrent $\{\mathrm{gr}, \oplus, \otimes, 1\}$-system, $b \in \mathbf{N}$ and there is $t \geq 0$ such that $b \in S(t)$.

Enumerating k-tuples. We first present an algorithm that enumerates all k-tuples over $\mathbf{N}$, $k \geq 1$. The algorithm $\mathtt{Enumerate}_k$, given in pseudocode in Figure 2, works like an iterated Cantor enumeration. Note that line 7 is short for a `for`-loop and can only be carried out for $i \geq 3$. Furthermore, variables m_0, b_0 and m_{k+1} are not initialized. Due to readability we avoid additional conditional tests, but the reader is free to include them in lines 8, 9 and 10. These variables are never used.

A *phase* of $\mathtt{Enumerate}_k$ is finished by reaching `goto` in line 11. The value of a variable in some phase is its content at the end of that phase. The *number* of a phase is represented by the counting variable n, and the start phase has number 0. A k-tuple $\mathbf{m} = (m_1, \ldots, m_k)$ over $\mathbf{N}$ is *generated* in phase n if $\mathbf{m}$ contains the contents of the variables $m_1, \ldots, m_k$ in phase n. In every phase the variables $b_1, \ldots, b_k$ are non-zero. We will show in the following that every $\mathbf{m} \in \mathbf{N}^k$ is generated in some phase. Consider the values of the variables of $\mathtt{Enumerate}_3$ during the first few phases.

n	0	1	2	3	**4**	5	6	7	**8**	9	10	**11**	12	13	**14**	15	16	17	**18**	19	20	**21**	22	23	**24**
i	4	4	3	2	1	4	3	2	1	3	2	1	2	1	1	4	3	2	1	3	2	1	2	1	1
b_3	1	2	2	2	2	3	3	3	3	3	3	3	3	3	3	4	4	4	4	4	4	4	4	4	4
b_2	1	1	2	2	2	1	2	2	2	3	3	3	3	3	3	1	2	2	2	3	3	3	3	3	3
b_1	1	1	1	2	2	1	1	2	2	1	2	2	3	3	3	1	1	2	2	1	2	2	3	3	3
m_3	0	1	0	0	0	2	1	1	1	0	0	0	0	0	0	3	2	2	2	1	1	1	1	1	1
m_2	0	0	1	0	0	0	1	0	0	2	1	1	0	0	0	0	1	0	0	2	1	1	0	0	0
m_1	0	0	0	1	0	0	0	1	0	0	1	0	2	1	0	0	0	1	0	0	1	0	2	1	0

For every $r \in [1, k]$ we define an *r-round* to be an interval $[n_1, n_2]$ for $n_1, n_2 \in \mathbf{N}$, $0 < n_1 < n_2$, such that $m_r = 0$ in phase $n_1{-}1$ and $m_r > 0$ in phase n_1 and n_2 is the smallest number such that $m_1 = \cdots = m_r = 0$ in phase n_2. We call phase n_1 the *beginning*, phase n_2 the *end* of r-round $[n_1, n_2]$. For example, phase 4 is the end of a 1-round, 2-round and 3-round of $\mathtt{Enumerate}_3$. Phase 11 is the end of another 1-round, but not the end of a 2-round. Bold numbers in our example above mark the ends of 1-rounds. We observe that, if $m_{r+1} > 0$ at the end of some r-round $[n_1, n_2]$, than $n_2{+}1$ is the beginning of another r-round. For example, in phase 12 a new 1-round starts *within* the 2-round $[9,14]$. Note that phase 5 is not the beginning of neither a 1-round nor a 2-round. In general, if ℓ is the value of m_r at the beginning of an r-round, $r>1$, then this r-round contains exactly ℓ $(r-1)$-rounds. We first show that number n_2 always exists.

Lemma 4. *Let $n_1>0$ fulfill the prerequisites of the beginning of an r-round for $r\in[1,k]$, and let ℓ be the value of m_r in phase n_1. Then $[n_1,n_2]$ is an r-round where $n_2 =_{\text{def}} n_1+\binom{\ell+r}{r}-1$.*

Proof: We show the claim by induction over r. If $r=1$, then ℓ is the value of m_1 in phase n_1. Since lines 7–9 of $\texttt{Enumerate}_k$ are not carried out, it takes ℓ phases to reduce the value of m_1 to 0. Hence, $[n_1,n_2]$ is a 1-round for $n_2 =_{\text{def}} n_1+\ell = n_1+\binom{\ell+1}{1}-1$. Let $r>1$. The value of m_r in phase $n_1{-}1$ is 0. Hence, $m_1=\cdots=m_r=0$ in phase $n_1{-}1$, since the value of i in phase n_1 is $r{+}1$. Furthermore, the values of $b_1,\ldots,b_{r-1}$ in phase n_1 are 1. So, in phase $n_1{+}1$ an $(r{-}1)$-round starts, and a new $(r{-}1)$-round starts hereafter if m_r is greater than 0 in phase $n_1{+}1$. This is repreated ℓ times, so that we find n_2 such that

$$n_2-n_1 = 1+\sum_{i=1}^{\ell}\binom{i+r-1}{r-1}-1 = \sum_{i=0}^{\ell}\binom{i+r-1}{r-1}-1 = \binom{\ell+r}{r}-1\,.$$

■

The set of k-rounds of $\texttt{Enumerate}_k$ partitions $\mathbf{N}\setminus\{0\}$. We can define a canonical ordering on the k-rounds where the 0-th k-round is phase 0 and the first k-round starts in phase 1.

Proposition 5. *The algorithm $\texttt{Enumerate}_k$ for $k\geq 1$ enumerates all k-tuples over $\mathbf{N}$.*

Proof: Let $\mathbf{a}=(a_1,\ldots,a_k)$ be a k-tuple over $\mathbf{N}$. Let $s =_{\text{def}} a_1+\cdots+a_k$. We will show that $\mathbf{a}$ is generated during the s-th k-round of $\texttt{Enumerate}_k$. If $s=0$ then $\mathbf{a}=0$, and $\mathbf{a}$ is generated in phase 0. At the beginning of the s-th k-round, it holds that $b_k=m_k{+}1=s{+}1$, $b_1=\cdots=b_{k-1}=1$ and $m_1=\cdots=m_{k-1}=0$. At the beginning of the $(s{-}a_k)$-th $(k{-}1)$-round within the s-th k-round, it holds that $m_1=\cdots=m_{k-2}=0$, $b_1=\cdots=b_{k-2}=1$, $m_{k-1}=s-a_k$, $b_{k-1}=s-a_k+1$, $m_k=a_k$ and $b_k=s+1$. An iterated application of this argumentation concludes the proof. ■

Note that $\texttt{Enumerate}_k$ does not establish a bijection—it is rather the case that every element of $\mathbf{N}^k$ is generated infinitely often.

The description of a component. The overall goal of our reduction is to simulate algorithm $\texttt{Enumerate}_k$ by an appropriate recurrent $\{\mathrm{gr},\oplus,\otimes,1\}$-system. In this subsection, we will define and describe a recurrent system that will generate one component of the enumerated k-tuple. For convenience, we define five auxiliary functions. Let $A,B\subseteq\mathbf{N}$. Then, $\mathrm{ueq}(A,B) =_{\text{def}} \mathrm{gr}(A,B)\oplus\mathrm{gr}(B,A)$ and

$$\begin{aligned}\mathrm{no}(A) =_{\text{def}} \mathrm{ueq}(A,1) \quad &\text{and}\quad 0 =_{\text{def}} \mathrm{no}(1)\\ \mathrm{eq}(A,B) =_{\text{def}} \mathrm{no}(\mathrm{ueq}(A,B)) \quad &\text{and}\quad \mathrm{or}(A,B) =_{\text{def}} \mathrm{ueq}(A\oplus B,0)\,.\end{aligned}$$

Note that ueq, no, eq, or can map to either 0 or 1. We define a recurrent $\{\mathrm{gr},\oplus,\otimes,1\}$-system.

Let $\mathcal{F}_M =_{\text{def}} \langle f_a, f_b, f_c, f_h, f_m \rangle$ where $\mathbf{x} =_{\text{def}} (a, b, c, h, m)$ and

$$
\begin{aligned}
f_a(\mathbf{x}) &=_{\text{def}} \text{eq}(h, m) \otimes \text{no}(f_{c'}(\mathbf{x}))\\
f_b(\mathbf{x}) &=_{\text{def}} (f_{a'}(\mathbf{x}) \otimes b) \oplus \text{or}(\text{no}(f_{a'}(\mathbf{x})), f_{c'}(\mathbf{x}))\\
f_c(\mathbf{x}) &=_{\text{def}} f_{c'}(\mathbf{x}) \otimes \text{eq}(h, 0)\\
f'(\mathbf{x}) &=_{\text{def}} \text{ueq}(h, 1 \oplus (m \otimes \text{no}(f_{c'}(\mathbf{x}))))\\
f''(\mathbf{x}) &=_{\text{def}} (m \otimes \text{no}(f_{c'}(\mathbf{x}))) \oplus (f'(\mathbf{x}) \otimes \text{ueq}(h, m \otimes \text{no}(f_{c'}(\mathbf{x}))))\\
f_m(\mathbf{x}) &=_{\text{def}} f_{a'}(\mathbf{x}) \otimes f''(\mathbf{x})\\
f_h(\mathbf{x}) &=_{\text{def}} (f_{a'}(\mathbf{x}) \otimes ((f'(\mathbf{x}) \otimes h) \oplus (\text{no}(f'(\mathbf{x})) \otimes f''(\mathbf{x})))) \oplus (\text{no}(f_{a'}(\mathbf{x})) \otimes f_{b'}(\mathbf{x}))\,.
\end{aligned}
$$

Functions $f_{a'}, f_{b'}, f_{c'}$ will be precised later. We define $f_{a'}(0) =_{\text{def}} 0$ and $f_{a'}(t{+}1) =_{\text{def}} f_{a'}(\mathcal{F}(t))$; $f_{b'}(t)$, $f_{c'}(t)$, $f'(t)$ and $f''(t)$ are defined similarly. Let $S_M = (\mathcal{F}_M, A_M)$ be our recurrent $\{\text{gr}, \oplus, \otimes, 1\}$-system where $A_M =_{\text{def}} (0,0,0,0,0)$. We will continue by proving important properties of S_M.

Lemma 6. *For every* $t \geq 0$

i. $f_b(1) \subseteq \mathbf{N} \oplus 1$ *implies* $f_b(t{+}1) \subseteq \mathbf{N} \oplus 1$,

ii. $f_h(t) \subseteq f_m(t) \oplus \mathbf{N}$.

Proof: We prove the claims by induction over t. Let the claims be true for $t \geq 0$. It holds that $f_b(t{+}2) \subseteq f_b(t{+}1) \oplus \mathbf{N}$ or $f_b(t{+}2) = 1$. For the second claim, we assume $f_{a'}(t{+}1) \neq 0$. If $f'(t{+}1) = 0$ then $f_h(t{+}1) = f_{a'}(t{+}1) \otimes f''(t{+}1)$. Let $f'(t{+}1) = 1$. Then $f_h(t{+}1) = f_{a'}(t{+}1) \otimes f_h(t)$. If $f_h(t) = 0$ then $f_m(t{+}1) = f_m(t) = 0$. Let $f_h(t) \neq 0$. If $\text{ueq}(f_h(t), f_m(t)) = 1$ then $f_h(t) \subseteq f_m(t) \oplus \mathbf{N} \oplus 1$ and $f_h(t{+}1) \subseteq f_m(t{+}1) \oplus \mathbf{N}$. If $\text{ueq}(f_h(t), f_m(t)) = 0$ then $f_m(t{+}1) \subset \{0, f_{a'}(t{+}1)\} \otimes \{1, f_m(t)\}$. ∎

Lemma 7. *Let* $0 \leq t_1 \leq t_2$.

i. Let $f_a(t_1) = f_{a'}(t_1) = 1$ *and* $f_{c'}(t_1) = 0$. *If* $f_{a'}(t_1{+}1) = 1$ *and* $f_{c'}(t_1{+}1) = 0$ *then* $f_a(t_1{+}1) = 1$, $f_c(t_1{+}1) = 0$, $f_b(t_1{+}1) = f_b(t_1)$ *and* $f_m(t_1{+}1) = f_m(t_1)$.

ii. If $f_{a'}(t) = 0$ *and* $f_{b'}(t) \subseteq \mathbf{N} \oplus 1$ *for all* $t \in [t_1, t_2]$ *and* $f_{c'}(t_1) = 1$ *then* $f_a(t) = 0$ *for all* $t \in [t_1, t_2]$.

Proof: Consider the assumptions of the first claim. $f_a(t_1) = 1$ implies $f_h(t_1{-}1) = f_m(t_1{-}1)$, hence $f_m(t_1) = f''(t_1) = f_m(t_1{-}1)$. Since $f_h(t_1) = f_h(t_1{-}1)$ or $f_h(t_1) = f''(t_1)$, it follows that $f_h(t_1) = f_m(t_1)$ which results in $f_a(t_1{+}1) = 1$ and $f_m(t_1{+}1) = f_m(t_1)$. The remaining claims of i. follow trivially. For the second claim: $f_{c'}(t_1) = 1$ implies $f_a(t_1) = 0$, and $f_m(t) = 0$ and $f_h(t) \subset \mathbf{N} \oplus 1$ for all $t \in [t_1, t_2]$, which yields $\text{eq}(f_m(t), f_h(t)) = 0$. ∎

Lemma 8. *Assume for all* $t \geq 0$ *that* $f_{c'}(t{+}1) = 1$ *implies* $f_a(t) = 1$, *and* $f_{a'}(t) = 1$ *and* $f_{a'}(t{+}1) = 0$ *implies* $f_{c'}(t{+}1) = 1$. *Let* $t_0 \in \mathbf{N}$ *such that* $f_a(t_0) = 0$ *and* $f_{a'}(t_0) = 1$. *Then, there is some* $t_1 > t_0$ *such that* $f_a(t_1) = 1$. *If* t_1 *is smallest possible and* $\text{eq}(f_h(t_0), f_m(t_0)) = 0$ *then* $\text{eq}(f_h(t_0), (f_m(t_1) + 1)) = 1$.

Proof: Let $k \geq 0$ be the smallest value such that $\text{eq}(f_h(t_0), (f_m(t_0) + \{k\})) = 1$. The existence of k is due to Lemma 6. We show the claim by induction over k. If $k = 0$ then $f_a(t_0{+}1) = 0$ would imply $f_{c'}(t_0{+}1) = 1$ and $f_a(t_0) = 1$. Hence, $t_1 =_{\text{def}} t_0 + 1$ fulfills the claim. If $k = 1$ then $f_a(t_0{+}1) = 0$, and $f_{a'}(t_0{+}1) = 1$ and $f_{c'}(t_0{+}1) = 0$ due to our assumptions. It follows that $f'(t_0{+}1) = 0$ and $f_h(t_0{+}1) = f_m(t_0{+}1) = f''(t_0{+}1) = f_m(t_0)$. Now, $\text{eq}(f_h(t_0{+}1), f_m(t_0{+}1)) = 1$ and the existence of $t_1 > t_0$ such that $f_a(t_1) = 1$ follows by applying case $k = 0$. Furthermore, $\text{eq}(f_h(t_0), (f_m(t_1) + 1)) = 1$. Let $k \geq 2$. Then,

$f_a(t_0{+}1) = 0$, $f_{a'}(t_0{+}1) = 1$ and $f_{c'}(t_0{+}1) = 0$. We conclude $f'(t_0{+}1) = 1$, $f_m(t_0{+}1) = f''(t_0{+}1) = f_m(t_0) + 1$ and $f_h(t_0{+}1) = f_h(t_0)$. It holds that the smallest number k' such that $\mathrm{eq}(f_h(t_0), (f_m(t_0) + \{k'\})) = 1$ is one less than k, and we can conclude the proof by applying the induction hypothesis. ∎

Assembling components. This subsection aims to achieve a recurrent $\{\mathrm{gr}, \oplus, \otimes, 1\}$-system that simulates the work of $\texttt{Enumerate}_k$. Let $S_M^{(1)}, \dots, S_M^{(k)}$ be k copies of system S_M defined in the last subsection. By $f_a^{(i)}, f_{a'}^{(i)}, \dots, f_h^{(i)}, f_m^{(i)}$ we denote the functions of $S_M^{(i)}$, $i \in [1,k]$. Consider the following definition.

$$\begin{aligned} \mathcal{F}_E^k &=_{\mathrm{def}} \langle f_a^{(1)}, f_b^{(1)}, f_c^{(1)}, f_h^{(1)}, f_m^{(1)}, f_a^{(2)}, \dots, f_b^{(k)}, f_c^{(k)}, f_h^{(k)}, f_m^{(k)}, f_A \rangle \\ \mathbf{x} &=_{\mathrm{def}} (a_1,\ b_1,\ c_1,\ h_1,\ m_1,\ a_2,\ \dots,\ b_k,\ c_k,\ h_k,\ m_k,\ x)\,. \end{aligned}$$

Then, $f_a^{(i)}(\mathbf{x}) = f_a(a_i, b_i, c_i, h_i, m_i)$ and similarly for $f_b^{(i)}, f_c^{(i)}, f_h^{(i)}, f_m^{(i)}$, $i \in [1,k]$. Furthermore, let $f_A(\mathbf{x}) =_{\mathrm{def}} x \oplus f_c^{(k)}(\mathbf{x})$. Now, we can precise the definitions of $f_{a'}^{(i)}, f_{b'}^{(i)}, f_{c'}^{(i)}$. For every $i \in [1, k{-}1]$, let

$$\begin{aligned} f_{a'}^{(i)}(\mathbf{x}) &=_{\mathrm{def}} f_a^{(i+1)}(\mathbf{x}) \\ f_{a'}^{(k)}(\mathbf{x}) &=_{\mathrm{def}} \mathrm{no}(f_c^{(k)}(\mathbf{x})) \end{aligned} \quad \text{and} \quad \begin{aligned} f_{b'}^{(i)}(\mathbf{x}) &=_{\mathrm{def}} f_b^{(i+1)}(\mathbf{x}) \\ f_{b'}^{(k)}(\mathbf{x}) &=_{\mathrm{def}} f_A(\mathbf{x}) \end{aligned} \quad \text{and} \quad \begin{aligned} f_{c'}^{(1)}(\mathbf{x}) &=_{\mathrm{def}} a_1 \\ f_{c'}^{(i+1)}(\mathbf{x}) &=_{\mathrm{def}} f_c^{(i)}(\mathbf{x})\,. \end{aligned}$$

Let $S_E^k =_{\mathrm{def}} (\mathcal{F}_E^k, A_E^k)$ where $A_E^k =_{\mathrm{def}} (1, 0, 0, \dots, 0)$. First observe that the functions of $\mathcal{F}_E^k$ are well-defined. We want to show that for every $\mathbf{a} \in \mathbf{N}^k$ there is some $t \in \mathbf{N}$ such that $\mathbf{a} \simeq (f^{(1)}(t), \dots, f^{(k)}(t))$. We begin by showing some properties of S_E^k that are needed to apply the results obtained in the last subsection.

Lemma 9. *$f_a^{(i)}(1) = 0$, $f_c^{(i)}(1) = 1$ and $f_b^{(i)}(t) \subseteq \mathbf{N} + 1$ for every $t \geq 1$ and $i \in [1,k]$.*

Note that $f_a^{(i)}(t), f_{a'}^{(i)}(t), f_c^{(i)}(t), f_{c'}^{(i)}(t) \in \{0,1\}$ for every $i \in [1,k]$ and $t \in \mathbf{N}$. Furthermore, every function of $\mathcal{F}_E^k$ always computes singleton sets.

Lemma 10. *For every $t \in \mathbf{N}$ and $i \in [1,k]$*

i. If $f_a^{(i)}(t) = 1$ and $f_a^{(i)}(t{+}1) = 0$ then $f_{c'}^{(i)}(t{+}1) = 1$,

ii. If $f_{a'}^{(i)}(t) = 0$ then $f_a^{(i)}(t) = 0$,

iii. $f_{c'}^{(1)}(t{+}1) = 1$ if and only if $f_a^{(1)}(t) = \ldots = f_a^{(k)}(t) = 1$.

Proof: We show statement i by induction over i. Suppose $f_{c'}^{(i)}(t{+}1) = 0$. Then, $f_h^{(i)}(t) \cap f_m^{(i)}(t) = \emptyset$. Due to $f_a^{(i)}(t) = 1$, it holds that $f_m^{(i)}(t) \neq f_m^{(i)}(t{-}1)$ if and only if $f_{a'}^{(i)}(t) = 0$. Hence, $\mathrm{eq}(f_h^{(i)}(t), f_m^{(i)}(t)) = 0$ if and only if $f_{a'}^{(i)}(t) = 0$. If $i = k$ then $f_{a'}^{(k)}(t) = \mathrm{no}(f_c^{(k)}(t))$, thus $f_c^{(k)}(t) = f_{c'}^{(k)}(t) = 1$ and $f_a^{(k)}(t) = 0$. If $i < k$ then $f_{a'}^{(i)}(t) = f_a^{(i+1)}(t) = 0$. Let $t_0 < t$ be smallest possible such that $f_a^{(i+1)}(t') = 0$ for all $t' \in [t_0, t]$. By induction hypothesis or due to Lemma 9, $f_c^{(i)}(t_0{+}1) = 1$, and Lemmata 9 and 7 contradict to $f_a^{(i)}(t) = 1$.

Concerning statement ii, let $f_{a'}^{(i)}(t) = 0$. Let $t_0 \leq t$ be smallest possible such that $f_{a'}^{(i)}(t') = 0$ for all $t' \in [t_0, t]$. If $t_0 = 0$ then $f_{c'}^{(i)}(0) = 1$ due to Lemma 9; otherwise if $i = k$ then $f_{c'}^{(k)}(t_0) = 1$ due to construction; otherwise $f_{c'}^{(i)}(t_0) = 1$ due to statement i. Then, statement ii follows by Lemmata 9 and 7. Finally, $f_a^{(1)}(t) = \ldots = f_a^{(k)}(t) = 1$ implies $f_{c'}^{(1)}(t{+}1) = 1$. If $f_a^{(i)}(t) = 0$ for some $i \in [1,k]$, statement ii yields $f_a^{(1)}(t) = 0$, hence $f_{c'}^{(1)}(t{+}1) = 0$. ∎

Lemma 11. *If $f_{c'}^{(1)}(t_0) = 1$ for $t_0 \in \mathbf{N}$ then there is $t_1 > t_0$ such that $f_a^{(1)}(t_1) = 1$.*

Proof: Let $i \in [1,k]$ be largest possible such that $f_{c'}^{(i)}(t_0) = 1$. We will show that there is $t_1 > t_0$ such that $f_a^{(i)}(t_1) = 1$. Iterated application of this claim proves the statement. Note that the assumptions of Lemma 8 are fulfilled due to Lemma 10. If $f_c^{(k)}(t_0) = 1$ then $f_a^{(k)}(t_0) = 0$ and $f_{c'}^{(1)}(t_0{+}1) = 0$. Hence, $f_{a'}^{(k)}(t_0{+}1) = 1$. If $f_c^{(k)}(t_0) = 0$ then $f_{a'}^{(i)}(t_0) = 1$. Then, by Lemma 8 there is $t_1 \geq t_0{+}1$ such that $f_a^{(i)}(t_1) = 1$. ∎

The last results justify the following definition of a phase of recurrent system S_E^k. A *phase* $[t_0, t_1]$ is an interval where $f_{c'}^{(1)}(t_0) = 1$, $f_a^{(1)}(t_1) = 1$ and $f_{c'}^{(1)}(t) = 0$ for every $t \in [t_0{+}1, t_1]$. In other words, t_1 is the smallest value such that $t_1 > t_0$ and $f_a^{(1)}(t_1) = 1$. Observe that $[t_1{+}1, t_2]$ is another phase for some t_2.

Lemma 12. *Let $[t_0, t_1]$ be a phase, and let $i \in [1,k]$ such that $f_{c'}^{(i)}(t_0) = 1$. If $f_c^{(i)}(t_0) = 0$ then $f_m^{(i)}(t_0) = f_m^{(i)}(t_1) + 1$; otherwise $f_{b'}^{(i)}(t_1) = f_m^{(i)}(t_1) + 1$.*

Proof: Let $f_c^{(i)}(t_0) = 0$. Then $f_{a'}^{(i)}(t_0) = 1$ and $f_a^{(i)}(t_0) = 0$. If $f_h^{(i)}(t_0) = 1$ then $f_h^{(i)}(t_0{+}1) = f_m^{(i)}(t_0{+}1) = 0$ and $f_a^{(i)}(t_0{+}2) = 1$. If $f_h^{(i)}(t_0) \neq 1$ then $f_{a'}^{(i)}(t_0{+}1) = 1$, $f_a^{(i)}(t_0{+}1) = 0$, $f_h^{(i)}(t_0{+}1) = f_h^{(i)}(t_0)$ and $f_m^{(i)}(t_0{+}1) = 1$. Hence, $f_h^{(i)}(t_0{+}1) \neq f_m^{(i)}(t_0{+}1)$, and due to Lemma 8, there is a smallest $t' > t_0{+}1$ such that $f_a^{(i)}(t') = 1$ and $f_h^{(i)}(t_0) = f_m^{(i)}(t') + 1$. Due to Lemma 7, the claim holds. Now, let $f_c^{(i)}(t_0) = 1$. Then, there is a largest $t \in [t_0, t_1]$ such that $f_{a'}^{(i)}(t{-}1) = 0$ and $f_{a'}^{(i)}(t) = 1$. Then $f_h^{(i)}(t{-}1) = f_{b'}^{(i)}(t{-}1)$ and $f_m^{(i)}(t{-}1) = 0$. Due to Lemma 7, $f_{b'}^{(i)}(t_1) = f_{b'}^{(i)}(t{-}1)$. Now the situation equals the situation of the first case. ∎

Corollary 13. *Let $[t_0, t_1]$ be a phase such that $f_c^{(i)}(t_0) = 1$ for $i \in [1,k]$. Then, $f_b^{(i)}(t_1) = \ldots = f_b^{(1)}(t_1) = 1$ and $f_m^{(i-1)}(t_1) = \ldots = f_m^{(1)}(t_1) = 0$.*

Proof: Let $t \in [t_0, t_1]$ be largest possible such that $f_{a'}^{(i)}(t) = 0$. Then, $f_b^{(j)}(t_1) = 1$ for every $j \in [1,i]$. The statement follows by Lemma 12. ∎

We can order the set of phases according to the left or right endpoints and enumerate them accordingly starting with phase 0. Let $s^{(i)}$ denote the right endpoint of phase i, $i \in \mathbf{N}$. Note that $s^{(i)} + 1$ is the left endpoint of phase $i{+}1$. Then, for every $i \geq 0$ let

$$\mu(i) =_{\text{def}} (f_m^{(k)}(s^{(i)}), \ldots, f_m^{(1)}(s^{(i)})) \quad \text{and} \quad \kappa(i) =_{\text{def}} (f_{b'}^{(k)}(s^{(i)}), \ldots, f_{b'}^{(1)}(s^{(i)}))\,.$$

Since functions $f_m^{(1)}, \ldots, f_m^{(k)}$ compute only singleton sets, we can say that $\mu(i)$ represents an element of $\mathbf{N}^k$. We will show that the sequences defined by μ and generated by $\texttt{Enumerate}_k$ can be considered equal. Let

$$\nu(i) =_{\text{def}} (m_k^{(i)}, \ldots, m_1^{(i)}) \quad \text{and} \quad \beta(i) =_{\text{def}} (b_k^{(i)}, \ldots, b_1^{(i)})$$

where $m_j^{(i)}$ and $b_j^{(i)}$ denote the values of variables m_j and b_j in phase i of $\texttt{Enumerate}_k$, respectively.

Proposition 14. *For every $n \geq 0$: $\mu(n) \simeq \nu(n)$.*

Proof: We will show the claim by induction over the phases of $\texttt{Enumerate}_k$ and S_E^k. Subsequently, i means that variable of $\texttt{Enumerate}_k$. It is obvious that $\nu(0) = (0, \ldots, 0)$ and $\beta(0) = (1, \ldots, 1)$. Due to Lemma 9, $f_c^{(k)}(1) = 1$, and Corollary 13 shows $\mu(0) = (f_m^{(k)}(s^{(0)}), 0, \ldots, 0)$ and $\kappa(0) = (f_A(s^{(0)}), 1, \ldots, 1)$. By construction and due to Lemma 10, it holds that $f_A(1) = f_A(s^{(0)}) = 1$. Finally, $f_m^{(k)}(s^{(0)}) = 0$ by Lemma 12. Now, let $\mu(n) \simeq \nu(n)$ and $\kappa(n) \simeq \beta(n)$ for $n \geq 0$. Let $j \in [1,k]$ be largest possible such that $f_{c'}^{(j)}(s^{(n)}{+}1) = 1$.

Recall that $s^{(n)}{+}1$ is the left endpoint of phase $n{+}1$. If $f_c^{(k)}(s^{(n)}{+}1) = 0$ then j is the least number such that $f_m^{(j)}(s^{(n)}) \neq 0$, and j is equal to the value of i in $\texttt{Enumerate}_k$ in phase $n{+}1$. Note that $f_{a'}^{(j)}(t) = 1$ for all $t \in [s^{(n)}{+}1, s^{(n+1)}]$ and $f_{c'}^{(j)}(t) = 0$ for all $t \in [s^{(n)}{+}2, s^{(n+1)}]$. Hence, $f_b^{(j)}(s^{(n)}{+}1) = f_b^{(j)}(s^{(n+1)}) = f_b^{(j)}(s^{(n)}) + 1$. Furthermore, due to Lemma 12, $f_m^{(j)}(s^{(n)}) = f_m^{(j)}(s^{(n)}{+}1) = f_m^{(j)}(s^{(n+1)}) + 1$ and $f_b^{(j)}(s^{(n+1)}) = f_m^{(j-1)}(s^{(n+1)}) + 1 = f_b^{(j)}(s^{(n)}) + 1$. By Lemmata 7 and 13, $\mu(n{+}1) = \nu(n{+}1)$ and $\kappa(n{+}1) = \beta(n{+}1)$. Now, let $f_c^{(k)}(s^{(n)}{+}1) = 1$. Then, $i = k + 1$ in phase $n{+}1$ of $\texttt{Enumerate}_k$. Similarly to the induction step, $f_A(s^{(n+1)}) = f_A(s^{(n)}) + 1$, and hence $f_{b'}^{(k)}(s^{(n+1)}) = f_{b'}^{(k)}(s^{(n)}) + 1 = f_m^{(k)}(s^{(n+1)}) + 1$. The remaining part follows analogously to the case discussed above. ■

Corollary 15. *For every* $\mathbf{a} \in \mathbf{N}^k$ *there is* $n \in \mathbf{N}$ *such that* $\mathbf{a} \simeq \mu(n)$.

Undecidable problems. It remains one step to accomplish the undecidability proof. Besides the halting problem the probably most famous undecidable problem is Hilbert's tenth problem concering Diophantine equations [2].

Definition 3. *A* ***Diophantine equation*** *is an equation on variables* $x_1, \ldots, x_k$ *of the form* $p(x_1, \ldots, x_k) = q(x_1, \ldots, x_k)$ *where* p *and* q *are polynomials in* $x_1, \ldots, x_k$ *with natural coefficients. The problem* DIOPHANTINE *asks whether a Diophantine equation has a solution in natural numbers.*

Hilbert expected an algorithm that would decide DIOPHANTINE. However, using the results of theoretical computer science, Matiyasevich proved the impossibility of finding such an algorithm.

Theorem 16. **[5]** DIOPHANTINE *is undecidable.*

We will show that DIOPHANTINE reduces to $\mathrm{M}_{ex}(\mathrm{gr}, \oplus, \otimes, 1)$, which proves undecidability of the latter problem. Finally, we will reduce $\mathrm{M}_{ex}(\mathrm{gr}, \oplus, \otimes, 1)$ to $\mathrm{M}_{ex}(\cup, \cap, \oplus, \otimes)$ and $\mathrm{M}_{ex}({}^{-}, \oplus, \otimes)$, which shows undecidability of these problems. Since the halting problem is complete in Σ_1, the first level of the arithmetical hierarchy, we can at least conclude Σ_1-completeness of $\mathrm{M}_{ex}(\cup, \cap, \oplus, \otimes)$. We assume that a Diophantine equation is given as a set of quadruples where each quadruple represents a summand: we find an entry indicating that it belongs to the left or the right hand side, we find the index i of the variable x_i, its factor and exponent. The latter two numbers are given in binary form.

Lemma 17. DIOPHANTINE $\leq_m^{\mathrm{L}} \mathrm{M}_{ex}(\mathrm{gr}, \oplus, \otimes, 1)$.

Proof: Let $p(x_1, \ldots, x_k) = q(x_1, \ldots, x_k)$ be an instance of DIOPHANTINE. It holds that $(x_i)^{2^\nu} = (x_i)^{2^{\nu-1}} \otimes (x_i)^{2^{\nu-1}}$ for $\nu > 0$ and $(x_i)^{\nu+\alpha} = (x_i)^\nu \otimes (x_i)^\alpha$ for $\nu > \alpha$, similarly $2^\nu = 2^{\nu-1} \otimes 2^{\nu-1}$ for $\nu > 0$. So, in logarithmic space we can compute a circuit representation for p and q and a $\{\mathrm{gr}, \oplus, \otimes, 1\}$-function $f_{p=q}$ such that $f_{p=q}(a_1, \ldots, a_k) = 1 \leftrightarrow p(a_1, \ldots, a_k) = q(a_1, \ldots, a_k)$. We obtain recurrent $\{\mathrm{gr}, \oplus, \otimes, 1\}$-system S from S_E^k by adding a new component $(f_{p=q}, 0)$, and $f_{p=q}$ depends on the variables of S_E^k associated with $f_m^{(1)}, \ldots, f_m^{(k)}$. Now, it holds that $1 \in [S]$ if and only if there are $a_1, \ldots, a_k \in \mathbf{N}$ such that $p(a_1, \ldots, a_k) = q(a_1, \ldots, a_k)$. ■

Theorem 18. $\mathrm{M}_{ex}(\cup, \cap, \oplus, \otimes)$ *and* $\mathrm{M}_{ex}({}^{-}, \oplus, \otimes)$ *are undecidable.*

Proof: Let (S, b) be an instance of $\mathrm{M}_{ex}(\mathrm{gr}, \oplus, \otimes, 1)$, $n =_{\mathrm{def}} \dim S$. By adding a component $(f_{n+1}(\mathbf{x}) =_{\mathrm{def}} x_{n+1}, 1)$ for $\mathbf{x} =_{\mathrm{def}} (x_1, \ldots, x_{n+1})$, replacing each occurence of 1 in the functions of S by x_{n+1} and reindexing, $\mathrm{M}_{ex}(\mathrm{gr}, \oplus, \otimes, 1)$ reduces to $\mathrm{M}_{ex}(\mathrm{gr}, \oplus, \otimes)$. It suffices to show that gr can be expressed as a $\{\cup, \cap, \oplus, \otimes\}$- and a $\{{}^{-}, \oplus, \otimes\}$-function. We begin with the second case. Consider the following definitions:

$$
\begin{aligned}
f_1(x,z) &=_{\mathrm{def}} \overline{\overline{\overline{\overline{0} \oplus z} \oplus ((x \oplus 1) \otimes (\overline{1} \oplus \overline{2}))} \otimes 0} \\
f_2(x,z) &=_{\mathrm{def}} \overline{2 \otimes f_1(x,z)} \otimes ((2 \otimes f_1(x,z)) \oplus 1) \\
f_3(x,z) &=_{\mathrm{def}} \overline{\overline{\overline{f_1(x,z)} \oplus 1} \oplus f_2(x,z)}\,.
\end{aligned}
$$

Let $A, B \subseteq \mathbf{N}$. Let $a_0 =_{\mathrm{def}} \min A$ and $b_0 =_{\mathrm{def}} \min B$. Observe that $[0, b_0] \subseteq \overline{\overline{0} \oplus B} \subseteq \mathbf{N} \setminus \{b_0{+}1\}$. Furthermore, $\{a_0{+}1\} \otimes \mathbf{N} \subseteq (A \oplus 1) \otimes (\overline{1} \oplus \overline{2}) \subseteq \mathbf{N} \setminus [1, a_0]$. Thus,

$$
f_1(A,B) = \begin{cases} \mathbf{N} & \text{, if } B \subseteq A \oplus \mathbf{N} \\ \mathbf{N} \oplus 1 & \text{otherwise}\,. \end{cases}
$$

Evaluating $f_2(A,B)$ and $f_3(A,B)$ for both cases shows $f_3(A,B) = \mathrm{gr}(A,B)$ for all $A, B \subseteq \mathbf{N}$. Now, let f be an n-ary $\{\mathrm{gr}, \oplus, \otimes\}$-function represented by a circuit containing ν vertices. Observe that for finite and non-empty $B_1, \ldots, B_n \subseteq \mathbf{N}$ it holds that $f(B_1, \ldots, B_n) \subseteq [0, (b_0{+}1)^{2^\nu}]$ where $b_0 =_{\mathrm{def}} \max(B_1 \cup \ldots \cup B_n)$. Furthermore, observe that for $\mathrm{el}(x) =_{\mathrm{def}} (((1 \oplus x) \otimes x) \oplus x)$ and $a \in \mathbf{N}$: $\mathrm{el}([0,a]) = [0, a(a{+}2)]$. Hence, we can generate function $\mathrm{el}_{\nu+1}(x) =_{\mathrm{def}} \mathrm{el}(\mathrm{el}_\nu(x))$ in space linear in ν, and for every $a \in \mathbf{N}$ holds $[0, a^{2^\nu}] \subseteq \mathrm{el}_\nu([0,a])$. Let $S = (\mathcal{F}, A)$ be a recurrent $\{\mathrm{gr}, \oplus, \otimes\}$-system, $n =_{\mathrm{def}} \dim S$. Let $a_0 =_{\mathrm{def}} \max A$. In space linear in the length if a_0 we can generate function $g(\mathbf{x})$ such that $[0, a_0] \subseteq g(A)$: $[0,1] \oplus [0,1] = [0,2], [0,2] \oplus [0,2] = [0,4], \ldots$. Let ν denote the largest number of vertices of the circuit representations of the functions in S. Let S' emerge from S by adding components $(\langle f_{n+1}(\mathbf{x}) =_{\mathrm{def}} x_{n+1}, f_{n+2}(\mathbf{x}) =_{\mathrm{def}} \mathrm{el}_\nu(g(x_{n+1} \cup x_{n+2}))\rangle, (1,0))$ where $\mathbf{x} =_{\mathrm{def}} (x_1, \ldots, x_{n+2})$. Then, for every $t \in \mathbf{N}$, $f_{n+2}(t{+}1)$ contains all numbers that are not larger than any number that can be generated by an $\{\mathrm{gr}, \oplus, \otimes\}$-function represented by a circuit on at most ν vertices applied to $\mathcal{F}(t)$. Combining all these results yields the following identity, which concludes the proof. For every $A =_{\mathrm{def}} \{a\}$ and $B =_{\mathrm{def}} \{b\}$, $a, b \in \mathbf{N}$, that can be computed from $\mathcal{F}(t)$:

$$
\mathrm{gr}(A,B) = \Big((B \cap (A \oplus f_{n+2}(t{+}1))) \otimes 0\Big) \cup \Big(((A \cap (B \oplus f_{n+2}(t{+}1) \oplus 1)) \otimes 0) \oplus 1\Big)\,.
$$

It only remains to make f_n the output function of the resulting recurrent $\{\cup, \cap, \oplus, \otimes\}$-system. ■

Corollary 19. *$\mathrm{M}_{ex}(\mathrm{gr}, \oplus, \otimes, 1)$ and $\mathrm{M}_{ex}(\cup, \cap, \oplus, \otimes)$ are Σ_1-complete.*

Proof: Since $\{\cup, \cap, \oplus, \otimes\}$-functions on finite sets compute finite sets, $\mathrm{M}_{tm}(\cup, \cap, \oplus, \otimes)$ is decidable, and $\mathrm{M}_{ex}(\cup, \cap, \oplus, \otimes)$ is the projection of $\mathrm{M}_{tm}(\cup, \cap, \oplus, \otimes)$. ■

5. Easiest membership problems. In the last section, we considered the hardest membership problems that we deal with in this paper. This section is dedicated to our easiest problems: $\mathrm{M}_{tm}(^-)$, $\mathrm{M}_{tm}(\cup)$, $\mathrm{M}_{tm}(\cap)$, $\mathrm{M}_{ex}(^-)$ and $\mathrm{M}_{ex}(\cup)$. These are the problems solvable in nondeterministic logarithmic space. One might suspect that nondeterministic logarithmic space is needed because of the input representation that we have chosen. As we already mentioned McKenzie and Wagner additionally required a topological ordering of the vertices of the circuits to test correctness of the input representation in (deterministic) logarithmic space. In our case, however, this would not lower the bound as we will see. At the first glance surprisingly, $\mathrm{M}_{tm}(^-)$ can nevertheless be solved in deterministic logarithmic space, but this is due to the fact that $\{^-\}$-functions are very easily representable.

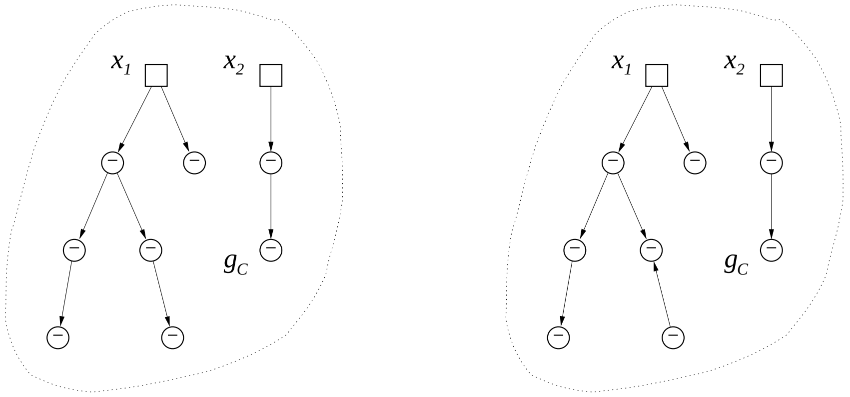

Figure 3. The left circuit represents the $\{^-\}$-function $f(x_1, x_2) = x_2$ whereas the right structure is not a circuit.

For the rest of the paper, proofs showing containedness results for existential membership problems follow the same model. First, we will give an upper bound for the complexity of deciding $\mathrm{M}_{tm}(\mathcal{O})$ for some set $\mathcal{O} \subseteq \{\cup, \cap, ^-, \oplus, \otimes\}$. Second, we bound the value of t by some number r such that $(S, b) \in \mathrm{M}_{ex}(\mathcal{O})$ if and only if there is $t < r$ such that $(S, t, b) \in \mathrm{M}_{tm}(\mathcal{O})$. Bound r normally depends on b and the dimension of S.

Lemma 20. $\mathrm{M}_{tm}(^-)$ *is in* L.

Proof: $\{^-\}$-Functions are represented by circuits that are unions of directed trees. (Remember that circuits can have several sinks.) Every vertex of such a circuit that is not an input vertex has exactly one predecessor (but may have several successors), and orientations point away from input vertices. (An example is given in Figure 3.) Hence, syntactical correctness of instances of $\mathrm{M}_{tm}(^-)$ can be tested in logarithmic space. Equally in logarithmic space it can be tested, given a $\{^-\}$-function $f(x)$, whether $f(x) = x$ or $f(x) = \overline{x}$ by determining the pairity of the length of the path from the output vertex to the unique accessible input vertex. Let (S, t, b) be an instance of $\mathrm{M}_{tm}(^-)$ where $S = (\mathcal{F}, A)$, $n =_{\mathrm{def}} \dim S$, $\mathcal{F} = \langle f_1, \ldots, f_n \rangle$ and $A = (a_1, \ldots, a_n)$. Let $x =_{\mathrm{def}} (x_1, \ldots, x_n)$. It holds that $b \in f_i(0)$ if and only if $b = a_i$, $i \in [1, n]$, and $b \in f_i(t'{+}1)$, $t' \geq 0$, if and only if $b \in f_j(t')$, if $f_i(\mathbf{x}) = x_j$, and $b \notin f_j(t')$, if $f_i(\mathbf{x}) = \overline{x_j}$. Consider the sequence $P_\ell =_{\mathrm{def}} (\nu_0, \ldots, \nu_\ell)$, $\ell \geq 0$, where $\nu_\ell =_{\mathrm{def}} n$ and $f_{\nu_{i+1}}(\mathbf{x}) = x_{\nu_i}$ or $f_{\nu_{i+1}}(\mathbf{x}) = \overline{x_{\nu_i}}$, $i \in [0, \ell{-}1]$. Consider P_n. Observe that P_n is the final part of every sequence $P_{t'}$, $t' \geq n$. There are largest $\alpha, \omega \in [0, n]$, $\alpha > \omega$, such that $\nu_\alpha = \nu_\omega$ in P_n. In logarithmic space we can determine α and ω. If $t \leq 3n$, $b \in S(t)$ can be decided straightforwardly. Let $t > 3n$. Note that in logarithmic space we cannot always count till t. Determine $r \in [0, 2\delta{-}1]$, $\delta =_{\mathrm{def}} \alpha - \omega$, such that $t \equiv n - \alpha + r \pmod{2\delta}$, which can be done in logarithmic space. It holds that $b \in S(t)$ if and only if $b \in S(n - \alpha + r)$, which can be decided in logarithmic space. ∎

If we call the subsequence $(\nu_\omega, \ldots, \nu_\alpha)$ of P_n a cycle, $n{-}\alpha$ denotes the length of the path to reach that cycle. The numbers $\nu_{\alpha+1}, \ldots, \nu_n$ appear only once in P_t, if $t > n$. The proof of the following lemma uses similar arguments. We say that a $\{\cup\}$-function $f = f(x_1, \ldots, x_n)$ *depends* on x_i, $i \in [1, n]$, if there is a circuit representation C of f such that there is a path in C from the input vertex labelled with i to the output vertex of C. Observe that every possible circuit representation has this property, if there is one such circuit. Equally, we can say that f depends on x_i, if $b \in A_i$ implies $b \in f(A_1, \ldots, A_n)$ for $A_1, \ldots, A_n \subseteq \mathbf{N}$. In a similar fashion we can extend the definition of dependency to all $\{\cup, \cap, ^-, \oplus, \otimes\}$-functions.

Lemma 21. $\mathrm{M}_{tm}(\cup)$ *and* $\mathrm{M}_{tm}(\cap)$ *are in* NL.

Proof: We will prove only the containedness of $\mathrm{M}_{tm}(\cup)$ in NL. A similar proof shows $\mathrm{M}_{tm}(\cap) \in \mathrm{coNL}$, and the statement holds due to Immerman's and Szelepcsényi's theorem [3], [11]. Let (S, t, b) be an instance of $\mathrm{M}_{tm}(\cup)$, whose syntactical correctness can be verified in nondeterministic logarithmic space. Let $S = (\mathcal{F}, A)$ where $\mathcal{F} = \langle f_1, \ldots, f_n \rangle$, $n =_{\mathrm{def}} \dim S$, and $A = (a_1, \ldots, a_n)$. Similar to the proof of Lemma 20, $b \in f_i(t'{+}1)$, $t' \geq 0$, if and only if $b \in f_j(t')$ for some $j \in [1, n]$ such that f_i depends on x_j. Then, $b \in S(t)$ if and only if there is a sequence $P =_{\mathrm{def}} (\nu_0, \ldots, \nu_t)$ where $\nu_t = n$, $\nu_i \in [1, n]$ and $f_{\nu_{i+1}}$ depends on x_{ν_i}, $i \in [0, t{-}1]$, and $a_{\nu_0} = b$. We will call such sequences *paths*. If $t \leq (n+1)^2$ we can find a path straightforwardly by guessing t values ν_i and verifying the dependency property. Now, let $t > (n+1)^2$. Let $\alpha \geq 0$ be smallest possible such that there is $\delta > 0$ satisfying $\nu_\alpha = \nu_{\alpha+\delta}$. We choose δ such that $\alpha + \delta \leq n$. Consider sequence $(\nu_\alpha, \nu_{\alpha+\delta}, \ldots, \nu_{\alpha+k\delta})$ where $t - \delta < \alpha + k\delta \leq t$. There is a path $P' =_{\mathrm{def}} (\nu'_0, \ldots, \nu'_t)$ where $\nu'_0 = \nu_0$ and $\nu'_\alpha = \nu'_{\alpha+\delta} = \ldots = \nu'_{\alpha+\ell\delta} = \nu_\alpha$ for $\ell =_{\mathrm{def}} k - n + 1$, since $\nu'_{\alpha+\ell\delta}, \nu'_{\alpha+(\ell+1)\delta}, \ldots, \nu'_{\alpha+k\delta}$ can be chosen mutually different. This characterisation gives an NL-algorithm to solve the problem. First, guess numbers $\nu \in [1, n]$ and $\delta \leq n$ and verify the existence of a path $(z_0, \ldots, z_\delta)$ where $z_0 = z_\delta = \nu$; $a_{z_0} = b$ does not have to hold. Finally, verify the existence of a path $(u_0, \ldots, u_m)$ where $m < n{+}(n{-}1){\cdot}\delta{+}\delta = n{\cdot}(\delta{+}1)$, $m \equiv t \pmod{\delta}$, there is $i < n$ such that $u_i = \nu$ and $a_{u_0} = b$ ($\mathrm{M}_{tm}(\cap)$ requires inequality in this condition, which is the only difference between $\mathrm{M}_{tm}(\cup)$ and $\mathrm{M}_{tm}(\cap)$). The remainder condition $m \equiv t \pmod{\delta}$ can be checked in logarithmic space. ■

Theorem 22. *i.* $\mathrm{M}_{ex}(\emptyset)$ *and* $\mathrm{M}_{ex}(^{-})$ *are in* L.

ii. $\mathrm{M}_{ex}(\cup)$ *is* NL*-complete.*

Proof: An n-ary $\emptyset$-function can only be of the form $\varphi(\mathbf{x}) =_{\mathrm{def}} x_i$ for $i \in [1, n]$. Hence, syntactical correctness of inputs for $\mathrm{M}_{ex}(\emptyset)$ can be tested in deterministic logarithmic space. Since S is a recurrent $\{^{-}\}$-system, $\mathrm{M}_{ex}(\emptyset)$ simply reduces to $\mathrm{M}_{ex}(^{-})$. Let (S, b) be an instance of $\mathrm{M}_{ex}(^{-})$, $n =_{\mathrm{def}} \dim S$. The proof of Lemma 20 already shows that $b \in [S]$ if and only if $b \in S(t)$ for some $t \leq 3n$. In deterministic logarithmic space, it can be tested whether one of $(S, 0, b), (S, 1, b), \ldots, (S, 3n, b)$ is contained in $\mathrm{M}_{tm}(^{-})$.

Let (S, b) be an instance of $\mathrm{M}_{ex}(\cup)$ where $\mathcal{F} = \langle f_1, \ldots, f_n \rangle$, $n =_{\mathrm{def}} \dim S$. Using the definitions of the proof of Lemma 21, it holds that $b \in [S]$ if and only if $b \in S(t)$ for some $t < n$, i.e., there must be a path $(\nu_0, \ldots, \nu_t)$ for $t < n$. Otherwise, there are $i, j \in [0, n]$, $i < j$, such that $\nu_i = \nu_j$. Hence, $\mathrm{M}_{ex}(\cup) \in \mathrm{NL}$. Hardness of $\mathrm{M}_{ex}(\cup)$ follows directly from completeness of ACYC. ■

Alternatively, we could prove hardness of $\mathrm{M}_{ex}(\cup)$ for NL by reducing the graph accessibility problem for acyclic graphs to $\mathrm{M}_{ex}(\cup)$ in the same manner as it was done by McKenzie and Wagner. In a similar fashion, NL-completeness of $\mathrm{M}_{tm}(\cup)$ and $\mathrm{M}_{tm}(\cap)$ can be proved. Furthermore, since $\mathrm{M}_{ex}(\cup)$ reduces to $\mathrm{M}_{tm}(\cup)$, which is discussed at length at the end of Section 8, we have another reason for $\mathrm{M}_{tm}(\cup)$ being NL-complete.

The proof of Lemma 21 impressively shows that we can decide in nondeterministic logarithmic space whether $b \in S(t)$ for some recurrent $\{\cap\}$-system S and $t \geq 0$. But why does the described procedure not lead to an NL-algorithm also for $\mathrm{M}_{ex}(\cap)$? In fact, in Section 7 we will show that $\mathrm{M}_{ex}(\cap)$ is NP-complete. The answer is simple: we cannot bound number t by a small value as in the case of $\mathrm{M}_{ex}(\cup)$. Let $S = (\mathcal{F}, A)$, $n =_{\mathrm{def}} \dim S$, $\mathcal{F} = \langle f_1, \ldots, f_n \rangle$ and $A = (a_1, \ldots, a_n)$. Suppose that $b \in S(t)$. This means that for the first element ν_0 of **every** sequence $(\nu_0, \ldots, \nu_t)$ such that $\nu_t = n$ and $f_{\nu_{i+1}}$ depends on x_{ν_i}, $i \in [0, t{-}1]$, holds that $a_{\nu_0} = b$. Suppose that there are only $\sqrt[4]{n}$ different paths. Every path may have another cycle length δ (for explanations see the proof of Lemma 21). To find a small number t' such that $b \in S(t')$ and $t' < t$ it should be possible to shorten all paths by the same value. However, combinational arguments require a value of t exponential in n, which is not better than the trivial upper bound 2^n. Such combinational arguments are used in Section 8 to bound the number of evaluation steps of recurrent $\{\cup, \oplus, \otimes\}$-systems. Since $\mathrm{M}_{ex}(\cap)$ is NP-complete a considerable improvement can only be achieved by not computing every evaluation step between 0 and 2^n.

6. Recurrent systems with arithmetic operations and intersection. In this section we investigate several problems concerning recurrent $\{\cap, \oplus, \otimes\}$-systems and their complexities. A striking property of $\{\cap, \oplus, \otimes\}$-functions is that they can compute sets of cardinality at most 1. Since the result may also be empty, multiplication with 0 can be considered an emptyness test. And deciding emptyness will be the major problem to solve in this context. Note that for $\{\cup, \oplus, \otimes\}$-functions multiplication with 0 is nothing more than an accessibility problem since such functions cannot compute empty sets.

The main result of this section will be that $\mathrm{M}_{tm}(\cap, \oplus)$ is contained in P. We only want to mention that, even though $\mathrm{M}_{tm}(\cap, \oplus, \otimes)$ is trivially decidable since only finite sets are involved, we cannot give an interesting upper bound for the complexity of deciding $\mathrm{M}_{tm}(\cap, \otimes)$ that is different from the trivial bound by applying the results of McKenzie and Wagner, which shows $\mathrm{M}_{tm}(\cap, \otimes) \in \mathrm{EXP}$.

We first want to understand $\{\cap, \oplus, \otimes\}$-functions. The $\cap$-operator in this context can be regarded as an equivalence test. With this notion in mind we can rewrite $\{\cap, \oplus, \otimes\}$-functions by systems of $\{\oplus, \otimes\}$-functions in a specific sense that is given by the following lemma. Note that a $\{\oplus, \otimes\}$-function is a polynomial in several variables with natural coefficients. A $\{\oplus, \otimes\}$-function over $\mathbf{Z}$ is a polynomial in several variables with integer coefficiants. Since every set involved in this section is of cardinality at most 1, we will avoid cumbersome braces. The meaning of every term will always be clear from the context.

Lemma 23. *Let $\mathcal{O} \subseteq \{\oplus, \otimes\}$, and let $f = f(x_1, \ldots, x_k)$ be a k-ary $(\{\cap\} \cup \mathcal{O})$-function. Then, there are a k-ary $\mathcal{O}$-function f' and k-ary $\mathcal{O}$-functions $g_1, \ldots, g_r$ over $\mathbf{Z}$, $r \in \mathbf{N}$, such that, for all $\mathbf{a} \in \mathbf{N}^k$,*

$$\begin{aligned} g_i(\mathbf{a}) = 0 \text{ for all } i \in [1, r] \quad &\Longrightarrow \quad f(\mathbf{a}) = f'(\mathbf{a}) \\ g_i(\mathbf{a}) \neq 0 \text{ for some } i \in [1, r] \quad &\Longrightarrow \quad f(\mathbf{a}) = \emptyset\,. \end{aligned}$$

Proof: We prove the claim by induction. If $f(x_1, \ldots, x_k) =_{\mathrm{def}} x_j$ for $j \in [1, k]$ then $f' =_{\mathrm{def}} f$ and $r =_{\mathrm{def}} 0$. Let $f = f_1 \cap f_2$, $f = f_1 \oplus f_2$ or $f = f_1 \otimes f_2$. By induction hypothesis there are f_1', f_2', $g_1^1, \ldots, g_{r_1}^1$ and $g_1^2, \ldots, g_{r_2}^2$, $r_1, r_2 \geq 0$, such that, for all $\mathbf{a} \in \mathbf{N}^k$,

$$\begin{aligned} g_i^1(\mathbf{a}) = 0 \text{ for all } i \in [1, r_1] \quad &\Longrightarrow \quad f_1(\mathbf{a}) = f_1'(\mathbf{a}) \\ g_i^2(\mathbf{a}) = 0 \text{ for all } i \in [1, r_2] \quad &\Longrightarrow \quad f_2(\mathbf{a}) = f_2'(\mathbf{a}) \\ g_i^1(\mathbf{a}) \neq 0 \text{ for some } i \in [1, r_1] \quad &\Longrightarrow \quad f_1(\mathbf{a}) = \emptyset \\ g_i^2(\mathbf{a}) \neq 0 \text{ for some } i \in [1, r_2] \quad &\Longrightarrow \quad f_2(\mathbf{a}) = \emptyset\,. \end{aligned}$$

If $f = f_1 \oplus f_2$ or $f = f_1 \otimes f_2$ then f has the claimed property using $f_1' \oplus f_2'$ or $f_1' \otimes f_2'$, respectively, and $g_1^1, \ldots, g_{r_1}^1, g_1^2, \ldots, g_{r_2}^2$. Let $f = f_1 \cap f_2$. Let $f' =_{\mathrm{def}} f_1'$, $g_i =_{\mathrm{def}} g_i^1$ for $i \in [1, r_1]$, $g_{r_1+i} =_{\mathrm{def}} g_i^2$ for $i \in [1, r_2]$ and $g_r =_{\mathrm{def}} f_1' - f_2'$, $r =_{\mathrm{def}} r_1 + r_2 + 1$. Let $g_i(\mathbf{a}) = 0$

for $\mathbf{a} \in \mathbf{N}^k$ and all $i \in [1, r]$. Then $f_1(\mathbf{a}) \neq \emptyset$, $f_2(\mathbf{a}) \neq \emptyset$, $f_1(\mathbf{a}) = f_1'(\mathbf{a})$, $f_2(\mathbf{a}) = f_2'(\mathbf{a})$ and $f_1(\mathbf{a}) = f_2(\mathbf{a}) = f'(\mathbf{a})$. If there is $i \in [1, r]$ such that $g_i(\mathbf{a}) \neq 0$ for $\mathbf{a} \in \mathbf{N}^k$ then $f_1(\mathbf{a}) = \emptyset$ or $f_2(\mathbf{a}) = \emptyset$ or $f_1(\mathbf{a}) \neq f_2(\mathbf{a})$. Hence, $f(\mathbf{a}) = \emptyset$. ■

Since we assume functions to be represented by arithmetic circuits, Lemma 23 provides a simple P-algorithm that reduces $\{\cap, \oplus, \otimes\}$-functions to $\{\oplus, \otimes\}$-functions by simply deleting all $\cap$-vertices in the circuits. Precisely, if v is a $\cap$-vertex in such a circuit then we delete v and connect one predecessor of v to all successors of v. Finally, all unnecessary vertices are deleted.

Corollary 24. *Let $\mathcal{O} \subseteq \{\oplus, \otimes\}$. Let $S = (\mathcal{F}, A)$ be a recurrent $(\{\cap\} \cup \mathcal{O})$-system. Then, there is a recurrent $\mathcal{O}$-system $S' = (\mathcal{F}', A)$ such that for every $t \in \mathbf{N}$, $S(t) \neq \emptyset$ implies $S(t) = S'(t)$, and S' can be computed by an* FP*-function.*

Proof: Let $n =_{\mathrm{def}} \dim S$ and $\mathcal{F} = \langle f_1, \ldots, f_n \rangle$. Let $\mathcal{F}' =_{\mathrm{def}} \langle f_1', \ldots, f_n' \rangle$ where f_i', $i \in [1, n]$, is the function corresponding to f_i according to Lemma 23. Induction over t shows that $f_i(t) \neq \emptyset$ implies $f_i(t) = f_i'(t)$, $i \in [1, n]$. ■

It is a well-known fact that integer multiplication can be reduced to componentwise addition of vectors. McKenzie and Wagner applied this idea to reduce $\mathrm{MC}(\mathcal{O} \cup \{\otimes\})$ to $\mathrm{MC}(\mathcal{O} \cup \{\oplus\})$ for all $\{\mathcal{O}\} \subseteq \{\cup, \cap, ^-\}$ [6]. As intermediate problems circuits operating on vectors over $\mathbf{N}$ were used. Since the finite character of circuits permits to bound the results, vectors could be regarded as natural numbers represented in a k-ary number system for a suitable very large k (which, in fact, is determined by the largest number that could be computed by such a circuit). The length of k can be bounded by the input length, i.e., by the number of vertices of the circuit and the length of the input numbers.

We would also like to reduce exact membership problems with arithmetic operation only $\otimes$ to exact membership problems with arithmetic operation only $\oplus$. However, the above sketched method does not work. Even though the concerned circuits may be considered finite, the numbers that can be computed may be of exponential length in the length of the input. Hence, number k as the base of the representing number system should be of exponential length, which cannot be realised by a polynomial-time reduction. However, in case of special recurrent $\{\cap, \otimes\}$-systems, we can apply a slightly modified idea. Let $\mathrm{M}_{tm}^{+}(\cap, \otimes)$ denote the set of all triples $(S, t, b) \in \mathrm{M}_{tm}(\cap, \otimes)$ where $b > 0$. Remember that two numbers p and q are relative prime, if their greatest common devisor is 1. A *gcd-base* $\mathcal{B}$ for $\mathcal{D} \subseteq \mathbf{N}$ is a set of pairwise relative prime numbers greater than 1 such that every number greater than 1 of $\mathcal{D}$ is the product of numbers from $\mathcal{B}$. Furthermore, for given set $\mathcal{D}$, a gcd-base $\mathcal{B}$ for $\mathcal{D}$ can be computed in polynomial time, and $\mathcal{B}$ then is only of polynomial size.

Lemma 25. $\mathrm{M}_{tm}^{+}(\cap, \otimes) \leq_m^{\mathrm{P}} \mathrm{M}_{tm}(\cap, \oplus)$.

Proof: Let (S, t, b) be an instance of $\mathrm{M}_{tm}^{+}(\cap, \otimes)$ where $S = (\mathcal{F}, A)$, $\mathcal{F} = \langle f_1, \ldots, f_n \rangle$, $n =_{\mathrm{def}} \dim S$, and $A = (a_1, \ldots, a_n)$. Let $\mathcal{B} =_{\mathrm{def}} \{q_1, \ldots, q_\ell\}$ be a gcd-base for $\{a_1, \ldots, a_n, b\}$. We define recurrent $\{\cap, \oplus\}$-systems $S_1, \ldots, S_\ell$, $S_i =_{\mathrm{def}} (\mathcal{F}_i, A_i)$, $i \in [1, \ell]$ as follows. Let $\mathcal{F}_i =_{\mathrm{def}} \langle f_1', \ldots, f_n' \rangle$ where f_j' emerges from f_j by replacing each $\otimes$ in f_j by $\oplus$, $j \in [1, n]$. For $i \in [1, \ell]$, $A_i =_{\mathrm{def}} (a_1^i, \ldots, a_n^i)$ where a_j^i is the largest number c such that $(q_i)^c$ is a divisor of a_j, if $a_j > 0$; if $a_j = 0$, a_j^i is the smallest number c such that $(q_i)^c$ is not a divisor of b. Then, it holds for every $i \in [1, \ell]$ and every $t \in \mathbf{N}$ that $S_i(t) = c$ is the largest number such that $(q_i)^c$ is a divisor of $S(t)$, if $S(t) \neq 0$ and $S(t) \neq \emptyset$. Let r_i, $i \in [1, \ell]$, be the largest number c such that $(q_i)^c$ is a divisor of b. Hence, $S(t) = b$ if and only if

$S_i(t) = r_i$ for all $i \in [1, \ell]$. We define the following functions: $f_j^i(\mathbf{x}) =_{\mathrm{def}} f_j'(\mathbf{x}_i)$, $i \in [1, \ell]$, $j \in [1, n]$, where $\mathbf{x} =_{\mathrm{def}} (x_1, \ldots, x_{\ell \cdot (n+1)+1})$ and $\mathbf{x}_i =_{\mathrm{def}} (x_{in-n+1}, \ldots, x_{in})$. Furthermore, let $\hat{f}_i(\mathbf{x}) =_{\mathrm{def}} x_{\ell n+i}$, $i \in [1, \ell]$, and $\hat{f}(\mathbf{x}) =_{\mathrm{def}} \sum_{i=1}^{\ell} (x_{in} \cap x_{\ell n+i})$. Finally, we define a recurrent $\{\cap, +\}$-system $S' =_{\mathrm{def}} (\mathcal{F}', A')$ where

$$\begin{aligned}\mathcal{F}' &=_{\mathrm{def}} \langle f_1^1, \ldots, f_n^1, f_1^2, \ldots, f_n^\ell, \hat{f}_1, \ldots, \hat{f}_\ell, \hat{f} \rangle \\ A' &=_{\mathrm{def}} (a_1^1, \ldots, a_n^1, a_1^2, \ldots, a_n^\ell, r_1, \ldots, r_\ell, \hat{r})\end{aligned}$$

and $\hat{r}$ is the smallest number that is not equal to $b' =_{\mathrm{def}} r_1 + \ldots + r_\ell$. With these definitions it holds that $S(t) = b$ if and only if $S'(t+1) \neq \emptyset$ if and only if $S'(t+1) = b'$. Then, $(S, t, b) \in \mathrm{M}_{tm}^{+}(\cap, \otimes)$ if and only if $(S', t+1, b') \in \mathrm{M}_{tm}(\cap, \oplus)$. The described reduction can be performed in polynomial time. ∎

The described construction fails if we want to reduce an instance $(S, t, 0)$ of $\mathrm{M}_{tm}(\cap, \otimes)$ to some instance of $\mathrm{M}_{tm}(\cap, \oplus)$. Multiplication with 0 just gives 0, and since addition is monotonic, it is not clear how to model such a behaviour.

In the following part of the section, we will show that $\mathrm{M}_{tm}(\cap, \oplus)$ can be decided in polynomial time. This implies that $\mathrm{M}_{tm}(\oplus)$ and $\mathrm{M}_{tm}(\otimes)$ can also be decided in polynomial time. In the following section, the corresponding existential membership problems turn out to be NP-complete. Let $f = f(x_1, \ldots, x_n)$ be a $\{\oplus\}$-function that is represented by circuit C. Let $a_1, \ldots, a_n \in \mathbf{N}$, and let ν be the number of vertices in C. Then, $e \leq 2^\nu \cdot \max\{a_1, \ldots, a_n\}$ where $\{e\} = f(\{a_1\}, \ldots, \{a_n\})$.

Proposition 26. $\mathrm{M}_{tm}(\oplus)$ *is in* P.

Proof: Let (S, t, b) be an instance of $\mathrm{M}_{tm}(\oplus)$ where $S = (\mathcal{F}, A)$ and $\mathcal{F} = \langle f_1, \ldots, f_n \rangle$, $n =_{\mathrm{def}} \dim S$. For each $i \in [1, n]$, it holds that $f_i(\mathbf{x}) = \sum_{j=1}^{n} c_j^i \cdot x_j$, $\mathbf{x} =_{\mathrm{def}} (x_1, \ldots, x_n)$, for $c_j^i \in \mathbf{N}$, $j \in [1, n]$. Since f_i is given as an arithmetic circuit C_i, c_j^i is equal to the number of paths from the input vertex in C_i with label j to the output vertex of C_i. This number can be computed in polynomial time by a straightforward strategy: if u' and u'' are the predecessors of u, then the number of paths from the input vertex with label j to u is the sum of those numbers for u' and u''. We define an $n \times n$-matrix M over the semiring $\mathrm{SR}(b)$ as follows. For all $i, j \in [1, n]$, $M_{ij} =_{\mathrm{def}} \alpha_b(c_j^i)$. Using this definition it holds for every $t' \in \mathbf{N}$ that $\alpha_b(S(t'+1)) = \alpha_b(f_n(M^{t'} A))$. We observe that $M^{2k} = M^k \cdot M^k$ and $M^{2k+1} = M^k \cdot M^k \cdot M$. Hence, we can compute M^t using at most $2 \cdot \lceil \log t \rceil$ matrix multiplications. In order to achieve polynomial time we calculate M^t in $\mathrm{SR}(b)$. ∎

It is time to consider an easy but interesting example: the Fibonacci numbers. We define a recurrent $\{\oplus\}$-system $\mathrm{Fib} = (\mathcal{F}_{\mathrm{Fib}}, A_{\mathrm{Fib}})$ as follows:

$$\begin{aligned}\mathcal{F}_{\mathrm{Fib}} &=_{\mathrm{def}} \langle f_1(x_1, x_2) =_{\mathrm{def}} x_2, f_2(x_1, x_2) =_{\mathrm{def}} x_1 \oplus x_2 \rangle \\ A_{\mathrm{Fib}} &=_{\mathrm{def}} (0, 1)\,.\end{aligned}$$

We compute the first three values of Fib: $\mathrm{Fib}(0) = 1$, $\mathrm{Fib}(1) = f_2(0, 1) = 1$, $\mathrm{Fib}(2) = f_2(f_1(0, 1), f_2(0, 1)) = f_2(1, 1) = 2$. Indeed, we obtain the Fibonacci numbers. The proof of Proposition 26 shows that we can generate this sequence by matrix multiplication. Let

$$M =_{\mathrm{def}} \begin{pmatrix} 0 & 1 \\ 1 & 1 \end{pmatrix}$$

and $A =_{\mathrm{def}} (0, 1)$. Then, the $(k+1)$-th Fibonacci number (where the second is 2) is equal to the last component of $M^k A$:

$$M^0 A = \begin{pmatrix} 0 \\ 1 \end{pmatrix}, \quad M^1 A = \begin{pmatrix} 0 & 1 \\ 1 & 1 \end{pmatrix} \cdot \begin{pmatrix} 0 \\ 1 \end{pmatrix} = \begin{pmatrix} 1 \\ 1 \end{pmatrix}, \quad M^2 A = \begin{pmatrix} 1 & 1 \\ 1 & 2 \end{pmatrix} \cdot \begin{pmatrix} 0 \\ 1 \end{pmatrix} = \begin{pmatrix} 1 \\ 2 \end{pmatrix}.$$

Corollary 24 determines one way to decide $\mathrm{M}_{tm}(\cap,\oplus)$. For a given recurrent $\{\cap,\oplus\}$-system S, we define S' as given by Corollary 24. If $S(t)\neq\emptyset$ for $t\geq 0$, then $S(t)=S'(t)$. This case is solved by Proposition 26. It remains to decide whether $S(t)=\emptyset$. Let $\mathcal{O}\subseteq\{\cup,\cap,^{-},\oplus,\otimes\}$. The problem $\textsc{Empty}(\mathcal{O})$ is the set of all pairs (S,t) where S is a recurrent $\mathcal{O}$-system and $t\in\mathbf{N}$ such that $S(t)=\emptyset$.

We do not know how to decide $\textsc{Empty}(\cap,\oplus)$ in polynomial time. However, for deciding $\mathrm{M}_{tm}(\cap,\oplus)$ in polynomial time it suffices to consider only a subset of $\textsc{Empty}(\cap,\oplus)$. Let $G=(V,A)$ be a directed graph. A *cycle* of length k in G is a sequence $C=(x_1,\ldots,x_k)$ of vertices of G such that C is an x_1,x_k-path in G and $(x_k,x_1)\in A$. Let $S=(\mathcal{F},A)$ be a recurrent $\{\cap,\oplus\}$-system, $\mathcal{F}=\langle f_1,\ldots,f_n\rangle$, $n=_{\mathrm{def}}\dim S$, and obtain S' from S by replacing every $\cap$ in the functions of S by $\oplus$. Let M' be the matrix for S' as it is defined in the proof of Proposition 26. We can interpret M' as the adjacency matrix of a directed graph. Let $G_S=([1,n],A_S)$ be the graph on n vertices where $(i,j)\in A_S$ if and only if $M'_{ij}\neq 0$. Equivalently, $(i,j)\in A_S$ if and only if f_i depends on the j-th input. Function f_i appears in a cycle of length k, if there is a cycle of length k in G_S containing vertex i. The problem $\circ$-$\textsc{Empty}(\cap,\oplus)$ is the set of all pairs (S,t) in $\textsc{Empty}(\cap,\oplus)$ where the output function of S is contained in a cycle.

First, we show that $\circ$-$\textsc{Empty}(\cap,\oplus)$ can be decided in polynomial time, which is heavily based on a lemma of linear algebra. Let L be a $1\times n$-matrix over $\mathbf{Z}$, $n\geq 1$. Let $[L]=_{\mathrm{def}}\{\mathbf{x}\in\mathbf{R}^n : L\mathbf{x}=0\}$ denote the hyperplane of $\mathbf{R}^n$ defined by L.

Lemma 27. *Let M be an $n\times n$-matrix over $\mathbf{N}$ and L be a $1\times n$-matrix over $\mathbf{Z}$, and let $\mathbf{a}\in\mathbf{N}$. If $M^i\mathbf{a}\in[L]$ for all $i\in[0,n-1]$ then $M^i\mathbf{a}\in[L]$ for all $i\geq 0$.*

Proof: Let $k\leq n-1$ be smallest possible such that $M^{k+1}\mathbf{a}=\sum_{i=0}^{k}c_i\cdot M^i\mathbf{a}$ for some $c_0,\ldots,c_k\in\mathbf{R}$. We show by induction that for every $r'\geq k+1$ there are $x_0,\ldots,x_k\in\mathbf{R}$ such that $M^{r'}\mathbf{a}=\sum_{i=0}^{k}x_i\cdot M^i\mathbf{a}$. Let $x_0,\ldots,x_k\in\mathbf{R}$ such that $M^r\mathbf{a}=\sum_{i=0}^{k}x_i\cdot M^i\mathbf{a}$ for $r\geq k+1$. One easily verifies that $M^{r+1}\mathbf{a}=\sum_{i=1}^{k}(x_{i-1}+x_kc_i)\cdot M^i\mathbf{a}+x_kc_0\cdot\mathbf{a}$. By assumption, $M^i\mathbf{a}\in[L]$, $i\leq k$. It holds then that $M^r\mathbf{a}\in[L]$ for every $r\in\mathbf{N}$. ■

Proposition 28. $\circ$-$\textsc{Empty}(\cap,\oplus)$ *is in* P.

Proof: Let (S,t) be an instance of $\circ$-$\textsc{Empty}(\cap,\oplus)$ where $S=(\mathcal{F},A)$ and $\mathcal{F}=\langle f_1,\ldots,f_n\rangle$, $n=_{\mathrm{def}}\dim S$. We apply Lemma 23 and obtain $f'_1,\ldots,f'_n$ and $g^1_1,\ldots,g^1_{r_1},g^2_1,\ldots,g^n_{r_n}$ such that, for every $i\in[1,n]$ and every $\mathbf{a}\in\mathbf{N}^n$,

$$\begin{aligned} g^i_j(\mathbf{a})=0 \text{ for all } j\in[1,r_i] &\quad\Longrightarrow\quad f_i(\mathbf{a})=f'_i(\mathbf{a})\\ g^i_j(\mathbf{a})\neq 0 \text{ for some } j\in[1,r_i] &\quad\Longrightarrow\quad f_i(\mathbf{a})=\emptyset\,. \end{aligned}$$

Let M be the $n\times n$-matrix that is defined by $\langle f'_1,\ldots,f'_n\rangle$ in the way described in the proof of Proposition 26. For $i\in[1,n]$ and $t\in\mathbf{N}$, $f_i(t+1)=\emptyset$ if and only if $f'_i(\mathcal{F}(t))=\emptyset$ or there is $j\in[1,r_i]$ such that $g^i_j(\mathcal{F}(t))\neq 0$. Let $k\leq n$ be the smallest number such that f_n appears in a cycle of length k. If there is $t'\leq 2n^2$ such that $S(t')=\emptyset$ and $t'\equiv t \pmod k$, then $S(t)=\emptyset$. We will show that this claim can be strengthened to a characterisation. Let $t>2n^2$, and let $S(t)=\emptyset$. Let $P=_{\mathrm{def}}(\nu_0,\nu_1,\ldots,\nu_r)$ be a longest path in G_S such that $\nu_0=n$ and $f_{\nu_i}(t-i)=\emptyset$, $i\in[0,r]$. Note that f_{ν_r} does not depend on a variable x_i such that $f_i(t-r-1)=\emptyset$. We might say that $f_{\nu_r}(t-r)=\emptyset$ *causes* $S(t)=\emptyset$, which is proven by path P. Consider the subsequence $P'=_{\mathrm{def}}(\nu_0,\nu_k,\ldots,\nu_{s\cdot k})$ of P where $s\cdot k\leq r<(s+1)\cdot k$. If a number appears twice in P', say ν_j and $\nu_{j'}$, $j<j'$, then $P''=_{\mathrm{def}}(\nu_0,\nu_1,\ldots,\nu_j,\nu_{j'+1},\ldots,\nu_r)$ proves that $f_n(t-(j'-j))=\emptyset$. Note that $t\equiv t-(j'-j) \pmod k$. We apply this argument repeatedly and obtain that, if $f_{\nu_r}(t')=\emptyset$ for $t'\in\mathbf{N}$, then $S(t'')=\emptyset$ for $t'+kn-k<t''\leq$

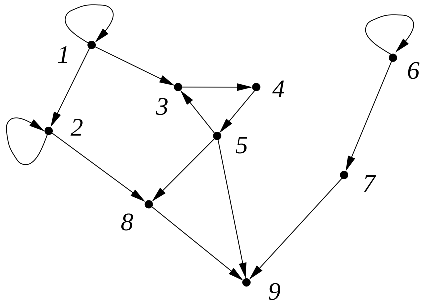

Figure 4. Graph G_S for some sample recurrent system S.

$t' + kn \leq t' + n^2$ and $t'' \equiv t' + r \pmod{k}$. Let $t_0 \in [1, k]$ such that $t_0 \equiv t \pmod{k}$. Then, for all $i \in \mathbf{N}$ such that $t_0 + ik < t - r$ holds that $f_{\nu_r}(t_0 + ik) \neq \emptyset$ by the maximality of P. Let $T =_{\text{def}} M^k$ and $A_{t_0} =_{\text{def}} M^{t_0-1}A$. Due to Lemma 23, if $f_{\nu_r}(t_0) \neq \emptyset$ then $f_{\nu_r}(t_0) = f'_{\nu_r}(A_{t_0})$. For $i < n$, if $f_{\nu_r}(t_0 + ik) \neq \emptyset$ then $f_{\nu_r}(t_0 + ik) = f'_{\nu_r}(T^i A_{t_0})$. Furthermore, $f_{\nu_r}(t_0 + ik) \neq \emptyset$ if and only if $g_j^{\nu_r}(T^i A_{t_0}) = 0$ for all $j \in [1, r_{\nu_r}]$. We apply Lemma 27 and obtain that $f_{\nu_r}(t_0 + (n-1) \cdot k) \neq \emptyset$ if and only if $g_j^{\nu_r}(T^i A_{t_0}) = 0$ for all $i \in [0, n-1]$ and $j \in [1, r_{\nu_r}]$. Since $f_{\nu_r}(t-r) = \emptyset$, there is $i < n$ such that $f_{\nu_r}(t_0 + ik) = \emptyset$, where $t_0 + ik \leq k + (n-1) \cdot k \leq n^2$. Hence, if $S(t) = \emptyset$ then there is $t' \leq 2n^2$ such that $S(t') = \emptyset$ and $t \equiv t' \pmod{k}$. It takes polynomial time to compute $\mathcal{F}(0), \ldots, \mathcal{F}(2n^2)$, which concludes the proof. ■

Before we finish the proof of our main result of this section, we want to understand the construction of the final part. Consider Figure 4. It is shown graph G_S of a 9-dimensional recurrent $\{\cap, \oplus\}$-system. For instance, function f_1 depends only on x_1, whereas function f_2 depends on x_1 and x_2. Proposition 28 shows that we can decide for every $t \geq 0$ in polynomial time whether $f_2(t) = \emptyset$ or whether $f_5(t) = \emptyset$. We want to decide whether $f_9(20) = 8$. Then, it must hold that $f_2(18) \neq \emptyset$, $f_5(18) \neq \emptyset$, $f_5(19) \neq \emptyset$ and $f_6(18) \neq \emptyset$. Furthermore, $f_2(18)$, $f_5(18)$, $f_5(19)$ and $f_6(18)$ must all have value not greater than 8. Applying the method of Corollary 24 and Proposition 26, we can calculate in polynomial time these values or decide that some of them exceeds 8. Finally, we can straightforwardly evaluate $f_7(19)$, $f_8(19)$ and $f_9(20)$ in polynomial time.

Theorem 29. $\mathrm{M}_{tm}(\cap, \oplus)$ *is in* P.

Proof: Let (S, t, b) be an instance of $\mathrm{M}_{tm}(\cap, \oplus)$ where $S = (\mathcal{F}, A)$ and $\mathcal{F} = \langle f_1, \ldots, f_n \rangle$, $n =_{\text{def}} \dim S$. Let $f'_1, \ldots, f'_n$ be the functions that we obtain by application of Lemma 23. Let $\mathcal{F}' =_{\text{def}} \langle f'_1, \ldots, f'_n \rangle$, and let $S' =_{\text{def}} (\mathcal{F}', A)$. Due to Lemma 23, if $S(t) \neq \emptyset$ then $S'(t) = S(t)$. If the output function f_n of S is contained in a cycle then $(S, t, b) \in \mathrm{M}_{tm}(\cap, \oplus)$ if and only if $(S', t, b) \in \mathrm{M}_{tm}(\oplus)$ and $(S, t) \notin \circ\text{-}\textsc{Empty}(\cap, \oplus)$, which can be verified in polynomial time.

Let f_n be not contained in a cycle. If $t < n$ $S(t)$ can be computed straightforwardly. Let $t \geq n$. Consider G_S. Since vertex n does not appear in cycle in G_S there is a maximal connected subgraph $G^{[n]}$ of G_S that contains n and that does not contain a vertex that appears in a cycle. (In Figure 4 this subgraph is induced by the vertex set $\{7, 8, 9\}$.) It holds that every vertex in $G^{[n]}$ has a predecessor, and all those predecessors that do not occur in $G^{[n]}$ are contained in cycles in G_S. Let N be the set of predecessors of vertices in $G^{[n]}$ that are not contained in $G^{[n]}$. (In Figure 4 this is the set $\{2, 5, 6\}$.) $G^{[n]}$ is acyclic, and therefore every path in $G^{[n]}$ contains less than n vertices. If we know all values $f_i(t')$ where $i \in N$ and $t - n < t' \leq t$ we can compute $f_n(t)$. However, for large t these values can become large. It suffices though to calculate the values in the semiring $\mathrm{SR}(b)$, since it does not matter whether $f_n(t) = \emptyset$ or $f_n(t) > b$, if $f_n(t) \neq b$. Let S' the recurrent system that emerges from S by application of Corollary 24. The proof of Proposition 26 shows that we can calculate in polynomial time $\mathcal{F}'(t-n+1), \dots, \mathcal{F}'(t)$ in $\mathrm{SR}(b)$, where $\mathcal{F}'$ belongs to S' in the obvious sense. For every $i \in N$ and every $t' \in [t-n+1, t]$ it can be verified whether $(S_i, t') \in \circ\text{-}\textsc{Empty}(\cap, \oplus)$ where S_i emerges from S by renaming such that f_i is output function. Compute a topological ordering of the vertices of G_S. According to this ordering, compute for every function f_j whose corresponding vertex j belongs to $G^{[n]}$ the values $f_j(t-n+c), \dots, f_j(t)$ where c is the length of the longest path from n to j in G_S (remember that G_S is acyclic). All this can be done in polynomial time. Accept if and only if $f_n(t) = b$. ■

Corollary 30. $\mathrm{M}_{tm}(\otimes)$ *and* $\mathrm{M}^{+}_{tm}(\cap, \otimes)$ *are in* P.

Proof: Let (S, t, b) be an instance of $\mathrm{M}_{tm}(\otimes)$ where $S = (\mathcal{F}, A)$. If $b = 0$ then obtain S' by replacing every $\otimes$ in the functions of $\mathcal{F}$ by $\cup$. We observe that $0 \in S(t)$ if and only if $0 \in S'(t)$. If $b > 0$ then $(S, t, b) \in \mathrm{M}_{tm}(\otimes)$ if and only if $(S, t, b) \in \mathrm{M}^{+}_{tm}(\cap, \otimes)$. ■

McKenzie and Wagner showed that $\mathrm{MC}(\oplus)$, $\mathrm{MC}(\cap, \oplus)$ and $\mathrm{MC}(\cap, \otimes)$ are $\mathrm{C_{=}L}$-hard, so that $\mathrm{M}_{tm}(\oplus)$, $\mathrm{M}_{tm}(\cap, \oplus)$ and $\mathrm{M}_{tm}(\cap, \otimes)$ are $\mathrm{C_{=}L}$-hard. Their $\mathrm{MC}(\otimes)$ problem, however, is NL-complete.

7. NP**-complete membership problems.** In the last sections we studied, among others, the exact membership problems for recurrent $\{\cap\}$-, $\{\oplus\}$-, $\{\otimes\}$- and $\{\cap, \oplus\}$-systems. All these problems can be decided in polynomial time. We will show in this section that the corresponding existential membership problems are NP-complete. Containedness in NP of all these problems is given by a rather formal argument. To show hardness we define a new problem that turns out to be NP-complete and that we reduce to $\mathrm{M}_{ex}(\cap)$. This new problem can be considered a generalization of the Chinese Remainder Theorem. The Chinese Remainder Theorem shows that a system of congruence equations where the moduli are pairwise relatively prime numbers has a solution that is unique in a determined interval of natural numbers. We extend this problem with respect to two aspects. Moduli are arbitrary numbers, and for each modul we find a set of congruence equations. A solution of this *Set-system of congruence equations* fulfills one equation from each set. Formally, we define the problem Set-SCE as follows.

Solution of a Set-System of Congruence Equations (Set-SCE).
Instance. $((A_1, b_1), \dots, (A_k, b_k))$ where $A_1, \dots, A_k$ are finite sets of natural numbers, and $b_1, \dots, b_k$ are natural numbers greater than 1 represented unarily.
Question. Are there $n \in \mathbf{N}$ and $a_1 \in A_1, \dots, a_k \in A_k$ such that $n \equiv a_i \pmod{b_i}$ for all $i \in [1, k]$?

Note that it is not important to require binary representation of the numbers in $A_1, \dots, A_k$. However, we assume a binary representation of them, only to fix a system.

Lemma 31. Set-SCE *is* NP*-hard.*

Proof: We reduce 3-Sat to Set-SCE. Let $H =_{\mathrm{def}} H(x_1, \dots, x_r)$ be a Boolean formula in 3-CNF, and no variable appears twice in a clause. Let k be the number of clauses of H, and let $K_1, \dots, K_k$ be the clauses of H. Let $p_1, \dots, p_r$ be r prime numbers. We define for every $j \in [1, k]$:

$$b_j =_{\mathrm{def}} \prod_{i=1}^{r} \begin{cases} p_i & \text{, if } x_i \text{ is variable in } K_j \\ 1 & \text{, if } x_i \text{ is not variable in } K_j\,. \end{cases}$$

Variables in each clause are ordered according to their indices. Let $f_j = f_j(z_1, z_2, z_3)$ be the Boolean function defined by K_j. Let $x^{(1)}, x^{(2)}, x^{(3)}$ be the variables in K_j and $p^{(1)}, p^{(2)}, p^{(3)}$ the corresponding primes. Let N_j be the set of all natural numbers a smaller than b_j such that, for all $i \in [1,3]$, $a \equiv 0 \pmod{p^{(i)}}$ or $a \equiv 1 \pmod{p^{(i)}}$. Note that, by the Chinese Remainder Theorem, N_j contains exactly eight numbers, among them 0 and 1. We define A_j as the set of all numbers $a \in N_j$ such that $f_j(c^{(1)}, c^{(2)}, c^{(3)}) = 1$ where

$$c^{(i)} =_{\text{def}} \begin{cases} 1 & \text{, if } a \equiv 0 \pmod{p^{(i)}} \\ 0 & \text{, if } a \equiv 1 \pmod{p^{(i)}} \end{cases}$$

and $i \in [1,3]$. Let β be a satisfying assignment for H. Due to the Chinese Remainder Theorem there is a number $n < p_1 \cdot \ldots \cdot p_r$ that satisfies the congruence equations

$$n \equiv 1 - \beta(x_i) \pmod{p_i}$$

for all $i \in [1,r]$. Let $j \in [1,k]$, and let $p^{(1)}, p^{(2)}, p^{(3)}$ correspond to the variables $x^{(1)}, x^{(2)}, x^{(3)}$ in K_j. Let $a < b_j$ satisfy

$$a \equiv 1 - \beta(x^{(i)}) \pmod{p^{(i)}}$$

for every $i \in [1,3]$. Observe that $a \in A_j$. Since the remainders of a and n correspond on $p^{(1)}$, $p^{(2)}$ and $p^{(3)}$, $n \equiv a \pmod{b_j}$ due to the Chinese Remainder Theorem. Hence, n is a solution of $\mathcal{S}_H =_{\text{def}} ((A_1, b_1), \ldots, (A_k, b_k))$. Now, let n be a solution of $\mathcal{S}_H$. We will show that β, defined as

$$\beta(x_i) =_{\text{def}} \begin{cases} 1 & \text{, if } n \equiv 0 \pmod{p_i} \\ 0 & \text{, if } n \not\equiv 0 \pmod{p_i} \end{cases}$$

for all $i \in [1,r]$, satisfies H. Let $a \in A_j$ such that $n \equiv a \pmod{b_j}$, and let x_i be a variable in K_j such that $\ell \in \{x_i, \neg x_i\}$ is literal in K_j and $\ell = x_i$ if and only if $a \equiv 0 \pmod{p_i}$. Then, $\ell = x_i$ if and only if $n \equiv 0 \pmod{p_i}$ by the Chinese Remainder Theorem. Therefore, K_j is satisfied by β.

It remains to show that the reduction can be carried out in logarithmic space. The interval $[2, r^2]$ contains r primes. The sets A_i and the numbers b_i, $i \in [1,k]$, can be constructed by straightforward enumeration and verification in logarithmic space. ■

Let $\mathrm{M}^{+}_{ex}(\cap, \otimes)$ denote the set of all pairs (S, b) in $\mathrm{M}_{ex}(\cap, \otimes)$ where $b > 0$.

Theorem 32. *$\mathrm{M}_{ex}(\cap)$, $\mathrm{M}_{ex}(\oplus)$, $\mathrm{M}_{ex}(\otimes)$, $\mathrm{M}_{ex}(\cap, \oplus)$ and $\mathrm{M}^{+}_{ex}(\cap, \otimes)$ are* NP*-complete.*

Proof: We first show that SET-SCE reduces to $\mathrm{M}_{ex}(\cap)$. Let $\mathcal{S} =_{\text{def}} ((A_1, b_1), \ldots, (A_k, b_k))$ be an instance of SET-SCE. We assume that A_i only contains numbers that are smaller than b_i; otherwise find appropriate sets A_i'. We define a recurrent $\{\cap\}$-system $S =_{\text{def}} (\mathcal{F}, A)$ as follows. For every $i \in [1,k]$, for every $j \in [1, b_i - 1]$ we define

$$f_j^{(i)}(\mathbf{x}) =_{\text{def}} x_{j-1}^{(i)} \quad \text{and} \quad f_0^{(i)}(\mathbf{x}) =_{\text{def}} x_{b_i - 1}^{(i)}$$

where $\mathbf{x} =_{\text{def}} (x_0^{(1)}, \ldots, x_{b_1-1}^{(1)}, x_0^{(2)}, \ldots, x_{b_k-1}^{(k)}, x')$. Let $A =_{\text{def}} (c_0^{(1)}, \ldots, c_{b_k-1}^{(k)}, 0)$ where $c_j^{(i)} \in \{0,1\}$ and $c_j^{(i)} = 1$ if and only if $j \in A_i$. Furthermore, let $f'(\mathbf{x}) =_{\text{def}} x_0^{(1)} \cap \ldots \cap x_0^{(k)}$ and $\mathcal{F} =_{\text{def}} \langle f_0^{(1)}, \ldots, f_{b_k-1}^{(k)}, f' \rangle$. It holds that $f_j^{(i)}(t) = 1$ if and only if $c_\nu^{(i)} = 1$ for $\nu < b_i$ and $\nu \equiv t - j \pmod{b_i}$. Hence, $(S, 1) \in \mathrm{M}_{ex}(\cap)$ if and only if $\mathcal{S} \in$ SET-SCE. An FL-function can perform this reduction since the numbers b_i are given in unary representation. By Lemma 31, $\mathrm{M}_{ex}(\cap)$ is NP-hard.

Now, let (S, b) be an instance of $\mathrm{M}_{ex}(\cap)$ where $S = (\mathcal{F}, A)$. We define recurrent $\{\oplus\}$- and $\{\otimes\}$-systems S' and S'' as follows. Let $\mathcal{F}'$ and $\mathcal{F}''$ emerge from $\mathcal{F}$ by replacing each occurrence of $\cap$ by $\oplus$ and $\otimes$, respectively. Let A' emerge from A by replacing b by 0 and all other numbers by 1. Similarly, obtain A'' by replacing b by 1 and all other numbers by 0. Let $S' =_{\text{def}} (\mathcal{F}', A')$ and $S'' =_{\text{def}} (\mathcal{F}'', A'')$. Then, $(S, b) \in \mathrm{M}_{ex}(\cap)$ if and only if $(S', 0) \in \mathrm{M}_{ex}(\oplus)$ if and only if $(S'', 1) \in \mathrm{M}_{ex}(\otimes)$ if and only if $(S'', 1) \in \mathrm{M}^{+}_{ex}(\cap, \otimes)$. It is obvious that $\mathrm{M}_{ex}(\oplus)$ reduces to $\mathrm{M}_{ex}(\cap, \oplus)$.

Let (S, b) be an instance of $\mathrm{M}_{ex}(\cap, \oplus)$ where $S = (\mathcal{F}, A)$. Let $n =_{\text{def}} \dim S$. S can be evaluated in the semiring $\mathrm{SR}(b)$, and the situation $\emptyset$ can be considered equal to the situation $b{+}1$. Hence, S can generate at most $(b{+}2)^n$ different configurations with respect to b, i.e., $b \in [S]$ if and only if $b \in S(t)$ for some $t < (b{+}2)^n$. Since $\mathrm{M}_{tm}(\cap, \oplus)$ is polynomial-time solvable by Theorem 29, $\mathrm{M}_{ex}(\cap, \oplus)$ is in NP.

$\mathrm{M}_{ex}(\otimes)$ reduces to $\mathrm{M}_{ex}(\cup) \cup \mathrm{M}^{+}_{ex}(\cap, \otimes)$ by the proof of Corollary 30. Let (S, b) be an instance of $\mathrm{M}^{+}_{ex}(\cap, \otimes)$ where $S = (\mathcal{F}, A)$. Let $n =_{\text{def}} \dim S$. Observe that S can be evaluated in the semiring $\mathrm{SR}(b)$ and that the situations $\emptyset$, a for $a > b$ and 0 can be considered equal. Hence, similar to recurrent $\{\cap, \oplus\}$-systems, S can generate at most $(b{+}1)^n$ different configurations with respect to b. Hence, $\mathrm{M}^{+}_{ex}(\cap, \oplus)$ is in NP by Corollary 30. ■

Corollary 33. SET-SCE *is* NP-*complete.*

It remains open not only whether $\mathrm{M}_{tm}(\cap, \otimes)$ is polynomial-time decidable but also whether $\mathrm{M}_{ex}(\cap, \otimes)$ is contained in NP. We do not know any upper bound c for t such that $0 \in [S]$ if and only if $0 \in S(t)$ for some $t < c$ where S is a recurrent $\{\cap, \otimes\}$-system.

8. PSPACE**-complete membership problems.** This section contains three interesting results. First, we will see that the existential membership problem for finite recurrent $\{\cup, \cap\}$-systems is PSPACE-complete. Containedness is less more than an observation. Hardness is shown by a reduction from QBF. Astoundingly at first glance, the corresponding exact membership problem is PSPACE-complete, too. This differs from the problems studied in Sections 6 and 7. Second, we will show that $\mathrm{M}_{tm}(\cup, \oplus, \otimes)$ can be decided in polynomial space. This result is surprising when we keep in mind that $\mathrm{MC}(\cup, \oplus, \otimes)$ is PSPACE-complete [16]. Third, we will show that a recurrent $\{\cup, \oplus, \otimes\}$-system S needs at most $(b+1) \cdot 2^{n^3}$ evaluation steps to generate number b where $n =_{\text{def}} \dim S$. This leads to a polynomial-space decision algorithm for $\mathrm{M}_{ex}(\cup, \oplus, \otimes)$.

A *quantified Boolean formula* is a first order logic-formula without functions, predicates and free variables. We assume that every quantified Boolean formula H is of the form $H = Q_1 x_1 \ldots Q_n x_n H'(x_1, \ldots, x_n)$ where $Q_1, \ldots, Q_n \in \{\exists, \forall\}$ and $H'(x_1, \ldots, x_n)$ is a Boolean formula with $\wedge$, $\vee$ and $\neg$ and every $\neg$ applies only to variables x_i, $i \in [1, n]$. Every quantified Boolean formula evaluates to *true* or *false* (1 or 0, respectively). The problem QBF is the set of all true quantified Boolean formulas. This problem was introduced by Stockmeyer and Meyer [10].

Theorem 34. [10] QBF *is* PSPACE-*complete.*

We want to enumerate all binary n-tuples over $\{\text{false}, \text{true}\}$ iteratively representing the Boolean values by the question whether a given set contains a specified element. Consider the following $\{\cup, \cap\}$-functions:

$$\zeta_1(a, \overline{a}, e, \overline{e}, c, \overline{c}) =_{\text{def}} (((a \cap \overline{c}) \cup (\overline{a} \cap c)) \cap \overline{e}) \cup (e \cap c) \quad \text{and}$$
$$\zeta_2(a, \overline{a}, e, \overline{e}, c, \overline{c}) =_{\text{def}} (((a \cap c) \cup (\overline{a} \cap \overline{c})) \cap \overline{e}) \cup (e \cap \overline{c})\,.$$

By checking all possible cases we obtain the following lemma. The operator $\triangle$ denotes the symmetric difference, i.e., the set-theoretic equivalent of the Boolean xor-function.

Lemma 35. *Let $A, B, C, D, E, F \subseteq \mathbf{N}$ and $b \in \mathbf{N}$. If $b \in (A \triangle B) \cap (C \triangle D) \cap (E \triangle F)$ then*

i. $b \in \zeta_1(A, B, C, D, E, F) \triangle \zeta_2(A, B, C, D, E, F)$,

ii. $b \in \zeta_1(A, B, C, D, E, F) \triangle E$ *if and only if* $b \in A \cap D$.

Using functions ζ_1 and ζ_2 we can construct a recurrent $\{\cup, \cap\}$-system that enumerates all binary n-tuples, $n \geq 1$. Let $\mathbf{u} =_{\text{def}} (u_1, \overline{u}_1, \ldots, u_n, \overline{u}_n)$. We define:

$$
\begin{aligned}
f_1(\mathbf{u}) &=_{\text{def}} \overline{u}_1 \\
f_1'(\mathbf{u}) &=_{\text{def}} u_1 \\
f_{i+1}(\mathbf{u}) &=_{\text{def}} \zeta_1(u_i, \overline{u}_i, f_i(\mathbf{u}), f_i'(\mathbf{u}), u_{i+1}, \overline{u}_{i+1}) \\
f_{i+1}'(\mathbf{u}) &=_{\text{def}} \zeta_2(u_i, \overline{u}_i, f_i(\mathbf{u}), f_i'(\mathbf{u}), u_{i+1}, \overline{u}_{i+1}), \ i \in [1, n{-}1].
\end{aligned}
$$

Observe that every function can be represented by a $\{\cup, \cap\}$-circuit of size at most $c \cdot n$ for some constant c. (A close look reveals $c < 20$.) Let $\mathcal{F}^n_{\text{enum}} =_{\text{def}} \langle f_1, f_1', \ldots, f_n, f_n' \rangle$, $A^n_{\text{enum}} =_{\text{def}} (\{0\}, \{1\}, \ldots, \{0\}, \{1\})$ and $S^n_{\text{enum}} =_{\text{def}} (\mathcal{F}^n_{\text{enum}}, A^n_{\text{enum}})$. With these definitions the prerequisites of Lemma 35 are fulfilled using $b = 1$. The next lemma shows that $f_n(t) \ldots f_1(t)$ can be interpreted as a representation of the n least significant bits of the binary representation of $t \in \mathbf{N}$. Let $\text{bin}_i(t)$, $i \geq 0$, denote the i-th least significant bit of the binary representation of t; the count starts with 0, i.e., $\text{bin}_i(t)$ is the coefficient of 2^i in the binary representation of t.

Lemma 36. *For every $t \in \mathbf{N}$ and $i \in [1, n]$:* $1 \in f_i(t) \iff \text{bin}_{i-1}(t) = 1$.

Proof: We prove the lemma by induction over i. The claim is obviously true for $i = 0$. Let $i < n$. First, $f_{i+1}(0) = \{0\}$. Consider $f_{i+1}(t{+}1)$. If $\text{bin}_{i-1}(t) \leq \text{bin}_{i-1}(t{+}1)$ then $\text{bin}_i(t) = \text{bin}_i(t{+}1)$. By induction hypothesis and Lemma 35, the claim holds. If $\text{bin}_{i-1}(t) > \text{bin}_{i-1}(t{+}1)$ then $\text{bin}_i(t) \neq \text{bin}_i(t{+}1)$. Induction hypothesis and Lemma 35 show the claim. ∎

Let $H = Q_1 x_1 \ldots Q_n x_n H'(x_1, \ldots, x_n)$ be a quantified Boolean formula. For $t \in \mathbf{N}$ and $i \in [0, n]$, let $H^{(i)}(t) =_{\text{def}} Q_{n-i+1} x_{n-i+1} \ldots Q_n x_n H'(\text{bin}_{n-1}(t), \ldots, \text{bin}_i(t), x_{n-i+1}, \ldots, x_n)$. Note that the functions bin_i are ordered by decreasing index whereas the variables x_j are ordered by increasing index. Suppose that $Q_1 = \exists$. Then, $H \equiv (H^{n-1}(0) \vee H^{n-1}(2^{n-1}))$. If we (inductively) construct an evaluation tree for H where the values of $H^{n-1}(0)$ and $H^{n-1}(2^{n-1})$ are the values of the predecessors of a final $\vee$-vertex, $H^n(0)$ denotes the value of this $\vee$-vertex. Hence, $H^i(t)$ for $i \in [0, n]$ and $t \geq 0$ represents the value of some $\vee$- or $\wedge$-vertex in that tree.

Theorem 37. QBF $\leq^{\text{L}}_m$ $\text{M}_{ex}(\cup, \cap)$.

Proof: Let $H = Q_1 x_1 \ldots Q_n x_n H'(x_1, \ldots, x_n)$ be a quantified Boolean formula in the variables $x_1, \ldots, x_n$. We will define a recurrent $\{\cup, \cap\}$-system that evaluates H as it would be done by an alternating Turing machine. Let

$$
\begin{aligned}
A &=_{\text{def}} (\{0\}, \{1\}, \ldots, \{0\}, \{1\}, \{1\}, \{0\}, \ldots, \{0\}, \{0\}, \ldots, \{0\}, \{0\}, \ldots, \{0\}) \\
\mathcal{F} &=_{\text{def}} \langle f_1,\ f_1',\ \ldots,\ f_n,\ f_n',\ e_0,\ e_1,\ \ldots,\ e_n,\ g_0,\ \ldots,\ g_n,\ h_0,\ \ldots,\ h_n \rangle \\
\mathbf{x} &=_{\text{def}} (u_1,\ \overline{u}_1,\ \ldots,\ u_n,\ \overline{u}_n,\ c_0,\ c_1,\ \ldots,\ c_n,\ s_0,\ \ldots,\ s_n,\ z_0,\ \ldots,\ z_n).
\end{aligned}
$$

where $\mathcal{F}^n_{\text{enum}} =_{\text{def}} \langle f_1, f_1', \ldots, f_n, f_n' \rangle$. The functions $f_1, \ldots, f_n'$ are defined on $u_1, \ldots, \overline{u}_n$, and the functions e_i, g_i, h_i, are defined as follows. For $i \in [1,n]$:

$$
\begin{aligned}
e_0(\mathbf{x}) &=_{\text{def}} c_0\\
e_i(\mathbf{x}) &=_{\text{def}} e_{i-1}(\mathbf{x}) \cap f_i(\mathbf{x})\\
g_0(\mathbf{x}) &=_{\text{def}} s_0\\
g_i(\mathbf{x}) &=_{\text{def}} (u_i \cap s_i) \cup z_{i-1}\\
h_0(\mathbf{x}) &=_{\text{def}} \mathrm{val}_{H'}(\mathbf{x})\\
h_i(\mathbf{x}) &=_{\text{def}} e_i(\mathbf{x}) \cap \begin{cases} (g_i(\mathbf{x}) \cup h_{i-1}(\mathbf{x})) & \text{, if } Q_{n-i+1} = \exists\\ (g_i(\mathbf{x}) \cap h_{i-1}(\mathbf{x})) & \text{, if } Q_{n-i+1} = \forall\,. \end{cases}
\end{aligned}
$$

Function $\mathrm{val}_{H'}(\mathbf{x})$ will be specified later. However, it holds that $1 \in \mathrm{val}_{H'}(t) \Leftrightarrow H^{(0)}(t) \equiv 1$. We will show that $h_n(2^n-1)$ contains 1 if and only if $H \in \mathrm{QBF}$. Observe that $1 \in e_i(t)$ for $t \in \mathbf{N}$ if and only if $\mathrm{bin}_{i-1}(t) = \cdots = \mathrm{bin}_0(t) = 1$.

Claim A. Let $i \in [1,n]$ and $t_0 \in \mathbf{N}$ such that $1 \in e_i(t_0)$. Then, $1 \in g_i(t_0)$ if and only if $1 \in h_{i-1}(t_0 - 2^{i-1})$.

We show by induction over $t \in [t_0 - 2^{i-1}+1, t_0]$ that $1 \in g_i(t)$ if and only if $1 \in h_{i-1}(t_0 - 2^{i-1})$. Since $1 \notin f_i(t_0 - 2^{i-1})$ it holds that $1 \in g_i(t_0 - 2^{i-1}+1) \Leftrightarrow 1 \in h_{i-1}(t_0 - 2^{i-1})$. Let $t \in [t_0 - 2^{i-1}+1, t_0-1]$. $1 \notin e_{i-1}(t)$ hence $1 \notin h_{i-1}(t)$. We obtain

$$1 \in g_i(t{+}1) \iff 1 \in (f_i(t) \cap g_i(t)) \cup h_{i-1}(t) \iff 1 \in g_i(t) \iff 1 \in h_{i-1}(t_0 - 2^{i-1})\,.$$

Claim B. Let $t_0 \in \mathbf{N}$ such that $1 \in e_n(t_0)$. Then, $1 \in h_n(t_0)$ if and only if $H \equiv 1$.

We show by induction over $i \in [0,n]$ that for every $t \in \mathbf{N}$ where $1 \in e_i(t)$ holds: $1 \in h_i(t) \iff H^{(i)}(t) \equiv 1$. For $i = 0$ the claim holds by definition. Let $i \in [0, n{-}1]$. If $Q_{n-i} = \exists$ then

$$
\begin{aligned}
1 \in h_{i+1}(t) &\iff 1 \in e_{i+1}(t) \cap (g_{i+1}(t) \cup h_i(t))\\
&\iff 1 \in g_{i+1}(t) \cup h_i(t)\\
&\iff 1 \in h_i(t - 2^i) \cup h_i(t)\\
&\iff H^{(i)}(t - 2^i) \equiv 1 \text{ or } H^{(i)}(t) \equiv 1\\
&\iff H^{(i+1)}(t) \equiv 1\,.
\end{aligned}
$$

If $Q_{k-i} = \forall$ the proof is similar to the previous case.

To obtain function $\mathrm{val}_{H'}$ we introduce new variables $\overline{x}_1, \ldots, \overline{x}_n$ and replace every occurrence of $\neg x_i$ by $\overline{x}_i$. (Remember that $\neg$ only applies to variables.) Replacing $\vee$ and $\wedge$ by $\cup$ and $\cap$, respectively, and x_i and $\overline{x}_i$ by u_{n-i+1} and $\overline{u}_{n-i+1}$, respectively, we obtain the function $\mathrm{val}_{H'}$ which has the desired property. $S =_{\text{def}} (\mathcal{F}, A)$ can be computed by an FL-function. It holds that $H \in \mathrm{QBF} \longleftrightarrow (S, 1) \in \mathrm{M}_{ex}(\cup, \cap)$. ■

We provide a small example that shall illustrate the construction of the last proof. We choose the quantified Boolean formula H as

$$H =_{\text{def}} \exists x_1 \forall x_2 \exists x_3((x_1 \wedge \neg x_2) \vee (\neg x_3 \wedge x_1))\,.$$

This formula obviously evaluates to 1. In Figure 5 the result of the transformation process is presented by means of a circuit representation. The S^3_{enum} part is not included into the picture due to legibleness. Of course, every function must be represented by its own circuit,

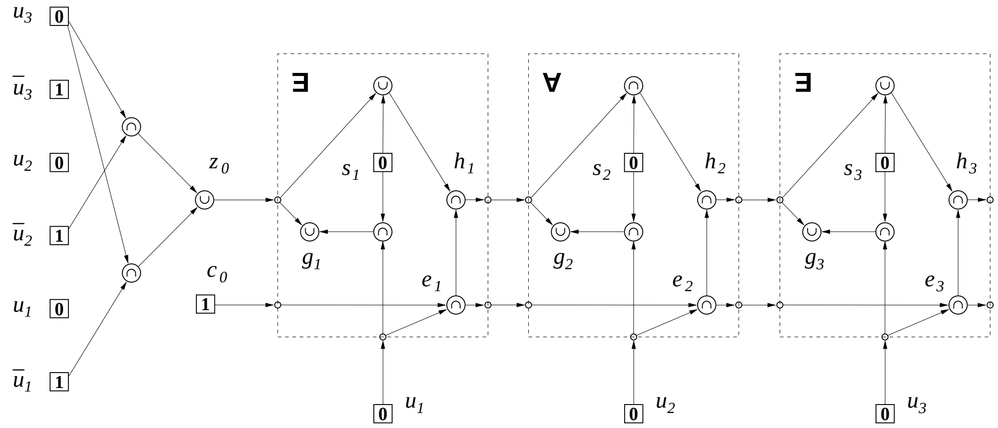

Figure 5. The result for formula $H =_{\rm def} \exists x_1 \forall x_2 \exists x_3((x_1 \wedge \neg x_2) \vee (\neg x_3 \wedge x_1))$.

which can be obtained from the given picture by deleting all irrelevant vertices. Vertices with the same name are considered identical, which applies to the vertices u_1, u_2, u_3.

We now turn our attention to recurrent $\{\cup, \oplus, \otimes\}$-systems. We will show that the exact as well as the existential membership problem for these systems are in PSPACE. To achieve this we will first show that we can bound the number of evaluation steps for proving that a number b can be generated by a given system by some "slowly growing" function. Let $S = (\mathcal{F}, A)$ be a recurrent $\{\cup, \oplus, \otimes\}$-system, $\mathcal{F} = \langle f_1, \ldots, f_n \rangle$, and let $t \in \mathbf{N}$. With each function of $\mathcal{F}$ we associate a combinatorial $\{\cup, \oplus, \otimes\}$-circuit. Consider the following definition:

$$f_i^{(0)}(\mathbf{x}) =_{\rm def} x_i \text{ and } f_i^{(r+1)}(\mathbf{x}) =_{\rm def} f_i(f_1^{(r)}(\mathbf{x}), \ldots, f_n^{(r)}(\mathbf{x})), \ r \geq 0,$$
$$f^{(t)}(\mathbf{x}) =_{\rm def} f_n^{(t)}(\mathbf{x}).$$

We obtain $f^{(t)}$ by t-fold superposition. Similarly, we obtain, by identifying vertices, a circuit representation C of $f^{(t)}$. Labelling the input vertices of C with the corresponding numbers of A results in a $\{\cup, \oplus, \otimes\}$-circuit representation C_A of $S(t) = f^{(t)}(A)$. The formula representation F of $S(t)$ is obtained from C_A by *unfolding* circuit C_A, i.e., F is a tree and each vertex of F is root of a subtree of F. A *computation tree* for F is a maximal subtree of F such that every vertex with label $\cup$ has exactly one predecessor and every other vertex (vertices with labels $\oplus$ or $\otimes$) has two predecessors. If we additionally label vertices of a computation tree with their value, which is for input vertices the label itself, for vertices with only one predecessor the value of the predecessor and for the other vertices the sum or product of the values of the predessors, then the root vertex has a member of $S(t)$ as value. Computation trees have been introduced by McKenzie and Wagner to analyse the complexity of MC$(\cup, \oplus)$. A number is in $S(t)$ if and only if it is the value of the root vertex of a computation tree for F.

Lemma 38. *Let $S = (\mathcal{F}, A)$ be a recurrent $\{\cup, \oplus, \otimes\}$-system, $n =_{\rm def} \dim S$. Let $b \in \mathbf{N}$. Then, $b \in [S]$ if and only if there is $t < (b+1) \cdot 2^{n^3}$ such that $b \in S(t)$.*

Proof: If $n = 1$ the claim is immediately clear. Let $n \geq 2$. If $b \leq 1$ then it suffices to evaluate S in the semiring SR(1). Since there are only eight possible sets for each input, the number of evaluation steps to decide $(S, b) \in \mathrm{M}_{ex}(\cup, \oplus, \otimes)$ can be bounded by $8^n = 2^{3n} < 2^{n^3}$. Let $b \geq 2$. We assume that no function of $\mathcal{F}$ is of the form $\varphi(\mathbf{x}) = x_i$. Otherwise, we would add an $(n+1)$-th component $(f_{n+1}(\mathbf{x}') =_{\rm def} x_1 \oplus x_{n+1}, b+1)$ to S and replace every function $f_i(\mathbf{x}) = x_j$ by $f_i'(\mathbf{x}') =_{\rm def} x_j \cup x_{n+1}$. Obviously, this does not effect the question $b \in [S]$ and is only of technical advantage. Let $t \geq 0$ be smallest possible such that $b \in S(t)$. Let F be the formula representation of $S(t)$, and let T be a computation tree for F with root value b. The *reduced* computation tree B of T is the maximal subtree of T containing the root vertex of T that does not contain vertices with value greater than 0 in a subtree with a root vertex labelled with $\otimes$ and with value 0. In other words, if u is a vertex in B that is labelled with $\otimes$ and has value 0, then all paths in B from input vertices to u contain only vertices with value 0. We define *levels* of B as follows. Each vertex of B that corresponds to the output vertex of the circuit representation of some function of $\mathcal{F}$ is labelled with the index of the corresponding function; the input vertices of B are labelled with the index of the corresponding variable. Note that some vertices now have two kinds of labels: numbers from A or $\cup, \oplus, \otimes$ and indices $1, \ldots, n$. Then, L_r contains exactly those vertices that are labelled with an index and that lie each on a leaf-root path in B with exactly r index-labelled vertices preceding it. Leaves or input vertices of B are exactly the vertices in L_0 whereas the root vertex of B is the only member of L_t. Observe that every leaf-root path in B passes the same number of index-labelled vertices. Note the following correspondence: if some vertex from L_r is labelled with index i and has value a then $a \in f_i(r)$.

By our definition of a reduced computation tree the value of a vertex in B is not smaller than the value of any of its predecessors. Let u and v be vertices in B with the same value such that there is a path from v to u. Then, every vertex on the v,u-path has the same value as u and v. Suppose there are $r_1, r_2 \in \mathbf{N}$ such that $r_1 \geq 2^{n^3} + r_2$ and for each vertex u in L_{r_1} there is a vertex $v \in L_{r_2}$ in the subtree of B rooted by u such that u and v have the same value. Such vertices constitute pairs. If u has value greater than 1, there is only one such pair for u. Let (u, v) be a pair with value of u greater than 1. Consider the v,u-path P. Let w be a vertex on P different from v. If w has label $\oplus$ then its predecessor that does not lie on P has value 0, if w has label $\otimes$ the respective predecessor has value 1. Let (u', v') be another pair such that u and u' as well as v and v' have the same index-labels. Then, we can replace the subtree rooted by u' by a copy of the subtree rooted by u, replace the subtree rooted by v in the copy by the subtree rooted by v' (see Figure 6) and obtain a reduced computation tree for F with root value b. We repeat this procedure for other pairs as long as possible and obtain a reduced computation tree B^* for F with root value b that contains at most n^2 different subtrees between L_{r_1} and L_{r_2} rooted by vertices with values greater than 1. To each level L_r we assign a tuple indicating for every $i \in [1, n]$ whether a vertex in L_r has label i and value 0 or value 1 and for every pair $(i, j) \in [1, n]^2$ the index of the vertex in L_r of B^* that is passed by the path between a vertex in L_{r_2} with label j and value greater than 1 and a vertex in L_{r_1} with label i. There are at most $2^{2n} \cdot n^{n^2} \leq 2^{n^3}$ different tuples possible. It follows that there must be two levels L_{s_1} and L_{s_2}, $r_1 \leq s_1 < s_2 \leq r_2$, with the same assigned tuple. Then, we can delete levels $L_{s_1}, \ldots, L_{s_2 - 1}$ and build a reduced computation tree with root value b that proves $b \in S(t - (s_2 - s_1))$, which is a contradiction to the assumption that t is smallest possible.

First, observe that B contains a leaf with value greater that 0. Second, for every $r \geq 2^{n^3}$ there is a vertex u in L_r such that every vertex in $L_{r - 2^{n^3}}$ in the subtree rooted by u has a smaller value than the value of u, and u has value greater than 1. Note that, if u has value 1, there is a path from an input vertex to u that contains only vertices with value 1. If $\sum^{\geq 2} L_r$ denotes the sum of the values of the vertices in level L_r with value greater than 1, it holds for every $r < t$ that $\sum^{\geq 2} L_r \leq \sum^{\geq 2} L_{r+1}$. It follows that $\sum^{\geq 2} L_{2^{n^3}} \geq 2$, $\sum^{\geq 2} L_{2 \cdot 2^{n^3}} \geq 3$, and so on, so that $t \leq b \cdot 2^{n^3}$. ■

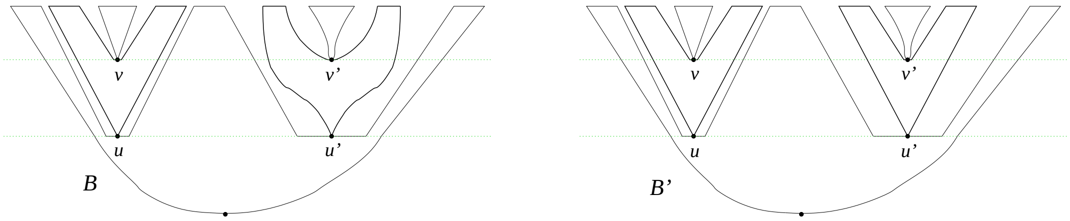

Figure 6. Construction of Lemma 38.

We want to make two remarks on the just proven result. The factor $b{+}1$ could be replaced for every $b > 1$ by $b{-}1$, as it is proved. But this would entail an unpleseant case study between $b \le 1$ and $b > 1$. Second, $\{\cup, \oplus, \otimes\}$ is a maximal set of operations for which we can bound the number of evaluation steps, since $\mathrm{M}_{ex}(\cup, \cap, \oplus, \otimes)$ and $\mathrm{M}_{ex}(\cup, \cap, {}^-, \oplus, \otimes)$ are undecidable. At first glance, one might argue that undecidability of the latter problem is due to (the expected) undecidability of the non-iterated variant $\mathrm{MC}(\cup, \cap, {}^-, \oplus, \otimes)$. However, every bound for the general problem serves as bound for a restricted variant, hence $\mathrm{M}_{ex}(\cup, \cap, \oplus, \otimes)$ would be decidable.

As we have already discussed the problems $\mathrm{M}_{tm}(\mathcal{O})$ for $\mathcal{O} \subseteq \{\cup, \cap, {}^-, \oplus, \otimes\}$ can be considered similar to the problems $\mathrm{MC}(\mathcal{O})$ with succinct input representation. For most of our problems succinctness led to an increasement of complexity. With this phenomenon in mind it is surprising that we can show that $\mathrm{M}_{tm}(\cup, \oplus, \otimes)$ is solvable in polynomial space. It is known that $\mathrm{MC}(\cup, \oplus, \otimes)$ is PSPACE-complete [16]. To give a short introduction to the following proof. Consider some recurrent $\{\oplus\}$-system S, and let F be the formula representation of $S(t)$ for some $t \ge 0$. Then, F is also a computation tree, and the value of the root vertex is determined by the number of paths from leaves to the root vertex. These numbers can be computed by matrix multiplication as shown in the proof of Proposition 26. If we may additionally have $\cup$- and $\otimes$-vertices, not all paths in the formula representation contribute to the result.

Theorem 39. $\mathrm{M}_{tm}(\cup, \oplus, \otimes)$ *is in* PSPACE.

Proof: Let (S, t, b) be an instance of $\mathrm{M}_{tm}(\cup, \oplus, \otimes)$ where $S = (\mathcal{F}, A)$, $\mathcal{F} = \langle f_1, \dots, f_n \rangle$, $n =_{\text{def}} \dim S$, and $A = (a_1, \dots, a_n)$. Let $C_1, \dots, C_n$ be the circuit representations of $f_1, \dots, f_n$. If $b \le 1$ we can evaluate all functions iteratively in the semiring $\mathrm{SR}(1)$ and compute $S(t) \cap [0, 1]$ in polynomial space. Let $b > 1$, and let $b \in S(t)$. Let F be the formula representation of $S(t)$ obtained by unfolding the circuit representation of $S(t)$, and let T be a computation tree for F with root value b. Observe the following: any leaf-root path that does not contain a vertex with value 0 contains at most $\lfloor \log b \rfloor$ $\otimes$-vertices without predecessors with value 1. We will give an algorithm that verifies the existence of a computation tree for F with root value b. The main idea is as follows. Let B' emerge from T by replacing every $\otimes$-vertex of T by an input vertex of the same value. (Delete all vertices that are not accessible from the root.) Obviously, the value of every vertex u in B' is the sum of all input vertices in the subtree with root vertex u and is equal to the value of the corresponding vertex of T. Now, obtain B from B' by removing all vertices with value 0. Then, there are at most b leaves in B, and $\oplus$-vertices may have only one predecessor.

A vertex of B is *marked* if it corresponds to a vertex that emerged in the circuit representation of $S(t)$ from identifying an input and an output vertex. These marked vertices are additionally labelled with the number of the original input vertex. (Recall

Figure 1: the right circuit contains two such vertices that emerged from an identification operation, and they would be labelled with 1 and 2.) We consider the levels of B defined in the same way as in the proof of Lemma 38, i.e., marked vertex u belongs to level r, denoted by L_r, $r \in [0,t]$, if and only if u is contained in the subtree of B of exactly $t-r+1$ marked vertices. The root vertex of B solely forms L_t. With L_r we associate an $(n+1)$-tuple $(\nu_1^r, \ldots, \nu_n^r, \sigma^r)$ where ν_i^r, $i \in [1,n]$, denotes the number of marked vertices in L_r with label i and σ^r is the sum of the values of all marked vertices in L_r. Note that this sum may be smaller than b since subtrees with root vertex a $\otimes$-vertex have been cut. Our algorithm will iteratively generate such tuples associated to levels starting from level t and verify that the difference between the last components of the tuples associated to consecutive levels is the sum of the values of appropriate computation trees (those that have been cut). The tuple $(0,\ldots,0,1,b)$ is associated to level t. Let tuple $(\nu_1^r, \ldots, \nu_n^r, \sigma^r)$ associated to level $r > 0$ be known. We describe how to find a tuple for level $r-1$. Remember that no vertex in B has value 0. Let $i \in [1,n]$ such that $\nu_i^r > 0$. Decrease ν_i^r by 1. The algorithm guesses how much a copy of a computation tree of the unfolded circuit representation C_i of f_i contributes to $\tau^{r-1} =_{\mathrm{def}} (\nu_1^{r-1}, \ldots, \nu_n^{r-1}, \sigma^{r-1})$, i.e., which vertices from L_{r-1} are contained in a subtree rooted by some marked vertex from L_r with label i. Let the vertices of C_i be ordered arbitrarily. In polynomial space we can examine the paths of C_i one after the other. If a path contains a $\cup$-vertex, the subtree of exactly one predecessor must be considered. If a path contains a $\otimes$-vertex the value of the $\otimes$-vertex at shortest distance to the output gate must be determined. This vertex then corresponds to a leaf of B that replaced a $\otimes$-vertex in T. This routine is described in the next paragraph. For each $\oplus$-vertex u it must be guessed whether it has value 0 or greater than 0. The former case can be verified in polynomial space as follows. Let f' be a sub-function of f_i that computes the value of u. (This function is represented by a sub-circuit of C_i.) The algorithm simply verifies $(S', r, 0) \in \mathrm{M}_{tm}(\cup,\oplus,\otimes)$ where S' is obtained from S by adding function f' as $(n+1)$-th function. If u is decided to have value 0 then the subtree rooted at u does not have to be considered and does not contribute to τ^{r-1}. If u has value greater than 0 all leaves with value greater than 0 in its subtree contribute to τ^{r-1}. The algorithm repeats this procedure until there is no $i \in [1,n]$ such that $\nu_i^r > 0$. Set $\sigma^{r-1} =_{\mathrm{def}} \sigma^r$. Tuple τ^{r-1} is generated and the algorithm proceeds with this tuple until the tuple associated with level 0 is obtained. If $\Sigma_{i=1}^n \nu_i^0 \cdot a_i = \sigma^0$ then accept. If $b \notin S(t)$, the existence of no reduced computation tree with root value b can be proved. It is obvious that the above described part of our algorithm needs only polynomial space.

It remains to show how to verify the value of a $\otimes$-vertex u. Similar to the case of a $\oplus$-vertex with value 0, we can verify in polynomial space whether u has value 0 or 1. Let u_1 and u_2 be the predecessors of u. If u has value $p > 1$ then u_1 must have value $p_1 > 0$ and u_2 must have value $p_2 > 0$ such that $p = p_1 \cdot p_2$. It can be verified in polynomial space as described above whether $p_1 = 1$ or $p_2 = 1$. If one is true, u is treated as $\oplus$-vertex with one predecessor having value 0. If neither of the cases hold then reduce σ^r by p and the algorithm verifies $(S_1', r, p_1) \in \mathrm{M}_{ex}(\cup,\oplus,\otimes)$ and then $(S_2', r, p_2) \in \mathrm{M}_{ex}(\cup,\oplus,\otimes)$, where S_1' and S_2' emerge from S by adding as $(n+1)$-th component the subfunctions of f_i computing the values of u_1 and u_2, respectively. Hence, the space needed by the algorithm depends on the number of verifications that are carried out at the same time. It holds that every additional verification process corresponds to some $\otimes$-vertex and these vertices are contained in one leaf-root path of F. Then, there can be at most $\lfloor \log b \rfloor$ verification processes running at the same time, since $2 \leq p_1, p_2 \leq \frac{p}{2}$. This gives a polynomial space bound for the algorithm. ■

Theorem 40. $\mathrm{M}_{ex}(\cup,\cap)$, $\mathrm{M}_{ex}(\cup,\cap,^{-})$, $\mathrm{M}_{ex}(\oplus,\otimes)$, $\mathrm{M}_{ex}(\cup,\oplus)$, $\mathrm{M}_{ex}(\cup,\otimes)$, $\mathrm{M}_{ex}(\cup,\oplus,\otimes)$ *are* PSPACE-*complete.*

Proof: We first show that $\mathrm{M}_{ex}(\cup,\cap,^{-})$ is contained in PSPACE. Let (S,b) be an instance of $\mathrm{M}_{ex}(\cup,\cap,^{-})$ where $S=(\mathcal{F},A)$, $n=_{\mathrm{def}}\dim S$. There are exactly 2^n possible configurations of $\mathcal{F}(t)$ with respect to b. Start the evaluation process of S. If a configuration appears that has already been encountered there will be no new configuration of S with respect to b. After 2^n evaluation steps all possible configurations that can be generated by S have been encountered, hence $\mathrm{M}_{ex}(\cup,\cap,^{-})$ is in PSPACE. Containedness in PSPACE of all other problems is due to Lemma 38 and Theorem 39.

It is obvious that $\mathrm{M}_{ex}(\cup,\cap)$ reduces to $\mathrm{M}_{ex}(\cup,\cap,^{-})$. Let (S,b) be an instance of $\mathrm{M}_{ex}(\cup,\cap)$ where $S=(\mathcal{F},A)$ and $\mathcal{F}=\langle f_1,\ldots,f_n\rangle$, $n=_{\mathrm{def}}\dim S$. Let f_i', $i\in[1,n]$, emerge from f_i be replacing every $\cap$ by $\oplus$ and every $\cup$ by $\otimes$. Let A' emerge from A by replacing b by 0 and every other number by 1. Let $S'=_{\mathrm{def}}(\langle f_1',\ldots,f_n'\rangle,A')$. It holds that $x+y=0\Leftrightarrow x=y=0$ and $x\cdot y=0\Leftrightarrow x=0$ or $y=0$. By induction, it follows that $b\in S(t)$ if and only if $0\in S'(t)$, $t\in\mathbf{N}$. Hence, $(S,b)\in\mathrm{M}_{ex}(\cup,\cap)$ if and only if $(S',0)\in\mathrm{M}_{ex}(\oplus,\otimes)$. Similarly, $\mathrm{M}_{ex}(\cup,\cap)$ reduces to $\mathrm{M}_{ex}(\cup,\oplus)$ and $\mathrm{M}_{ex}(\cup,\otimes)$. ■

Corollary 41. $\mathrm{M}_{tm}(\cup,\cap)$, $\mathrm{M}_{tm}(\cup,\cap,^{-})$, $\mathrm{M}_{tm}(\oplus,\otimes)$, $\mathrm{M}_{tm}(\cup,\oplus)$, $\mathrm{M}_{tm}(\cup,\otimes)$, $\mathrm{M}_{tm}(\cup,\oplus,\otimes)$ *are* PSPACE-*complete.*

Proof: Let (S,b) be an instance of $\mathrm{M}_{ex}(\cup,\cap)$, $n=_{\mathrm{def}}\dim S$. Let $\mathbf{x}=_{\mathrm{def}}(x_1,\ldots,x_{n+1})$, and let b' be the smallest number such that $b'\neq b$. Add component $(f_{n+1}(\mathbf{x})=_{\mathrm{def}}x_n\cup x_{n+1},b')$ to S and obtain S'. Then, $(S,b)\in\mathrm{M}_{ex}(\cup,\cap)$ if and only if $(S',2^n,b)\in\mathrm{M}_{tm}(\cup,\cap)$. Since $\mathrm{M}_{tm}(\cup,\cap)$ reduces to all other problems, the statement holds. ■

9. More complicated problems. In this final section we consider those problems that we have not yet solved. These are most of the problems that allow $\cap$- and $\otimes$-operations. But also $\mathrm{M}_{tm}(\cup,\cap,^{-},\oplus)$ and $\mathrm{M}_{ex}(\cup,\cap,^{-},\oplus)$ are still open. We will not give tight upper and lower bounds. In most cases, we obtain upper bounds by adequately restate results by McKenzie and Wagner. However, the complexity of the mentioned problem $\mathrm{M}_{tm}(\cup,\cap,^{-},\oplus)$ can significantly be improved with respect to the corresponding result from [6]. Let us first recall some necessary results. The complexity class EXP is the class of all sets that can be decided in deterministic exponential time; NEXP is its nondeterministic counterpart (see also [8]). EXPSPACE is the class of all sets that can be decided in exponential space.

Theorem 42. [6] *i.* $\mathrm{MC}(\cap,\otimes)$ *is in* P.

ii. $\mathrm{MC}(\cap,\oplus,\otimes)$ *is in* coNP.

iii. $\mathrm{MC}(\cup,\cap,^{-},\oplus)$ *and* $\mathrm{MC}(\cup,\cap,^{-},\otimes)$ *are in* PSPACE.

iv. $\mathrm{MC}(\cup,\cap,\oplus,\otimes)$ *is* NEXP-*complete.*

Given a recurrent system S and some number t, we find a circuit representation of $S(t)$ by simply concatenating circuits as it was described in the last section. The class 2−NEXP is the class of all sets A for which there is a polynomial p such that A can be accepted in nondeterministic time 2^{2^p}. We obtain the following corollary.

Corollary 43. *i.* $\mathrm{M}_{tm}(\cap,\otimes)$ *is in* EXP.

ii. $\mathrm{M}_{tm}(\cap,\oplus,\otimes)$ *is in* coNEXP.

iii. $\mathrm{M}_{tm}(\cup,\cap,{}^{-},\oplus)$ *and* $\mathrm{M}_{tm}(\cup,\cap,{}^{-},\otimes)$ *are in* EXPSPACE.

iv. $\mathrm{M}_{tm}(\cup,\cap,\oplus,\otimes)$ *is in* 2−NEXP.

In case of recurrent $\{\cup,\cap,{}^{-},\oplus\}$-systems we can improve the trivial exponential-space upper bound.

Lemma 44. $\mathrm{M}_{tm}(\cup,\cap,{}^{-},\oplus)$ *is in* EXP.

Proof: Let (S,t,b) be an instance of $\mathrm{M}_{tm}(\cup,\cap,{}^{-},\oplus)$, $S=(\mathcal{F},A)$ and $\mathcal{F}=\langle f_1,\ldots,f_n\rangle$, $n=_{\mathrm{def}}\dim S$. Let $C_1,\ldots,C_n$ be the circuit representations of $f_1,\ldots,f_n$, respectively. Let ν denote the number of vertices in the circuits $C_1,\ldots,C_n$. Then, ν is bounded above by the length of the representation of (S,t,b). To compute $\mathcal{F}(i{+}1)$ from $\mathcal{F}(i)$, $i\geq 0$, means to compute ν results (the result on every vertex of each circuit). Hence, $S(t)$ can be computed using (at most) $\nu\cdot t$ intermediate results, and at most ν of them have to be kept at the same time. To answer the question whether $b\in S(t)$ it suffices to always consider only numbers not larger than b. Hence, each result can be represented by a sequence of (at most) $b{+}1$ numbers or an appropriate binary string. Given two such representations, union, intersection, complementation and addition can be computed in $c\cdot b^2$ steps, $c\geq 1$ independent of the input. Then, $b\in S(t)$ can be decided in $c'\cdot t\cdot \nu^3\cdot b^4$ steps, $c'\geq 1$, which gives exponential time in the length of the representation of (S,t,b). ■

Observe that $\mathrm{M}_{tm}(\cup,\cap)$ reduces to $\mathrm{M}_{tm}(\cup,\cap,\oplus)$ and $\mathrm{M}_{tm}({}^{-},\oplus)$. The latter reduction is done by replacing $\cap$ by $\oplus$, $A\cup B$ by $\overline{\overline{A}\oplus\overline{B}}$, the queried number b by 0 and every other number by 1. So, $\mathrm{M}_{tm}(\cup,\cap,\oplus)$, $\mathrm{M}_{tm}({}^{-},\oplus)$ and $\mathrm{M}_{tm}(\cup,\cap,{}^{-},\oplus)$ are PSPACE-hard. Since the corresponding non-iterative problem versions $\mathrm{MC}(\cup,\cap,\oplus)$, $\mathrm{MC}({}^{-},\oplus)$ and $\mathrm{MC}(\cup,\cap,{}^{-},\oplus)$ are PSPACE-complete [6], [12], we should expect that none of our problems is in PSPACE. However, we already encountered this effect in case of $\mathrm{M}_{tm}(\cup,\oplus,\otimes)$, which is PSPACE-complete even though $\mathrm{MC}(\cup,\oplus,\otimes)$ is also PSPACE-complete [16].

One easy way to achieve containedness of $\mathrm{M}_{tm}(\cup,\cap,\oplus)$—or $\mathrm{M}_{tm}({}^{-},\oplus)$—in PSPACE is by representing computed sets, i.e., results of $\{\cup,\cap,\oplus\}$-functions, efficiently such that the required operations can also by carried out efficiently. Efficiency in the present case means polynomial space. Simple representations that may be manipulated in polynomial time do not suffice, since this would immediately imply containedness of, for instance, $\mathrm{MC}(\cup,\cap,\oplus)$ in P, hence P = PSPACE (if the representation of singleton sets that we start with can be generated in polynomial time).

Proposition 45. $\mathrm{M}_{ex}(\cup,\cap,{}^{-},\oplus)$ *is in* EXPSPACE.

Proof: Let (S,b) be an instance of $\mathrm{M}_{ex}(\cup,\cap,{}^{-},\oplus)$, $n=_{\mathrm{def}}\dim S$. It suffices to evaluate S in the semiring $\mathrm{SR}(b)$, hence, input and output of the involved functions can be represented in space polynomial in b. We have n sets and every set can be one out of 2^b sets. S can generate at most $2^{n\cdot b}$ different configurations, i.e., $b\in[S]$ if and only if there is $t<2^{n\cdot b}$ such that $b\in S(t)$. This gives a decision algorithm working in exponential space by application of Lemma 44. ■

In a way similar to the reduction from $\mathrm{M}_{tm}(\cup,\cap)$ to $\mathrm{M}_{tm}({}^{-},\oplus)$, the former problem reduces to $\mathrm{M}_{tm}({}^{-},\otimes)$. Replace every $\cup$ by $\otimes$ and every $A\cap B$ by $\overline{\overline{A}\otimes\overline{B}}$, replace the queried number b by 0 and every other number by 1. Note that no $\{{}^{-},\otimes\}$-function on inputs only $\{0\}$ or $\{1\}$ can compute $\emptyset$, since no such function can compute a set that contains 0 and 1. This shows PSPACE-hardness of $\mathrm{M}_{tm}({}^{-},\otimes)$.

<table>
<tr><th>Operation set</th><th colspan="2">Exact problem M_{tm}</th><th colspan="2">Existential problem M_{ex}</th></tr>
<tr><td></td><td>Lower bound</td><td>Upper bound</td><td>Lower bound</td><td>Upper bound</td></tr>
<tr><td>$\cup\ \cap\ ^-\ \oplus\ \otimes$</td><td>NEXP</td><td>?</td><td>RE</td><td>?</td></tr>
<tr><td>$\cup\ \cap\ \oplus\ \otimes$</td><td>NEXP</td><td>2−NEXP</td><td colspan="2">RE</td></tr>
<tr><td>$^-\ \oplus\ \otimes$</td><td>PSPACE</td><td>?</td><td>RE</td><td>?</td></tr>
<tr><td>$\cup\ \oplus\ \otimes$</td><td colspan="2">PSPACE</td><td colspan="2">PSPACE</td></tr>
<tr><td>$\cap\ \oplus\ \otimes$</td><td>PSPACE</td><td>coNEXP</td><td>PSPACE</td><td>RE</td></tr>
<tr><td>$\oplus\ \otimes$</td><td colspan="2">PSPACE</td><td colspan="2">PSPACE</td></tr>
<tr><td>$\cup\ \cap\ ^-\ \oplus$</td><td>PSPACE</td><td>EXP</td><td>PSPACE</td><td>EXPSPACE</td></tr>
<tr><td>$\cup\ \cap\ \oplus$</td><td>PSPACE</td><td>EXP</td><td>PSPACE</td><td>EXPSPACE</td></tr>
<tr><td>$^-\ \oplus$</td><td>PSPACE</td><td>EXP</td><td>PSPACE</td><td>EXPSPACE</td></tr>
<tr><td>$\cup\ \oplus$</td><td colspan="2">PSPACE</td><td colspan="2">PSPACE</td></tr>
<tr><td>$\cap\ \oplus$</td><td>$\mathrm{C_{=}L}$</td><td>P</td><td colspan="2">NP</td></tr>
<tr><td>$\oplus$</td><td>$\mathrm{C_{=}L}$</td><td>P</td><td colspan="2">NP</td></tr>
<tr><td>$\cup\ \cap\ ^-\ \otimes$</td><td>PSPACE</td><td>EXPSPACE</td><td>PSPACE</td><td>RE</td></tr>
<tr><td>$\cup\ \cap\ \otimes$</td><td>PSPACE</td><td>EXPSPACE</td><td>PSPACE</td><td>RE</td></tr>
<tr><td>$^-\ \otimes$</td><td>PSPACE</td><td>EXPSPACE</td><td>PSPACE</td><td>RE</td></tr>
<tr><td>$\cup\ \otimes$</td><td colspan="2">PSPACE</td><td colspan="2">PSPACE</td></tr>
<tr><td>$\cap\ \otimes$</td><td>NL</td><td>EXP</td><td>NP</td><td>RE</td></tr>
<tr><td>$\otimes$</td><td>NL</td><td>P</td><td colspan="2">NP</td></tr>
<tr><td>$\cup\ \cap\ ^-$</td><td colspan="2">PSPACE</td><td colspan="2">PSPACE</td></tr>
<tr><td>$\cup\ \cap$</td><td colspan="2">PSPACE</td><td colspan="2">PSPACE</td></tr>
<tr><td>$\cap$</td><td colspan="2">NL</td><td colspan="2">NP</td></tr>
<tr><td>$\cup$</td><td colspan="2">NL</td><td colspan="2">NL</td></tr>
<tr><td>$^-$</td><td colspan="2">L</td><td colspan="2">L</td></tr>
<tr><td></td><td colspan="2">L</td><td colspan="2">L</td></tr>
</table>

Table 1. Our results. The question mark stands for Δ_2^{P} or Σ_2^{P}.

10. Conclusions. In this work we introduced finite recurrent systems over the power set of the natural numbers. We defined two natural problems for these systems. The one of these problems, the exact membership problem, solves a classical question, if it is restated in terms of arithmetic circuit: Doesa given circuit generate a given number? In this case the circuit is given in a concise representation, that is not as succinct as those representations used for typical succinctness versions of well-studied NP-complete problems. The second type of problems, the existential membership problems, asks whether a given number can be generated by a given system. Our systems could be defined via functions that used $\cup$, $\cap$, $^-$, $\oplus$ or $\otimes$ as operations. We studied the complexities of these problems with respect to the set of used operations. The results are summarised in Table 1. The question marks stand for complexity classes beyond the class of recursively enumerable sets.

The exact membership problems can be understood as concise variants of the membership problems for arithmetic circuits studied by McKenzie and Wagner [6]. In some cases we could show that our chosen representation does not result in an increment of complexity with respect to the corresponding McKenzie-Wagner problems. In other cases, e.g., $\mathrm{M}_{tm}(\cup, \oplus)$, we could state an increment from NP-completeness to PSPACE-completeness.

McKenzie and Wagner studied the emptyness problem for some circuits as an auxiliary problem. It would be interesting to solve the emptyness problem for recurrent $\{\cap, \oplus\}$-systems. The only systems for which the emptyness problem is not trivial and that we could solve are recurrent $\{\cap\}$-systems.

As we initially mentioned recurrent systems describe sets of natural numbers. It should be worth finding characterisations of sets that can be generated by special recurrent systems. One could also define further set construction rules, for instance, intersection instead of union. Do complexities change unpredictably?

Historical note. When Volker Diekert was at Würzburg for a talk in May 2002, he surprised us with an astounding representation of the Fibonacci numbers using a matrix. The k-th power of this matrix contained the $(k + 1)$-th Fibonacci number. Essentially, it was the example of Section 6.

Acknowledgements. The idea to study finite recurrent systems was the result of a discussion with Klaus Wagner when a lot of people at Würzburg were studying membership problems for arithmetic circuits. I thank Bernhard Schwarz for his help.

Bibliography

[1] Chr. Glasser, *private communication*, 2003.

[2] D. Hilbert, *Mathematische Probleme*, Nachrichten von der Königlichen Gesellschaft der Wissenschaften zu Göttingen 1900, Göttingen, pp. 253–297, 1900.

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

[4] S.C. Kleene, *Recursive Predicates and Quantifiers*, Transactions of the American Mathematical Society 53, pp. 41–73, 1943.

[5] Y.V. Matiyasevich, *Hilbert's Tenth Problem*, The MIT Press, 1993.

[6] P. McKenzie, K.W. Wagner, *The Complexity of Membership Problems for Circuits over Sets of Natural Numbers*, Proceedings of the 20th Annual Symposium on Theoretical Aspects of Computer Science, STACS 2003, Lecture Notes in Computer Science 2607, Springer, pp. 571–582, 2003.

[7] A. Mostowski, *On definable sets of positive integers*, Fundamenta Mathematicae 34, pp. 81–112, 1947.

[8] Ch.H. Papadimitriou, *Computational Complexity*, Addison-Wesley, 1994.

[9] W.J. Savitch, *Relationships Between Nondeterministic and Deterministic Tape Complexities*, Journal of Computer and System Sciences 4, pp. 177–192, 1970.

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

[11] R. Szelepcsényi, *The method of forced enumeration for nondeterministic automata*, Acta Informatica 26, pp. 279–284, 1988.

[12] St. Travers, *The Complexity of Membership Problems for Circuits over Sets of Integers*, Forschungsbericht 334, Institut für Informatik, Bayerische Julius-Maximilians-Universität Würzburg, 2004.

[13] H. Vollmer, *Introduction to Circuit Complexity*, Springer, 1999.

[14] K. Wagner, *The Complexity of Problems Concerning Graphs with Regularities*, Proceedings of the 11th International Symposium on Mathematical Fondations of Computer Science, MFCS 1984, Lecture Notes in Computer Science 176, Springer, pp. 544–552, 1984.

[15] K.W. Wagner, *The Complexity of Combinatorial Problems with Succinct Input Representation*, Acta Informatica 23, pp. 325–356, 1986.

[16] K. Yang, *Integer Circuit Evaluation Is* PSPACE-*Complete*, Journal of Computer and System Sciences 63, pp. 288–303, 2001.