Complexity of Constraint Satisfaction Problems over Finite Subsets of Natural Numbers

Titus Dose

Julius-Maximilians-Universität, Würzburg, Germany
dose@informatik.uni-wuerzburg.de

Abstract

We study the computational complexity of constraint satisfaction problems that are based on integer expressions and algebraic circuits. On input of a finite set of variables and a finite set of constraints the question is whether the variables can be mapped onto finite subsets of $\mathbb{N}$ (resp., finite intervals over $\mathbb{N}$) such that all constraints are satisfied. According to the operations allowed in the constraints, the complexity varies over a wide range of complexity classes such as L, P, NP, PSPACE, NEXP, and even Σ_1, the class of c.e. languages.

1998 ACM Subject Classification F.2.2 Nonnumerical Algorithms and Problems

Keywords and phrases Computational complexity, Constraint satisfaction problems, Integer expressions and circuits

Digital Object Identifier 10.4230/LIPIcs.MFCS.2016.32

1 Introduction

The problems investigated in this paper are motivated by constraint satisfaction problems and integer expressions. We first introduce these notions and then explain their connection.

Constraint satisfaction problems. A constraint satisfaction problem (CSP) is a computational problem that on input of a finite set of variables and a finite set of constraints asks whether there is a mapping from the variables to some fixed domain such that all constraints are satisfied.

An example of a classical CSP is 3-Colorability, i.e., the question of whether there is a mapping α from a graph's vertices onto $\{0, 1, 2\}$ such that for adjacent nodes u and v it holds that $\alpha(u) \neq \alpha(v)$.

The set of relations permitted in the constraints is called *constraint language*. Obviously CSPs over finite domains permitting arbitrary constraints belong to NP. The question of which constraint languages lead to CSPs even decidable in polynomial time has been a topic of intensive research over the past decades.

Feder and Vardi [4] conjectured a dichotomy for CSPs over finite domains such that these CSPs are either in P or NP-complete. This conjecture is still open.

In the past years there has been an increasing interest in CSPs over *infinite* domains. Here much higher complexities are obtained, and some problems are even undecidable.

Integer expressions and algebraic circuits. In 1973, Meyer and Stockmeyer [14] asked for the complexity of decision problems regarding so-called integer expressions. An integer expression is a term built by singleton sets of natural numbers, the pairwise addition, and set operations like union, intersection, or complement. Meyer and Stockmeyer investigated the membership problem, i.e., the question of whether a given natural number is contained in the subset of ℕ described by an integer expression. Moreover, they studied the inequality problem, i.e., the question of whether two given integer expressions describe the same subset of ℕ.

For some constant $m \in \mathbb{N}$ the integer expression $(0 \cup 2^1) + (0 \cup 2^2) + \cdots + (0 \cup 2^{m-1})$ describes the set of all even natural numbers $< 2^m$ in a succinct way. Note that here the natural number n is an abbreviation for the set $\{n\}$.

McKenzie and Wagner [11] considered generalized integer expressions and the complexity of membership problems for circuits over finite subsets of ℕ.

Here a circuit is a directed, acyclic graph with two kinds of nodes: on the one hand, there are input nodes containing a single natural number. On the other hand all remaining nodes, so-called operation nodes, perform one of the following operations: union, intersection, complement, pairwise addition, and pairwise multiplication. Each operation node has an indegree equal to the number of operands required by the operation.

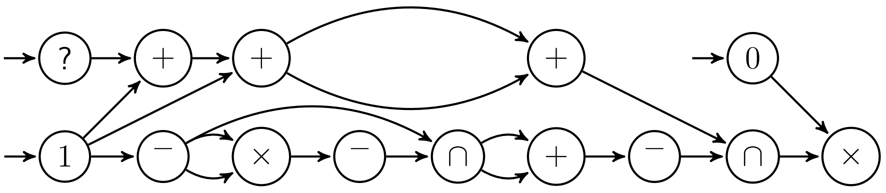
So each of the nodes computes a set of natural numbers: the input nodes compute singletons, and each operation node computes the corresponding set obtained from its predecessors. The set computed by the circuit overall is defined as the set computed by some fixed output node.

In contrast to integer expressions, these circuits are able to store intermediate results and reuse them several times. Thus, it is possible to describe large numbers and sets in a succinct way.

Similar to Meyer and Stockmeyer, who investigated the inequality problem for integer expressions, Glaßer et al. [5] considered the equivalence problem for circuits.

Moreover, Glaßer et al. [7] studied the satisfiability problem for circuits where the corresponding circuits are allowed to have unassigned input nodes. Now the problem is to decide for a given natural number b whether there exists an assignment for the input nodes such that b is contained in the set computed by the circuit. This modification makes the circuits even more similar to CSPs.

Consider the following circuit. Goldbach's conjecture fails if and only if there is an assignment for the ?-node in the circuit below such that the set computed by the rightmost node contains 0. Note that $\overline{\overline{1} \times \overline{1}} \cap \overline{1} = \mathbb{P}$ where $\mathbb{P}$ is the set of all primes.



Hence, if the satisfiability problem for circuits is decidable, then Goldbach's conjecture can be proven or refuted. Up to now it is not known whether this is possible.[1]

Connection of circuits and CSPs. Glaßer et al. [6] combined CSPs and integer circuits, and investigated CSPs over the domain of singletons of natural numbers and with operations from $\{+, \times, \cup, \cap, ^-\}$.

[1] Knuth [9] even assumes that Goldbach's conjecture is a problem that will never be solved. He mentions that "it might very well be that the conjecture happens to be true, but there is no rigorous way to prove it".

As we will see later, each CSP-instance can be represented by a primitive positive fo-sentence such as $\exists X\ (X + X + 4) \cap \big(\mathbb{P} + \mathbb{P}\big) = 0 \cap 1$, where $\mathbb{P}$ can be expressed by $\overline{\overline{1} \times \overline{1}} \cap \overline{1}$. Here we use natural numbers as abbreviations for singletons. Since each variable stands for a singleton, this sentence is true if and only if Goldbach's conjecture fails.

However, the set of singletons of natural numbers is not closed under the mentioned operations. As it can be seen in the example above, variables and constants are singletons, whereas there are terms that describe bigger and even infinite sets. Therefore, we consider CSPs over the domain $\mathcal{P}_{\text{fin}}(\mathbb{N}) = \{A \subseteq \mathbb{N} \mid A \text{ is finite}\}$, and replace the complement with the set difference. Thus, compared to the CSPs considered by Glaßer et al. [6] we consider problems that allow a straighter definition (i.e., variables and terms have the same domain), but that is more distant to the satisfiability problem for circuits.

Some results of the mentioned papers can be translated to our situation, but in general the questions for the enlarged domain turn out to be different ones.

Once an arithmetical and a set operation are permitted, we are not able to show the (un)decidability of the corresponding problems.

Therefore, we also consider a restricted version of the described CSPs, in which the domain is the set of all finite intervals over $\mathbb{N}$, denoted by $[\mathbb{N}]$. Note that this domain is not closed under all permitted operations anymore, but in several cases it is easier to obtain completeness results for this domain.

Our contribution. In the first part we consider all CSPs that only permit set operations. Here, for each problem we are able to show the $\leq_{\text{m}}^{\log}$-completeness for one of the complexity classes L, P, and NP. We observe that there is a case in which the problem over $[\mathbb{N}]$ is more difficult than the corresponding problem over $\mathcal{P}_{\text{fin}}(\mathbb{N})$.

When also admitting arithmetical operations, the complexity is much higher. Each of the problems is $\leq_{\text{m}}^{\log}$-hard for NP, and once both arithmetical operations are permitted, we obtain Σ_1-completeness.

The case with the addition as the only operation is particularly interesting. Here for CSPs over arbitrary finite sets we obtain only NP-hardness and membership in NEXP, and one of this paper's open questions is to close this gap. In contrast, the corresponding CSP over $[\mathbb{N}]$ turns out to be NP-complete. For the variant over $\mathcal{P}_{\text{fin}}(\mathbb{N})$ we even do not know whether the problem admitting addition and intersection is decidable. The corresponding problem over $[\mathbb{N}]$ is shown to be NP-complete again.

Furthermore, we consider the multiplication of intervals, and show that the CSP over $[\mathbb{N}]$ permitting only multiplication belongs to Σ_3^{p}. This result can be improved if the problem of testing whether two products of intervals are equal can be shown to belong to some class of the polynomial hierarchy lower than Π_2^{p}.

An overview over all results received in this paper can be found on pages 7 and 12. The technical report [3] provides a more comprehensive version of this paper.

2 Preliminaries

Let $\mathbb{N}$ (resp., $\mathbb{N}^+$) denote the set of non-negative (resp., positive) integers. $\mathcal{P}_{\text{fin}}(\mathbb{N})$ is the set of finite subsets of $\mathbb{N}$. For $A, B \in \mathcal{P}_{\text{fin}}(\mathbb{N})$ we define $A + B =_{\text{def}} \{a + b \mid a \in A, b \in B\}$ and $A \times B =_{\text{def}} \{ab \mid a \in A, b \in B\}$. By "$-$" we denote the set difference. Furthermore, $[\mathbb{N}]$ denotes the set of finite intervals over $\mathbb{N}$. An interval $\{x \mid a \leq x \leq b\}$ for non-negative integers a and b is represented by $[a, b]$. For $a > b$ it holds that $[a, b] = \emptyset$.

The set $f(A) = \{f(x) \mid x \in A\}$ for a function $f : A \to B$ and arbitrary sets A and B is also denoted by W_f.

We denote by L, P, NP, Σ_i^{p} and Π_i^{p} for $i \in \mathbb{N}$, PSPACE, and NEXP the standard complexity classes whose definitions can be found in textbooks on computational complexity [12]. The class of c.e. languages is denoted by Σ_1.

Note that commonly known $\leq_{\mathrm{m}}^{\mathrm{p}}$-complete problems for NP like SAT, 3SAT, or SOS are even $\leq_{\mathrm{m}}^{\log}$-complete for NP where $\leq_{\mathrm{m}}^{\log}$ (resp., $\leq_{\mathrm{m}}^{\mathrm{p}}$) denotes the many-one reduction computable in logarithmic space (resp., polynomial time).

We presume that finite subsets of natural numbers $\{a_1, \dots, a_k\}$ for $k \in \mathbb{N}^+$ are encoded such that the encoding's length is in $\Theta(\sum_{i=1}^{k} |a_k|)$, where $|a_k|$ denotes the length of the binary representation of a_k. An interval $[a, b]$ for $a < b$ is encoded such that the length of the encoding is in $\Theta(|a| + |b|)$. Note that all problems studied in this paper should be interpreted as subsets of $\mathbb{N}$ although for the sake of simplicity problems might be defined differently.

We successively define the CSPs this paper deals with. Let $\mathcal{O} \subseteq \{+, \times, \cup, \cap, -\}$ and X be a variable or a constant, where constants are finite subsets of $\mathbb{N}$.

Then X is a **term**. For terms β and γ as well as $\oplus \in \{+, \times, \cup, \cap, -\}$ the expression $(\beta \oplus \gamma)$ is a **term.** $X = Y$ for terms X and Y is an **atom**.

Let $a_1 = (t_{1,1} = t_{1,2}), \dots, a_m = (t_{m,1} = t_{m,2})$ for $m \in \mathbb{N}$ be atoms. Furthermore, let $X_1, \dots, X_n$ be the variables in the mentioned atoms. Then $\varphi = \exists X_1 \dots \exists X_n\ a_1 \wedge \cdots \wedge a_m$ (or more shortly $\varphi = a_1 \wedge \cdots \wedge a_m$) is an **$\mathcal{O}$-sentence**. If $\mathcal{O}$ is apparent from the context, we also write **sentence** instead of $\mathcal{O}$-sentence.

Let Var_φ denote the set of variables in φ, and let Const_φ denote the set of constants in φ. Moreover, we denote the set of all terms occurring in φ by T_φ.

We define the semantics of terms. A mapping $\alpha : T_\varphi \to \mathcal{P}_{\mathrm{fin}}(\mathbb{N})$ is an **assignment of terms** if the following conditions are satisfied.

- For constants C it holds that $\alpha(C) = C$.
- For variables X it holds $\alpha(X) \in \mathcal{P}_{\mathrm{fin}}(\mathbb{N})$.
- For $\oplus \in \{+, \times, \cup, \cap, -\}$ and all terms $X \oplus Y$ it holds that $\alpha(X \oplus Y) = \alpha(X) \oplus \alpha(Y)$.

φ is **true** if and only if there is an assignment of terms α with $\alpha(t_{i,1}) = \alpha(t_{i,2})$ for all $i = 1, \dots, m$. We call such an α **satisfying**.

The restriction of an assignment of terms to Var_φ (resp., in some cases $\mathrm{Var}_\varphi \cup \mathrm{Const}_\varphi$) is called **assignment of variables** or **assignment**.

Moreover, an assignment of variables β is **satisfying** if the assignment of terms induced by β is satisfying. Note that for each assignment of variables β there is exactly one assignment of terms α such that $\beta(X) = \alpha(X)$ for all $X \in \mathrm{Var}_\varphi$.

▶ **Definition 1.** Let $\mathcal{O}$ denote an arbitrary subset of $\{+, \times, \cup, \cap, -\}$. Then we define $\mathrm{CSP}\Big(\mathcal{P}_{\mathrm{fin}}(\mathbb{N}), \mathcal{O}\Big) =_{\mathrm{def}} \{\varphi \mid \varphi \text{ is a true } \mathcal{O}\text{-sentence}\}$.

Furthermore, we define

$$\mathrm{CSP}\Big([\mathrm{N}], \mathcal{O}\Big) =_{\mathrm{def}} \{\varphi \mid \varphi \text{ is an } \mathcal{O}\text{-sentence, all constants in } \varphi \text{ are intervals,}$$
$$\text{there is a satisfying assignment } \alpha \text{ with } \alpha(\mathrm{Var}_\varphi) \subseteq [\mathrm{N}]\}.$$

Here all constants are encoded as intervals. Hence, the length of the encoding of a constant $[a, b]$ for $a, b \in \mathbb{N}$ and $a \leq b$ is in $\Theta(|a| + |b|)$.

We also consider a slightly different problem. This enables us to simplify numerous proofs.

▶ **Definition 2.** Let $\mathcal{O} \subseteq \{+, \times, \cup, \cap, -\}$. We define

$$\mathrm{CSP}'\big(\mathcal{P}_{\mathrm{fin}}(\mathbb{N}), \mathcal{O}\big) =_{\mathrm{def}} \{\varphi \,|\, \varphi \text{ is true, and each atom in } \varphi \text{ is of the form } X \oplus Y = Z \text{ for } X, Y, Z \in \mathrm{Const}_\varphi \cup \mathrm{Var}_\varphi \text{ and } \oplus \in \mathcal{O}\},$$

where we identify the atoms $X \oplus Y = Z$ and $Z = X \oplus Y$. $\mathrm{CSP}'\big([\mathbb{N}], \mathcal{O}\big)$ for $\mathcal{O} \subseteq \{\cap, +\}$ is defined analogously to $\mathrm{CSP}'\big(\mathcal{P}_{\mathrm{fin}}(\mathbb{N}), \mathcal{O}\big)$.

The following lemma often allows us to presume that an input sentence is of the described simpler form. This is so because we can resolve a bigger term into several smaller terms by storing intermediate results in new variables. For CSPs over intervals, however, this works only when the set $[\mathbb{N}]$ is closed under the permitted operations.

▶ **Lemma 3.** $\mathrm{CSP}'\big(\mathcal{P}_{\mathrm{fin}}(\mathbb{N}), \mathcal{O}\big) \equiv_m^{\log} \mathrm{CSP}\big(\mathcal{P}_{\mathrm{fin}}(\mathbb{N}), \mathcal{O}\big)$ *for* $\mathcal{O} \subseteq \{+, \times, \cup, \cap, -\}$.
For $\mathcal{O} \subseteq \{+, \cap\}$ *it holds that* $\mathrm{CSP}'\big([\mathbb{N}], \mathcal{O}\big) \equiv_m^{\log} \mathrm{CSP}\big([\mathbb{N}], \mathcal{O}\big)$.

3 CSPs Permitting Set Operations Exclusively

We firstly focus on CSPs permitting only set operations.

Here all problems belong to NP, which is not the case when also arithmetic operations are allowed. In Section 4 we will see that some CSPs also permitting arithmetic operations are hard for PSPACE or even Σ_1.

Furthermore, when admitting only set operations it is much easier to prove CSPs to be complete for particular complexity classes. For each $\mathcal{O} \subseteq \{\cup, \cap, -\}$ and for $M \in \{\mathcal{P}_{\mathrm{fin}}(\mathbb{N}), [\mathbb{N}]\}$ we show that $\mathrm{CSP}(\mathrm{M}, \mathcal{O})$ is $\le_{\mathrm{m}}^{\log}$-complete for one of the classes L, P, and NP. Thus, all the problems studied in this section are considered exhaustively.

It can be proven: if an $\mathcal{O}$-sentence φ for $\mathcal{O} \subseteq \{\cup, \cap, -\}$ is true, then there is a satisfying assignment α with $W_\alpha \subseteq \bigcup_{C \in \mathrm{Const}_\varphi} C$. Hence, we obtain the following upper bound for all CSPs admitting only set operations.

▶ **Lemma 4.** *Let* $\mathcal{O} \subseteq \{-, \cup, \cap\}$ *and* $M \in \{[\mathbb{N}], \mathcal{P}_{\mathrm{fin}}(\mathbb{N})\}$. *Then* $\mathrm{CSP}\big(\mathrm{M}, \mathcal{O}\big) \in \mathrm{NP}$.

If we do not allow any operations at all, the corresponding CSP belongs to L. This can be proven by a Turing reduction to the problem USTCON, i.e., the question of whether two nodes in an undirected, finite graph are connected. This problem was shown to be in L by Reingold [13].

▶ **Lemma 5.** $\mathrm{CSP}\big(\mathrm{M}, \emptyset\big) \in \mathrm{L}$.

Thus, $\overline{\mathrm{CSP}(\mathrm{M}, \emptyset)}$ is trivially $\le_{\mathrm{m}}^{\log}$-complete for L. Nevertheless, we remark that the reduction $\overline{\mathrm{USTCON}} \le_{\mathrm{m}}^{\log} \mathrm{CSP}(\mathrm{M}, \emptyset)$ is simple. This shows that a direct proof of $\mathrm{CSP}(\mathrm{M}, \emptyset) \in \mathrm{L}$ is as difficult as a proof for $\mathrm{USTCON} \in \mathrm{L}$.

Problems admitting union only

▶ **Theorem 6.** $\mathrm{CSP}(\mathcal{P}_{\mathrm{fin}}(\mathbb{N}), \{\cup\})$ *is* $\le_{\mathrm{m}}^{\log}$*-complete for* P.

Proof. We first show $\mathrm{CSP}(\mathcal{P}_{\mathrm{fin}}(\mathbb{N}), \{\cup\}) \in \mathrm{P}$. According to lemma 3 it is sufficient to decide $\mathrm{CSP}'((\mathcal{P}_{\mathrm{fin}}(\mathbb{N}), \{\cup\})$ in polynomial time.

The following algorithm decides whether there is a satisfying assignment of variables for an input sentence φ of the form described in definition 2. For this purpose it successively computes an assignment α. Let $K = \bigcup_{C \in \mathrm{Const}_\varphi} C$.

1. For each constant C set $\alpha(C) = C$. For variables X set $\alpha(X) = K$.
2. Apply the following rules to every atom $X \cup Y = Z$.
   a. Set $\alpha(Z) = \alpha(Z) \cap (\alpha(X) \cup \alpha(Y))$.
   b. Set $\alpha(X) = \alpha(X) \cap \alpha(Z)$ and analogously $\alpha(Y) = \alpha(Y) \cap \alpha(Z)$.
   c. If $\alpha(C)$ for a constant C has been changed, return 0.
3. If $\alpha(X)$ for an $X \in \mathrm{Var}_\varphi$ has been changed in step 2, execute step 2 again.
4. Return 1.

Since the sets $\alpha(X)$ are decreased monotonically regarding the inclusion relation, there are at most $\mathrm{Var}_\varphi \cdot |\mathrm{K}|$ changes of values $\alpha(X)$ for variables X. Hence, the algorithm can be executed in polynomial time. Moreover, it can be proven that the algorithm returns 1 if and only if φ is true.

It remains to prove that $\mathrm{CSP}(\mathcal{P}_{\mathrm{fin}}(\mathbb{N}), \{\cup\})$ is $\leq_{\mathrm{m}}^{\log}$-hard for P: let MCVE be the problem of whether a monotone Boolean circuit, i.e., a Boolean circuit with solely OR- and AND-gates, outputs 1 on some fixed input. According to Greenlaw et al. [8] this problem is $\leq_{\mathrm{m}}^{\log}$-complete for P.

We simulate such a circuit by a $\{\cup\}$-sentence such that the set $\{1\}$ stands for the truth value 1, and $\emptyset$ stands for the truth value 0.

Each OR-gate can be simulated by an atom $X \cup Y = Z$ where X and Y stand for the gate's inputs, and Z stands for the gate's output.

Now we describe how AND-gates can be simulated. For an AND-gate consider the atoms $Z \cup H = X \wedge Z \cup H' = Y$ where X and Y stand for the gate's inputs, Z stands for the gate's output, and H and H' are auxiliary variables. These atoms are satisfied by an assignment α only if $\alpha(Z) = \{1\} \Rightarrow (\alpha(X) = \{1\} \wedge \alpha(Y) = \{1\})$. Note that there might indeed be an assignment of terms β with $\beta(Z) = \emptyset$ and $\beta(X) = \beta(Y) = \{1\}$. But as the circuit is monotone, this is no problem.

Finally, if G is the variable standing for the circuit's output value, we add the atom $G = \{1\}$. This completes the proof. ◀

Contrary to expectation the problem $\mathrm{CSP}([\mathbb{N}], \{\cup\})$ is – in case $\mathrm{P} \neq \mathrm{NP}$ – more difficult than the problem $\mathrm{CSP}(\mathcal{P}_{\mathrm{fin}}(\mathbb{N}), \{\cup\})$. Since the CSP over $[\mathbb{N}]$ might appear to be a restriction of the CSP over $\mathcal{P}_{\mathrm{fin}}(\mathbb{N})$, one might at first view rather expect the opposite. The $\{\cup\}$-sentences over $[\mathbb{N}]$ are indeed a restricted version of sentences over $\mathcal{P}_{\mathrm{fin}}(\mathbb{N})$ because the constants belong to a strict subset of $\mathcal{P}_{\mathrm{fin}}(\mathbb{N})$.

However, since for the CSP over $[\mathbb{N}]$ also the variables are mapped onto intervals only, we obtain greater expressive power in this specific situation.

For instance, the atom $\{0\} \cup \{2\} = X \cup Y$ expresses that X is mapped onto $\{0\}$ (resp., $\{2\}$) by a satisfying assignment if and only if Y is mapped onto $\{2\}$ (resp., $\{0\}$). Furthermore, for $X, Y \in \{\{0\}, \{2\}\}$ the atoms $\{0\} \cup X \cup Y = \{0\} \cup Z \wedge Z \cup Z' = \{0\} \cup \{2\}$ express that Z is mapped onto $\{2\}$ by any satisfying assignment if and only if X or Y is mapped onto $\{2\}$. Otherwise Z is mapped onto $\{0\}$. Hence, interpreting $\{2\}$ (resp., $\{0\}$) as the truth value 1 (resp., 0), the logical disjunction and negation can be expressed. Thus, 3SAT is reducible to $\mathrm{CSP}([\mathbb{N}], \{\cup\})$ and we obtain the following theorem.

▶ **Theorem 7.** $\mathrm{CSP}([\mathbb{N}], \{\cup\})$ *is* $\leq_{\mathrm{m}}^{\log}$*-complete for* NP.

Further CSPs admitting set operations only. The methods and results for all other CSPs permitting only set operations are similar.

So we obtain the P-completeness for $\mathrm{CSP}(\mathcal{P}_{\text{fin}}(\mathbb{N}), \{\cap\})$. Yet, as $[\mathbb{N}]$ is not closed under union, but under intersection, $\mathrm{CSP}([\mathbb{N}], \{\cap\})$ is – in contrast to $\mathrm{CSP}([\mathbb{N}], \{\cup\})$ – not NP-complete, but P-complete.

▶ **Theorem 8.** $\mathrm{CSP}(\mathcal{P}_{\text{fin}}(\mathbb{N}), \{\cap\})$ *and* $\mathrm{CSP}([\mathbb{N}], \{\cap\})$ *are* $\leq_{\mathrm{m}}^{\log}$*-complete for* P.

By use of sentences in $\mathrm{CSP}(\mathcal{P}_{\text{fin}}(\mathbb{N}), \{\cup, \cap\})$ and $\mathrm{CSP}([\mathbb{N}], \{\cup, \cap\})$ arbitrary propositional formulas can be described. Thus we obtain the following theorem.

▶ **Theorem 9.** $\mathrm{CSP}(\mathcal{P}_{\text{fin}}(\mathbb{N}), \{\cup, \cap\})$ *and* $\mathrm{CSP}([\mathbb{N}], \{\cup, \cap\})$ *are* $\leq_{\mathrm{m}}^{\log}$*-complete for* NP.

As $X = A \cup B$ can be expressed by $(X - A) - B = \emptyset \wedge A - X = \emptyset \wedge B - X = \emptyset$, and as $X = A \cap B$ holds if and only if $X = (A - (A - B))$, we obtain the first statement of the following theorem.

▶ **Theorem 10.** *Let* $\mathcal{O} \subseteq \{-, \cap, \cup\}$ *with* $- \in \mathcal{O}$.
1. $\mathrm{CSP}\big(\mathcal{P}_{\text{fin}}(\mathbb{N}), \mathcal{O}\big)$ *is* $\leq_{\mathrm{m}}^{\log}$*-complete for* NP.
2. $\mathrm{CSP}\big([\mathbb{N}], \mathcal{O}\big)$ *is* $\leq_{\mathrm{m}}^{\log}$*-complete for* NP.

Overview. The following table provides an overview over the results obtained in this section.

$\mathrm{CSP}\big(\mathrm{M}, \mathcal{O}\big)$ with	$M = \mathcal{P}_{\text{fin}}(\mathbb{N})$		$M = [\mathbb{N}]$	
	$\leq_{\mathrm{m}}^{\log}$-hard for	belongs to	$\leq_{\mathrm{m}}^{\log}$-hard for	belongs to
$\mathcal{O} = \emptyset$	L	L, 5	L	L, 5
$\mathcal{O} = \{\cup\}$	P, 6	P, 6	NP, 7	NP, 4
$\mathcal{O} = \{\cap\}$	P, 8	P, 8	P, 8	P, 8
$\mathcal{O} = \{\cup, \cap\}$	NP, 9	NP, 4	NP, 9	NP, 9
$\mathcal{O} \supseteq \{-\}$	NP, 10	NP, 4	NP, 10	NP, 4

For each of the problems the $\leq_{\mathrm{m}}^{\log}$-completeness for one of the complexity classes L, P, and NP is proven.

Additionally, it can be seen that in both situations we receive the same results for all $\mathcal{O} \subseteq \{\cup, \cap, -\}$ but $\mathcal{O} = \{\cup\}$. If it holds $\mathrm{P} \neq \mathrm{NP}$, then $\mathcal{O} = \{\cup\}$ is the only case throughout this paper where deciding a problem over $[\mathbb{N}]$ is more difficult than deciding the corresponding problem over $\mathcal{P}_{\text{fin}}(\mathbb{N})$. Contrary to that, in Section 4 there is an example for which – under the assumption $\mathrm{PSPACE} \neq \mathrm{NEXP}$ – the opposite is true.

4 CSPs Permitting Arithmetic Operations

Before we move to the consideration of some concrete problems, we prove an upper bound for all CSPs investigated in this paper:

▶ **Lemma 11.** *Let* $M \in \{\mathcal{P}_{\text{fin}}(\mathbb{N}), [\mathbb{N}]\}$ *and* $\mathcal{O} \subseteq \{+, \times, \cup, \cap, -\}$. *Then* $\mathrm{CSP}(\mathrm{M}, \mathcal{O}) \in \Sigma_1$.

Proof. The set $\{(\varphi, \alpha) \mid \varphi \in \mathrm{CSP}(\mathrm{M}, \mathcal{O}),\ \alpha$ is a satisfying assignment of variables for $\varphi\}$ is decidable. Hence, the problem $\mathrm{CSP}(\mathrm{M}, \mathcal{O})$ is a projection of a decidable set. ◀

4.1 CSPs over a Single Arithmetical Operation

In this section we consider the problems $\mathrm{CSP}(\mathcal{P}_{\text{fin}}(\mathbb{N}), \{+\})$ and $\mathrm{CSP}([\mathbb{N}], \{+\})$ as well as $\mathrm{CSP}(\mathcal{P}_{\text{fin}}(\mathbb{N}), \{\times\})$ and $\mathrm{CSP}([\mathbb{N}], \{\times\})$. All these problems turn out to be $\leq_{\mathrm{m}}^{\log}$-hard for NP. This shows that all CSPs permitting an arithmetical operation are $\leq_{\mathrm{m}}^{\log}$-hard for NP.

▶ **Definition 12.** Let $\mathrm{MSOS} =_{\mathrm{def}} \{x_1, \ldots, x_n, b \mid \exists a_1, \ldots, a_n \in \mathbb{N} : \sum_{i=1}^{n} a_i x_i = b\}$.

▶ **Lemma 13** ([1])**.** *MSOS is $\leq_{\mathrm{m}}^{\log}$-complete for* NP.

Let $M \in \{\mathcal{P}_{\mathrm{fin}}(\mathbb{N}), [\mathbb{N}]\}$. In order to see that $\mathrm{MSOS} \leq_{\mathrm{m}}^{\log} \mathrm{CSP}(\mathrm{M}, \{+\})$, consider a $\{+\}$-sentence in which the factors a_i are guessed. Using the shift-and-add technique it can be made sure that some variable S_i in that sentence is mapped onto $\{a_i x_i\}$ by each satisfying assignment. Obviously $(x_1, \ldots, x_n, b) \in \mathrm{MSOS}$ if and only if the a_i can be chosen such that $\sum_{i=1}^{n} S_i = \{b\}$.

$\sum_{i \in I} a_i x_i = b$ is equivalent to $\prod_{i=1}^{n} 2^{a_i x_i} = 2^b$. Thus, with a similar approach using the square-and-multiply technique $\mathrm{MSOS} \leq_{\mathrm{m}}^{\log} \mathrm{CSP}(\mathrm{M}, \{\times\})$ can be shown.

▶ **Theorem 14.** *The following statements hold:*
1. $\mathrm{CSP}(\mathcal{P}_{\mathrm{fin}}(\mathbb{N}), \{+\})$ *and* $\mathrm{CSP}([\mathbb{N}], \{+\})$ *are $\leq_{\mathrm{m}}^{\log}$-hard for* NP.
2. $\mathrm{CSP}(\mathcal{P}_{\mathrm{fin}}(\mathbb{N}), \{\times\})$ *and* $\mathrm{CSP}([\mathbb{N}], \{\times\})$ *are $\leq_{\mathrm{m}}^{\log}$-hard for* NP.

Yet we find different upper bounds. The problems over $\mathcal{P}_{\mathrm{fin}}(\mathbb{N})$ belong to NEXP, whereas $\mathrm{CSP}([\mathbb{N}], \{\times\}) \in \Sigma_3^{\mathrm{p}}$ and even $\mathrm{CSP}([\mathbb{N}], \{+\}) \in \mathrm{NP}$ hold. We start with the consideration of CSPs that permit the addition only.

At first we show: if there is a satisfying assignment of variables for a given $\{+\}$-sentence φ, then there is also a satisfying assignment for φ that is "small" in some sense.

▶ **Lemma 15.** *Let $\varphi \in \mathrm{CSP}'\big(\mathrm{M}, \{+\}\big)$ for $M \in \{\mathcal{P}_{\mathrm{fin}}(\mathbb{N}), [\mathbb{N}]\}$ and $n =_{def} |\varphi|$. Furthermore, let $x = \max(\bigcup_{C \in \mathrm{Const}_\varphi} C \cup \{0\})$. Then there is a satisfying assignment of variables α for φ with $\forall X \in \mathrm{Var}_\varphi : \ \max(\alpha(\mathrm{X}) \cup \{0\}) \leq \mathrm{x}2^{\mathrm{n}}$.*

Proof. The following non-deterministic algorithm successively constructs an assignment α.
1. Set $\alpha(C) = C$ for each constant C.
2. For each variable X with undefined $\alpha(X)$ occurring in an atom $Y + Z = X$ such that $\alpha(Y)$ and $\alpha(Z)$ are already defined, set $\alpha(X) = \alpha(Y) + \alpha(Z)$.
3. For each variable X with undefined $\alpha(X)$ occurring in an atom $X + Z_1 = Z_2$ such that $\alpha(Z_2)$ is defined and unequal to $\emptyset$, guess a set $S \in M$ with $\max\{S\} \leq \max(\alpha(Z_2))$ non-deterministically, and set $\alpha(X) = S$.
4. If there is a variable X such that $\alpha(X)$ has been defined in the last execution of the steps 2 and 3, then go to step 2.
5. For all variables X with undefined $\alpha(X)$ set $\alpha(X) = \emptyset$.
6. If α is a satisfying assignment, then return α.

It can be shown inductively that on at least one computation path the algorithm returns a satisfying assignment. Let $X_1, \ldots, X_{|\mathrm{Var}_\varphi|}$ denote the variables such that for $i < j$ the value $\alpha(X_i)$ is defined before $\alpha(X_j)$. Then it holds $\max(\alpha(X_i)) \leq x \cdot 2^i$ for $1 \leq i \leq |\mathrm{Var}_\varphi|$. ◀

Through this result we can proceed as follows: guess all assignments whose range is a subset of the power set of $\{0, 1, \ldots, x \cdot 2^n\}$. Test whether the respective assignment is satisfying, and return the corresponding return value.

Note that for CSPs over $[\mathbb{N}]$ only assignments with range $\subseteq \{[a, b] \mid a, b \leq x \cdot 2^n\}$ have to be considered. Hence, we obtain the following results.

▶ **Theorem 16.** *It holds that*
1. $\mathrm{CSP}(\mathcal{P}_{\mathrm{fin}}(\mathbb{N}), \{+\}) \in \mathrm{NEXP}$
2. $\mathrm{CSP}([\mathbb{N}], \{+\}) \in \mathrm{NP}$.

▶ Remark. It is also possible to show $\mathrm{CSP}([\mathbb{N}], \{+\}) \in \mathrm{NP}$ by using ILP (the problem of whether an integer linear program has a solution) as an oracle. With that approach one can even show $\mathrm{CSP}([\mathbb{N}], \{+, \cap\}) \in \mathrm{NP}$ (theorem 24).

Now we consider $\mathrm{CSP}([\mathbb{N}], \{\times\})$. Contrary to $\mathrm{CSP}([\mathbb{N}], \{+\})$ we are only able to show $\mathrm{CSP}([\mathbb{N}], \{\times\})$ belonging to Σ_3^{P}. The reason for this difference is that $[\mathbb{N}]$ is not closed under multiplication. In particular, we may not assume that the input sentences are of the simplified form described in definition 2.

Nevertheless, by investigating the multiplication of intervals we find some properties that significantly simplify deciding $\mathrm{CSP}([\mathbb{N}], \{\times\})$.

▶ **Lemma 17.** *Let $A_1, \ldots, A_m, B_1, \ldots, B_n$ be finite intervals over $\mathbb{N}$ such that it holds $\emptyset \neq \prod_{i=1}^m A_i = \prod_{i=1}^n B_i \neq \{0\}$.*
Then $\prod_{1 \leq i \leq m, |A_i|=1} A_i = \prod_{1 \leq i \leq n, |B_i|=1} B_i$ and $\prod_{1 \leq i \leq m, |A_i| \neq 1} A_i = \prod_{1 \leq i \leq n, |B_i| \neq 1} B_i$.

▶ **Lemma 18.** *Let $A_1, \ldots, A_m, B_1, \ldots, B_n$ be intervals with at least two elements each such that $\max(A_1) \leq \max(A_2) \leq \cdots \leq \max(A_m)$ and $\max(B_1) \leq \max(B_2) \leq \cdots \leq \max(B_n)$. Let furthermore $\prod_{i=1}^m A_i = \prod_{i=1}^n B_i$. Then $\max(A_m) = \max(B_n)$.*

Proof. Let $L =_{\mathrm{def}} \prod_{i=1}^m A_i$ and $R =_{\mathrm{def}} \prod_{i=1}^n B_i$. In addition, let the greatest elements of A_i (resp., B_i) be denoted by a_i (resp., b_i). Because of $L = R$ the second greatest elements of L and R are equal. Thus $\max(L - \{\max(L)\}) = \max(R - \{\max(R)\})$.

We show $\max(L - \{\max(L)\}) = \left(\prod_{i=1}^{m-1} a_i\right) \cdot (a_m - 1) = \max(L) - \prod_{i=1}^{m-1} a_i$: the right equation is obvious. Furthermore, it apparently holds that $\left(\prod_{i=1}^{m-1} a_i\right) \cdot (a_m - 1) \in L - \{\max(L)\}$.

Let $x \in L - \{\max(L)\}$. There are $x_i \in A_i$ for $i = 1, \ldots, m$ such that $x = \prod_{i=1}^m x_i$. Due to $x \neq \max(L)$ there is a j such that $x_j < a_j$. Then

$$x \leq \Big(\prod_{1 \leq i \leq m, i \neq j} a_i\Big) \cdot (a_j - 1) = \max(L) - \prod_{1 \leq i \leq m, i \neq j} a_i \leq \max(L) - \prod_{i=1}^{m-1} a_i.$$

Analogously it can be seen that $\max(R - \{\max(R)\}) = \max(R) - \prod_{i=1}^{n-1} b_i$. Hence $\max(L) - \prod_{i=1}^{m-1} a_i = \max(L - \{\max(L)\}) = \max(R - \{\max(R)\}) = \max(R) - \prod_{i=1}^{n-1} b_i$.

Because of $\max(L) = \max(R)$ we obtain $\prod_{i=1}^{m-1} a_i = \prod_{i=1}^{n-1} b_i$ and as a consequence $a_m = \frac{\max(L)}{\prod_{i=1}^{m-1} a_i} = \frac{\max(R)}{\prod_{i=1}^{n-1} b_i} = b_n$. ◀

We roughly describe a non-deterministic polynomial time algorithm satisfying the following properties: on input of a $\mathrm{CSP}([\mathbb{N}], \{\times\})$-instance the algorithm simplifies this sentence until no variables are left, and finally returns it. If and only if the input sentence is in $\mathrm{CSP}([\mathbb{N}], \{\times\})$, then there is a computation path on which a true sentence is returned.

After some preprocessing we obtain a possibly modified sentence and may neglect the sets $\emptyset$ and $\{0\}$ henceforth. Then we guess which variables stand for singletons. According to lemma 17 each atom can be split in two atoms: one for the "singleton variables and constants" and one for the other variables and constants. Hence, we obtain two sets of atoms, which can be considered independently.

For singletons we only store the exponents occurring in the prime decomposition. Two singletons are multiplied by adding the vectors of exponents componentwise. Hence, testing whether there is a satisfying assignment for the "singleton-atoms" can be done the same way as deciding $\mathrm{CSP}([\mathbb{N}], \{+\})$, which we have shown to be in NP. If there is a satisfying assignment, all "singleton-atoms" can be deleted. Otherwise return $[0, 1] = [1, 2]$ for instance.

Now consider the remaining atoms. Without loss of generality there is an atom $A_1 \times \cdots \times A_m = B_1 \times \cdots \times B_n$ such that all A_i are constants (if for each atom there are variables on both sides, then the remainig sentence is obviously true). For each variable B_j we guess an interval whose upper endpoint is $\leq \max(\bigcup_{i=1}^m A_i)$, and replace each occurrence of B_j with this interval. This approach is backed up by lemma 18 and can be repeated until no variable is left.

▶ **Lemma 19.** *There is a non-deterministic polynomial time algorithm $\mathcal{A}$ satisfying the following properties:*

- *As input $\mathcal{A}$ receives a $\{\times\}$-sentence φ whose constants are intervals.*
- *On each computation path $\mathcal{A}$ returns a $\{\times\}$-sentence ψ without any variables such that all constants are finite intervals with at least two elements each.*
- *If and only if $\varphi \in \mathrm{CSP}([\mathbb{N}], \{\times\})$, then on at least one computation path $\mathcal{A}$ returns a sentence $\psi \in \mathrm{CSP}([\mathbb{N}], \{\times\})$.*

Since $\mathcal{A}$ is a polynomial time algorithm, it holds that $|\psi| \in O(p(|\varphi|))$ for each ψ returned by the algorithm and some polynomial p.
It remains to test whether two products of intervals are equal.

▶ **Definition 20.** Let EPI be the set

$$\{(([a_1, a_1'], \ldots, [a_k, a_k']), ([b_1, b_1'], \ldots, [b_n, b_n'])) \mid a_i < a_i',\ b_i < b_i',\ \prod_{i=1}^{k}[a_i, a_i'] = \prod_{i=1}^{n}[b_i, b_i']\}\,.$$

▶ **Lemma 21.** $\mathrm{EPI} \in \Pi_2^{\mathrm{p}}$.

Proof. Let $A_1, \ldots, A_m, B_1, \ldots, B_n$ be non-empty intervals with $|A_i|, |B_j| \geq 2$.
It holds

$$\begin{aligned}\prod_{i=1}^{m} A_i = \prod_{i=1}^{n} B_i \Leftrightarrow & \forall x_1 \in A_1 \ldots \forall x_m \in A_m \forall y_1 \in B_1 \ldots \forall y_n \in B_n \\ & \exists x_1' \in B_1 \ldots \exists x_n' \in B_n \exists y_1' \in A_1 \ldots \exists y_m' \in A_m \\ & \prod_{i=1}^{m} x_i = \prod_{i=1}^{n} x_i' \wedge \prod_{i=1}^{n} y_i = \prod_{i=1}^{m} y_i'.\end{aligned}$$

Thus $\mathrm{EPI} \in \forall^{\mathrm{p}}\exists^{\mathrm{p}}\mathrm{P} = \Pi_2^{\mathrm{p}}$. ◀

▶ **Theorem 22.** $\mathrm{CSP}([\mathbb{N}], \{\times\}) \in \Sigma_3^{\mathrm{p}}$.

Proof. According to theorem 19 and lemma 21 the problem $\mathrm{CSP}([\mathbb{N}], \{\times\})$ can be decided by an NP-algorithm with Π_2^{p}-oracle. ◀

▶ Remark. Our decision algorithm for EPI tests whether two products of intervals are equal by considering all elements of the two products. Maybe this can be done more efficiently. In lemma 18 we have shown that a necessary condition for the equality of two products of intervals is that the greatest upper interval endpoint is the same in both products. If this condition can be extended to a sufficient condition that can still be tested efficiently, we would obtain a better upper bound for $\mathrm{CSP}([\mathbb{N}], \{\times\})$.

For the variant over $\mathcal{P}_{\mathrm{fin}}(\mathbb{N})$ we obtain membership in NEXP, which can be proven similarly to $\mathrm{CSP}(\mathcal{P}_{\mathrm{fin}}(\mathbb{N}), \{+\}) \in \mathrm{NEXP}$.

▶ **Theorem 23.** $\mathrm{CSP}(\mathcal{P}_{\mathrm{fin}}(\mathbb{N}), \{\times\}) \in \mathrm{NEXP}$.

4.2 Addition and Intersection

Whereas we are not able to show $\mathrm{CSP}(\mathcal{P}_{\mathrm{fin}}(\mathbb{N}), \{+, \cap\})$ to be decidable, the NP-hardness for the problem $\mathrm{CSP}([\mathbb{N}], \{+, \cap\})$ can be proven by use of integer linear programs.

Furthermore, due to the NP-hardness of $\mathrm{CSP}([\mathbb{N}], \{+\})$ also $\mathrm{CSP}([\mathbb{N}], \{+, \cap\})$ is NP-hard.

▶ **Theorem 24.** $\mathrm{CSP}([\mathbb{N}], \{+, \cap\})$ *is* $\leq_{\mathrm{m}}^{\log}$*-complete for* NP.

Proof. According to the lemmata 3 and 14 it suffices to show $\mathrm{CSP}'([\mathbb{N}], \{+, \cap\}) \in \mathrm{NP}$. Hence, let φ be a $\{+, \cap\}$-sentence whose constants are solely intervals such that each atom is of the form $X \oplus Y = Z$ for $X, Y, Z \in \mathrm{Var}_\varphi \cup \mathrm{Const}_\varphi$ and $\oplus \in \{+, \cap\}$.

During a polynomial time computable preprocessing step φ is modified non-deterministically such that the following holds: $\varphi \in \mathrm{CSP}'([\mathbb{N}], \{+, \cap\})$ if and only if on at least one computation path a sentence φ' satisfying the following conditions has been computed: there is a satisfying assignment α for φ' with $\emptyset \notin W_\alpha$, and if there is an atom $X \oplus Y = Z$ in φ' containing $\emptyset$ as a constant, then $\oplus = \cap$, $Z = \emptyset$, and $X, Y \neq \emptyset$.

The problem of testing these conditions can be solved with the help of integer linear programs (ILP). For each $R \in (\mathrm{Var}_\varphi \cup \mathrm{Const}_\varphi) - \{\emptyset\}$ we introduce two ILP-variables r_0, r_1. If R is a constant and $R = [l, u]$, set $r_0 = l$ and $r_1 = u$.

1. For each atom $X + Y = Z$ we set up the equations $x_0 + y_0 = z_0$ and $x_1 + y_1 = z_1$.
2. For each atom $X \cap Y = Z$ with $Z \neq \emptyset$ use four further variables d, e, d', e'. We express $z_0 = \max(x_0, y_0)$ and $z_1 = \min(x_1, y_1)$:
   - On $z_0 = \max(x_0, y_0)$: Add $x_0 \leq z_0$, $y_0 \leq z_0$, $z_0 = dx_0 + ey_0$, and $d + e = 1$.
   - On $z_1 = \min(x_1, y_1)$: Add $x_1 \geq z_1$, $y_1 \geq z_1$, $z_1 = d'x_1 + e'y_1$, and $d' + e' = 1$.
3. For each atom $X \cap Y = Z$ with $Z = \emptyset$ we want to express $y_1 < x_0 \vee x_1 < y_0$. Hence, we guess a bit b. If $b = 0$, we add the inequation $y_1 < x_0$. Otherwise we add $x_1 < y_0$.
4. Furthermore, for every two ILP-variables x_0 and x_1 that describe the lower and upper endpoint of some interval we add the inequation $x_0 \leq x_1$.

If and only if one of the ILPs has a solution, it holds $\varphi \in \mathrm{CSP}([\mathbb{N}], \{+, \cap\})$. ◀

4.3 Lower Bounds for CSPs Permitting One Arithmetical and One Set Operation

We present two lower bounds obtained from literature. It should be possible to improve them in at least some cases.

By use of some results by Meyer and Stockmeyer [14] the following lower bound can be proven.

▶ **Theorem 25.**
1. $\mathrm{CSP}(\mathcal{P}_{\mathrm{fin}}(\mathbb{N}), \{+, \cup\})$ *is* $\leq_{\mathrm{m}}^{\log}$*-hard for* Π_2^{P}.
2. $\mathrm{CSP}(\mathcal{P}_{\mathrm{fin}}(\mathbb{N}), \{\times, \cup\})$ *is* $\leq_{\mathrm{m}}^{\log}$*-hard for* Π_2^{P}.

These results can be shown by a reduction not making use of CSP-variables. This yields evidence that probably a better lower bound – such as the $\leq_{\mathrm{m}}^{\log}$-hardness for Σ_3^{P} – can be proven.

If beside one arithmetical operation there is also one of the two operations intersection and set difference available, we can revisit some results by McKenzie and Wagner [11] and show the corresponding problem to be PSPACE-hard.

▶ **Theorem 26.**
1. $\mathrm{CSP}(\mathcal{P}_{\mathrm{fin}}(\mathbb{N}), \{+, \cap\})$ *is* $\leq_{\mathrm{m}}^{\mathrm{p}}$*-hard for* PSPACE*.*
2. $\mathrm{CSP}(\mathcal{P}_{\mathrm{fin}}(\mathbb{N}), \{\times, \cap\})$ *is* $\leq_{\mathrm{m}}^{\mathrm{log}}$*-hard for* PSPACE*.*
3. $\mathrm{CSP}(\mathcal{P}_{\mathrm{fin}}(\mathbb{N}), \{+, -\})$ *is* $\leq_{\mathrm{m}}^{\mathrm{log}}$*-hard for* PSPACE*.*
4. $\mathrm{CSP}(\mathcal{P}_{\mathrm{fin}}(\mathbb{N}), \{\times, -\})$ *is* $\leq_{\mathrm{m}}^{\mathrm{log}}$*-hard for* PSPACE*.*

In Section 3 we observed that under the assumption $\mathrm{P} \neq \mathrm{NP}$ there are CSPs for which the variant over $[\mathbb{N}]$ is more difficult than the variant over $\mathcal{P}_{\mathrm{fin}}(\mathbb{N})$. Here we have the opposite situation.

According to corollary 24 $\mathrm{CSP}([\mathbb{N}], \{+, \cap\})$ belongs to NP. According to theorem 26, however, $\mathrm{CSP}(\mathcal{P}_{\mathrm{fin}}(\mathbb{N}), \{+, \cap\})$ is $\leq_{\mathrm{m}}^{\mathrm{p}}$-hard for PSPACE. Hence, under the assumption $\mathrm{NP} \neq \mathrm{PSPACE}$ deciding $\mathrm{CSP}(\mathcal{P}_{\mathrm{fin}}(\mathbb{N}), \{+, \cap\})$ is more difficult than deciding $\mathrm{CSP}([\mathbb{N}], \{+, \cap\})$.

4.4 Undecidability Results

As soon as both addition and multiplication are permitted, we receive undecidable, but computably enumerable problems.

More precisely, it can be shown that the problem of solving Diophantine equations, which was proven to be undecidable by Matiyasevich [2, 10], can be reduced to $\mathrm{CSP}(\mathrm{M}, \mathcal{O})$ for $M \in \{\mathcal{P}_{\mathrm{fin}}(\mathbb{N}), [\mathbb{N}]\}$ and $\{+, \times\} \subseteq \mathcal{O}$.

▶ **Theorem 27.** *Let* $M \in \{\mathcal{P}_{\mathrm{fin}}(\mathbb{N}), [\mathbb{N}]\}$ *and* $\mathcal{O} \subseteq \{\cup, \cap, -\}$*. Then* $\mathrm{CSP}\big(\mathrm{M}, \{+, \times\} \cup \mathcal{O}\big)$ *is* $\leq_{\mathrm{m}}^{\mathrm{log}}$*-complete for* Σ_1*.*

4.5 Overview

The following tables yield an overview over the results obtained in this section. For both tables there are sets of operations which do not occur in the list. For these problems we only know those bounds that follow directly from other entries in the corresponding tables (recall that whenever the set difference is permitted, then one may assume without loss of generality that also union and intersection are allowed).

The first table deals with the CSPs over $\mathcal{P}_{\mathrm{fin}}(\mathbb{N})$.

$\mathrm{CSP}\big(\mathcal{P}_{\mathrm{fin}}(\mathbb{N}), \mathcal{O}\big)$ with	hardness	member of
$\mathcal{O} = \{+\}$	$\leq_{\mathrm{m}}^{\mathrm{log}}$-hard for NP, 14	NEXP, 16
$\mathcal{O} = \{\times\}$	$\leq_{\mathrm{m}}^{\mathrm{log}}$-hard for NP, 14	NEXP, 23
$\mathcal{O} = \{+, \cup\}$	$\leq_{\mathrm{m}}^{\mathrm{log}}$-hard for Π_2^{p}, 25	Σ_1, 11
$\mathcal{O} = \{+, \cap\}$	$\leq_{\mathrm{m}}^{\mathrm{p}}$-hard for PSPACE, 26	Σ_1, 11
$\mathcal{O} = \{+, -\}$	$\leq_{\mathrm{m}}^{\mathrm{log}}$-hard for PSPACE, 26	Σ_1, 11
$\mathcal{O} \supseteq \{+, \times\}$	$\leq_{\mathrm{m}}^{\mathrm{log}}$-hard for Σ_1, 27	Σ_1, 11

The following tabular contains the results concerning CSPs over $[\mathbb{N}]$.

$\mathcal{O} = \mathrm{CSP}\big([\mathbb{N}], \mathcal{O}\big)$ with	$\leq_{\mathrm{m}}^{\mathrm{log}}$-hard for	member of
$\mathcal{O} = \{+\}$	NP, 14	NP, 16
$\mathcal{O} = \{\times\}$	NP, 14	Σ_3^{p}, 22
$\mathcal{O} = \{+, \cap\}$	NP, 14	NP, 24
$\mathcal{O} \supseteq \{+, \times\}$	Σ_1, 27	Σ_1, 11

The bounds for CSPs over $[\mathbb{N}]$ are in general lower than those for the corresponding CSPs over $\mathcal{P}_{\text{fin}}(\mathbb{N})$. If $[\mathbb{N}]$ is closed under all allowed operations, then we know the corresponding CSP to be complete for one of the classes L, NP, and Σ_1.

For the variant over $\mathcal{P}_{\text{fin}}(\mathbb{N})$ there are in almost all cases gaps between the lower and upper bound. It seems to be difficult to close these gaps.

In contrast to the section before, here remain several open questions. The following are particularly interesting:

Is $\text{CSP}(\mathcal{P}_{\text{fin}}(\mathbb{N}), \{+, \cap\})$ decidable? Is $\text{CSP}(\mathcal{P}_{\text{fin}}(\mathbb{N}), \{+\})$ complete for some class between NP and NEXP? Does EPI belong to some class of the polynomial hierarchy lower than Π_2^{p}?

References

1 E. Böhler, C. Glaßer, B. Schwarz, and K. W. Wagner. Generation problems. *Theor. Comput. Sci.*, 345(2-3):260–295, 2005.

2 M. Davis, H. Putnam, and J. Robinson. The decision problem for exponential Diophantine equations. *Annals of Mathematics*, 74(2):425–436, 1961.

3 T. Dose. Complexity of constraint satisfaction problems over finite subsets of natural numbers. Technical Report 16-031, Electronic Colloquium on Computational Complexity (ECCC), 2016.

4 T. Feder and M. Y. Vardi. The computational structure of monotone monadic snp and constraint satisfaction: A study through datalog and group theory. *SIAM J. Comput.*, 28(1):57–104, February 1999.

5 Christian Glaßer, Katrin Herr, Christian Reitwießner, Stephen D. Travers, and Matthias Waldherr. Equivalence problems for circuits over sets of natural numbers. *Theory Comput. Syst.*, 46(1):80–103, 2010.

6 C. Glaßer, B. Martin, and P. Jonsson. Circuit satisfiability and constraint satisfaction problems around skolem arithmetic. In *Proceedings of the 12th International Conference on Computability in Europe (CiE-2016)*, Lecture Notes in Computer Science. Springer Verlag, 2016. To appear.

7 C. Glaßer, C. Reitwießner, S. Travers, and M. Waldherr. Satisfiability of algebraic circuits over sets of natural numbers. *Discrete Applied Mathematics*, 158(13):1394 – 1403, 2010.

8 R. Greenlaw, H. J. Hoover, and W. L. Ruzzo. A Compendium of Problems Complete for P, 1991.

9 D. E. Knuth. All questions answered. *Notices of the AMS*, 49(3):318–324, 2002.

10 Y. V. Matiyasevich. Enumerable sets are Diophantine. *Doklady Akad. Nauk SSSR*, 191:279–282, 1970. Translation in Soviet Math. Doklady, 11:354–357, 1970.

11 Pierre McKenzie and Klaus W. Wagner. The complexity of membership problems for circuits over sets of natural numbers. *Computational Complexity*, 16(3):211–244, 2007.

12 C. M. Papadimitriou. *Computational complexity.* Addison-Wesley, Reading, Massachusetts, 1994.

13 Omer Reingold. Undirected connectivity in log-space. *J. ACM*, 55(4):17:1–17:24, September 2008.

14 L. J. Stockmeyer and A. R. Meyer. Word problems requiring exponential time: Preliminary report. In *Proceedings of the Fifth Annual ACM Symposium on Theory of Computing*, STOC '73, pages 1–9, New York, NY, USA, 1973. ACM.