Electronic Colloquium on Computational Complexity, Revision 1 of Report No. 55 (2018)

Balance Problems for Integer Circuits

Titus Dose

Abstract

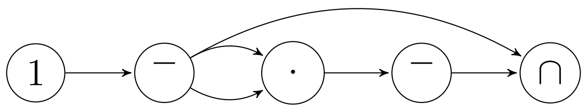
We investigate the computational complexity of *balance problems* for $\{-,\cdot\}$-circuits computing finite sets of natural numbers. These problems naturally build on problems for integer expressions and integer circuits studied by Stockmeyer and Meyer (1973), McKenzie and Wagner (2007), and Glaßer et al (2010).

Our work shows that the balance problem for $\{-,\cdot\}$-circuits is undecidable which is the first natural problem for integer circuits or related constraint satisfaction problems that admits only one arithmetic operation and is proven to be undeciable.

Starting from this result we precisely characterize the complexity of balance problems for proper subsets of $\{-,\cdot\}$. These problems turn out to be complete for one of the classes L, NL, and NP. The case where only the multiplication is allowed turns out to be particular interesting as it leads us to the general non-trivial observation that the product S of two sets with sufficiently large maxima is subbalanced (i.e., $|S| \leq \max(S)/2$), which might be interesting on its own.

1 Introduction

In 1973, Stockmeyer and Meyer [SM73] defined and studied membership and equivalence problems for *integer expressions.* They considered expressions built up from single natural numbers by using set operations ($\cup$, $\cap$, $^{-}$), pairwise addition ($+$), and pairwise multiplication ($\cdot$). For example, $\overline{\overline{1}\cdot\overline{1}}\cap\overline{1}$ describes the of primes $\mathbb{P}$.
The *membership problem for integer expressions* asks whether some given number is contained in the set described by a given integer expression, whereas the *equivalence problem for integer expressions* asks whether two given integer expression describe the same set. Restricting the set of allowed operation results in problems of different complexities.
Wagner [Wag84] studied a more succinct way to represent such expressions, namely *circuits over sets of natural numbers*, also called integer circuits. Each input gate of such a circuit is labeled with a natural number, the inner gates compute set operations and arithmetic operations ($\cup$, $\cap$, $^{-}$, $+$, $\cdot$). The following circuit with only 4 inner gates computes the set of primes.



Starting from this circuit, one can use integer circuits to express fundamental number theoretic questions: thus, a circuit describing the set of all twin primes or the set of all Sophie Germain primes can be constructed. McKenzie and Wagner [MW07] constructed a circuit C computing a set that contains 0 if and only if the Goldbach conjecture holds.
Wagner [Wag84], Yang [Yan01], and McKenzie and Wagner [MW07] investigated the complexity of membership problems for circuits over natural numbers: here, for a given circuit C, one has to decide whether a given number n belongs to the set described by C. Travers [Tra06] and

ISSN 1433-8092

Breunig [Bre07] considered membership problems for circuits over integers and positive integers, respectively. Glaßer et al [GHR$^+$10] studied *equivalence problems for circuits over sets of natural numbers*, i.e., the problem of deciding whether two given circuits compute the same set.
Satisfiability problems for circuits over sets of natural numbers, investigated by Glaßer et al [GRTW10], are a generalization of the membership problems investigated by McKenzie and Wagner [MW07]: the circuits can have *unassigned input gates* and the question is: on input of a circuit C with gate labels from $\mathcal{O} \subseteq \{\cup, \cap, ^-, +, \cdot\}$ and a natural number b, does there exist an assignment of the unassigned input gates with natural numbers such that b is contained in the set described by the circuit?
Barth et al [BBD$^+$17] investigated emptiness problems for integer circuits. Here, for both circuits with unassigned inputs and circuits without unassigned inputs, the question of whether an integer circuit computes the empty set (for some/all assignment(s) if the circuits allow unassigned inputs) is raised and investigated.
Apart from the mentioned research on circuit problems there has been work on related variants like functions computed by circuits [PD09] and constraint satisfaction problems (csp) over natural numbers [GJM17, Dos16]. The constraint satisfaction problems by Glaßer, Jonsson, and Martin [GJM17] can be considered as conjunctions of equations of integer expressions with variables standing for singleton sets of natural numbers. Here the question is whether there is an assignment of the variables such that all equations are satisfied. These constraint satisfaction problems have the peculiarity that expressions describe sets of integers whereas variables can only store singleton sets of natural numbers. Dose [Dos16] addressed this and studied constraint satisfaction problems over finite subsets of $\mathbb{N}$, consequently replaced the set complement $^-$ with the set difference $-$, and allowed the variables to describe arbitrary finite subsets of $\mathbb{N}$.

Our Model and Contributions The definition of the circuits investigated in this paper follows the definition of previous papers such as [MW07, GHR$^+$10, GRTW10, BBD$^+$17]. Yet there are some differences:
Our circuit problems are about *balanced sets* where a finite and non-empty set $S \subseteq \mathbb{N}$ is balanced if $|S| = |\{0, 1, \ldots, \max(S)\} - S|$. Analogously, S is unbalanced if $|S| \neq |\{0, 1, \ldots, \max(S)\} - S|$. That means, the maximum of a set marks the relevant area and then we ask whether there are as many elements inside the set as outside of it. As the notion of balanced sets only makes sense for finite sets, our circuits should solely compute finite sets. Due to that we replace the commonly used set complement $^-$ with the set difference $-$ as otherwise, infinite sets could be generated. Now, as the circuits only work over the domain of finite subsets of $\mathbb{N}$, it suggests itself to also allow the input gates of a circuit to compute arbitrary finite subsets of $\mathbb{N}$ and not only singleton sets (cf. Dose [Dos16] where the analogous step was made for constraint satisfaction problems).
For such circuits we ask: is there an assignment of the unassigned inputs with arbitrary finite subsets of $\mathbb{N}$ under which the circuit computes a balanced set. This problem is denoted by $\mathrm{BC}(\mathcal{O})$, where $\mathcal{O} \subseteq \{\cup, \cap, -, +, \cdot\}$ is the set of allowed operations.
The notion of balance is important in computational complexity. It occurs when considering counting classes like $\mathrm{C_{=}L}$ or $\mathrm{C_{=}P}$ for instance. There, the question is whether for some problem A there is a non-deterministic logarithmic space or polynomial-time machine M accepting A, where M accepts some input x if and only if the number of accepting paths equals the number of rejecting paths.
Balance problems for integer circuits are interesting for another reason. To our knowledge, there is no natural decision problem for integer circuits or constraint satisfaction problems over sets of natural numbers allowing only one arithmetic operation that is known to be undecidable. In this paper, however, it is shown that $\mathrm{BC}(-, \cdot)$ is undecidable.

Starting from this undecidable problem $\mathrm{BC}(-,\cdot)$, we also investigate $\mathrm{BC}(\mathcal{O})$ for arbitrary subsets of $\{-,\cdot\}$ and precisely characterize the complexity of each such problem. It turns out that all these problems are in NP. In detail, we show that $\mathrm{BC}(\cdot)$ is NL-complete, $\mathrm{BC}(-)$ is NP-complete, and $\mathrm{BC}(\emptyset) \in \mathrm{L}$.
Here, the NL-complete problem $\mathrm{BC}(\cdot)$ is particularly interesting as it leads us to the general question of whether the product $S = A \cdot B$ for two sets A and B is always subbalanced (i.e., $|S| < |\{0, 1, \ldots, \max(S)\} - S|$). We show that this holds if the maxima of A and B are sufficiently large, which is a non-trivial observation and might be interesting on its own.

2 Preliminaries

Basic Notions Let $\mathbb{N}$ denote the set of natural numbers. $\mathbb{N}^+ = \mathbb{N} - \{0\}$ is the set of positive naturals. For $n \in \mathbb{N}$ let $|n|$ denote the length of the binary representation of n (without leading zeros). The greatest common divisor of positive naturals a and b is denoted by $\gcd(a, b)$ and $\gcd(a, b)$ for arbitrary non-zero integers a and b is defined to be $\gcd(\max(a, -a), \max(b, -b))$.
We extend the arithmetical operations $+$ and $\cdot$ to sets of naturals: for $A, B \subseteq \mathbb{N}$ define $A+B = \{a + b \mid a \in A, b \in B\}$ and $A \cdot B = \{a \cdot b \mid a \in A, b \in B\}$. In contrast to previous papers, in this paper the multiplication of sets is not denoted by $\times$ but by $\cdot$. Instead, $\times$ denotes the cartesian product. Furthermore, for arbitrary sets, the operations $\cup$, $\cap$, and $-$ define the union, intersection, and set difference, respectively. The power set of a set M is denoted by $\mathcal{P}(M)$ whereas $\mathcal{P}_{\text{fin}}(M) = \{A \in \mathcal{P}(M) \mid A \text{ finite}\}$. For a finite and non-empty set S let $\max(S)$ (resp., $\min(S)$) denote the maximum (resp., minimum) number of S. Finite intervals $\{x \mid a \le x \le b\}$ for $a, b \in \mathbb{Z}$ are denoted by $[a, b]$.
L, NL, and NP denote standard complexity classes [Pap94] and RE is the set of computably enumerable problems.
For problems A and B we say that A is (logarithmic-space) many-one reducible to B if there is some (logarithmic-space) computable function f with $c_A(x) = c_B(f(x))$, where c_X for a set X is the characteristic function of X. We denote this by $A \le_{\mathrm{m}} B$ (resp., $A \le_{\mathrm{m}}^{\log} B$). Moreover, A is logarithmic-space Turing reducible to B if there exists a logarithmic-space-bounded oracle Turing machine (with one oracle tape) that accepts A with B as its oracle. This is denoted by $A \le_{\mathrm{T}} B$.
For pairs (A, B) and (C, D) with $A \cap B = C \cap D = \emptyset$ we say that (A, B) is many-one reducible to (C, D) (denoted as $(A, B) \le_{\mathrm{m}} (C, D)$) if there is a computable function f with $x \in A \Rightarrow f(x) \in C$ and $x \in B \Rightarrow f(x) \in D$. Note that if $B = \overline{A}$ and $D = \overline{C}$ this coincides with the usual many-one reducibility, i.e., $(A, \overline{A}) \le_{\mathrm{m}} (C, \overline{C}) \Leftrightarrow A \le_{\mathrm{m}} C$.
CSAT is the circuit satisfiability problem, i.e., the problem of determining whether a given boolean circuit has an assignment of the unassigned inputs that makes the output gate true. The problem is $\le_{\mathrm{m}}^{\log}$-complete for NP via a trivial reduction from SAT which itself can be shown to be $\le_{\mathrm{m}}^{\log}$-complete for NP via a construction by Cook [Coo71].

Balanced Sets A finite and non-empty set $S \subseteq \mathbb{N}$ is *balanced* (resp., *unbalanced*) if $|S| = |\{0, 1, \ldots, \max(S)\} - S|$ (resp., $|S| \ne |\{0, 1, \ldots, \max(S)\} - S|$). Intuitively spoken, $\max(S)$ defines the universe $\{0, 1, \ldots, \max(S)\}$ and then S is balanced if it contains the same number of elements as its complement. Note that the notion of balance/unbalance only makes sense if there is some maximum element defining the universe. Hence the empty set is neither balanced nor unbalanced.
The following lemma immediately follows from the definition.

Lemma 1. *Let $S \in \mathcal{P}_{\text{fin}}(\mathbb{N})$ be balanced. Then $S \ne \emptyset$ and $\max(S)$ is odd.*

Moreover, we say that S is *subbalanced* if $|S| < (\max(S) + 1)/2$ which is equivalent to $|S| \leq \max(S)/2$. As we want to investigate the complexity of balance problems with respect to deterministic logarithmic-space reductions, it is important to see that the test of whether some input set is balanced can be done in deterministic logarithmic space. Define $\mathrm{Bal} = \{S \in \mathcal{P}_{\mathrm{fin}}(\mathbb{N}) \mid S \text{ is balanced}\}$. We want to observe that $\mathrm{Bal} \in \mathrm{L}$, but we show a stronger result. For that we introduce another more general problem. For a finite and non-empty set M let $\mathrm{Bal}_M = \{S \in \mathcal{P}_{\mathrm{fin}}(\mathbb{N}) \mid M \cdot S \text{ is balanced}\}$.

Proposition 2. *For $M \in \mathcal{P}_{\mathrm{fin}}(\mathbb{N})$ non-empty it holds $\mathrm{Bal}_M \in \mathrm{L}$. In particular, $\mathrm{Bal} \in \mathrm{L}$.*

Proof. The second statement follows from the first as $\mathrm{Bal} = \mathrm{Bal}_{\{1\}}$.
It suffices to consider the cases where $M \neq \emptyset$ and $\max(M) \geq 1$. The following algorithm decides Bal_M on input of a finite set $S \subseteq \mathbb{N}$ Let n denote the length of the input. For the sake of simplicity, we assume that the elements of S are encoded in binary representation, $\max(S) \geq 1$, and $n \geq 4$.

```
1. Reject if $\log(n+1) + 2 < |\max(S)|$.
2. Let $c = 0$.
3. For $\alpha = 0, 1, \ldots, \max(M) \cdot \max(S)$:
   (a) Let $d = 0$. For $(m, s) \in \{(m', s') \mid m' \in M, s' \in S\}$:
       i. If $m \cdot s = \alpha$ and $2 \cdot c = \max(M) \cdot \max(S) + 1$, then reject.
       ii. If $m \cdot s = \alpha$ and $2 \cdot c < \max(M) \cdot \max(S) + 1$, then $d = 1$.
   (b) Let $c = c + d$.
4. If $2 \cdot c = \max(M) \cdot \max(S) + 1$, then accept. Otherwise reject.
```

Step 1 can be executed in logarithmic space. If the algorithm executes step 2, then $|\max(S)| \leq \log(n + 1) + 2 \leq 3 \cdot \log(n)$. Hence all numbers m and s considered in the loop 3 are of logarithmic length. Moreover, multiplication can be computed in deterministic logarithmic space. Apart from the multiplications and comparisons the algorithm only counts to a number at most $(\max(M){\cdot}\max(S){+}1)/2 \leq \max(M){\cdot}\max(S) < \max(M){\cdot}2^{|\max(S)|} \leq \max(M){\cdot}2^{3\cdot\log(n)} = 8 \cdot \max(M) \cdot n$, where $\max(M)$ is a constant. Hence c can be stored in logarithmic space.
If the algorithm rejects in step 1, then $\max(S) > 2^{|\max(S)|-1} > 2^{\log(n+1)+1} = 2n + 2$. As S contains at most n elements, $|S| \leq n < (\max(S) - 2)/2$ and thus $|M \cdot S| < \max(M) \cdot (\max(S) - 2)/2 + 1 < \max(M \cdot S)/2$. Consequently, $M \cdot S$ is subbalanced and $S \notin \mathrm{Bal}_M$.
In the steps 3 and 4 the algorithm accepts and rejects correctly by construction. $\square$

Definition of Circuits In previous papers such as [BBD+17] it was differentiated between completely and partially assigned circuits. As we restrict on partially assigned circuits in this paper, we define circuits in general as partially assigned circuits.
A *circuit* C is a triple (V, E, g_C) where (V, E) is a finite, non-empty, directed, acyclic graph with a designated vertex $g_C \in V$ and a topologically ordered vertex set $V \subseteq \mathbb{N}$, i.e., if $u, v \in V$ are vertices with $u < v$, then there is no edge from v to u. Here, graphs may contain multi-edges and are not necessarily connected. But we require that C is topologically ordered. Note that the test of whether a graph is topologically ordered or not is possible in deterministic logarithmic space. Consequently, we are able to check in deterministic logarithmic space whether an input graph is acyclic. Hence there is a deterministic logarithmic-space algorithm that on input of a graph tests whether the input is a circuit. Therefore, when presenting algorithms for circuits we may always assume that the input is a valid circuit.

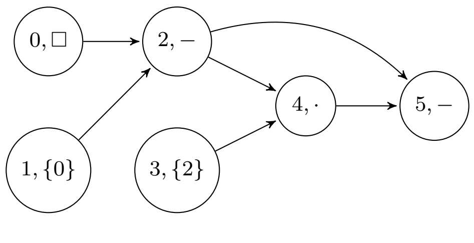
Without loss of generality we may assume that $V = \{1, \dots, r\}$ for some $r \in \mathbb{N}$ since circuits can be renumbered in logarithmic space.
Let $\mathcal{O} \subseteq \{\cup, \cap, -, +, \cdot\}$. An *$\mathcal{O}$-circuit* (or *circuit* for short if $\mathcal{O}$ is apparent from the context) is a quadruple $C = (V, E, g_C, \alpha)$ where (V, E, g_C) is a circuit whose nodes are labeled by the *labeling function* $\alpha : V \to \mathcal{O} \cup \mathcal{P}_{\text{fin}}(\mathbb{N}) \cup \{\square\}$ such that each node has indegree 0 or 2, nodes with indegree 0 have a label from $\mathcal{P}_{\text{fin}}(\mathbb{N})$ (encoded as a list of all the numbers in the set) or from $\{\square\}$, and nodes with indegree 2 have labels from $\mathcal{O}$. In the context of circuits, nodes are also called gates. A gate with indegree 0 is called *input gate*, all other nodes are *inner gates*, the designated gate g_C is also called *output gate*. Input gates with a label from $\mathcal{P}_{\text{fin}}(\mathbb{N})$ are *assigned input gates* whereas input gates with label $\square$ are *unassigned input gates*.
$\mathcal{O}$-circuits are also called *integer circuits*. If g is some gate of C with predecessors $g' < g''$ and $\alpha(g) = \otimes \in \mathcal{O}$, then we also write $g = g' \otimes g''$. Note that in case $\otimes = -$ it is important to consider the order of the operands.

The Set Computed by a Circuit For an $\mathcal{O}$-circuit C with unassigned input gates $g_1 < \dots, g_n$ and $X_1, \dots, X_n \in \mathcal{P}_{\text{fin}}(\mathbb{N})$, let $C(X_1, \dots, X_n)$ be the circuit that arises from C by modifying the labeling function α such that $\alpha(g_i) = X_i$ for every $1 \le i \le n$.
For a circuit $C = (V, E, g_C, \alpha)$ without unassigned input gates we inductively define the set $I(g; C)$ computed by a gate $g \in V$ for $g = 1, \dots, |V|$ by

$$I(g; C) = \begin{cases} \alpha(g) \subseteq \mathbb{N} & \text{if } g \text{ has indegree } 0, \\ I(g', C) \otimes I(g'', C) & \text{if } g = g' \otimes g'' \text{ and } g' < g''. \end{cases}$$

The set computed by the circuit $I(C)$ is defined to be the set computed by the output gate $I(g_C; C)$.

Basic Constructions It is convenient to introduce notations for basic constructions of circuits. For $X \in \mathcal{P}_{\text{fin}}(\mathbb{N})$ we use X as an abbreviation for the circuit $(\{1\}, \emptyset, \{1\}, 1 \mapsto X)$. For $\mathcal{O}$-circuits C, C' for some $\mathcal{O}$ and $\otimes \in \{\cup, \cap, , -+, \cdot\}$ let $C \otimes C'$ be the circuit obtained from C' and C'' by feeding their output gates to the new output gate $\otimes$ (and renumbering the nodes in a reasonable way; in particular it should be made sure that the nodes of C have lower numbers than the nodes of C'). This construction is possible in logarithmic space.
As an example, for an unassigned input gate $g = 0$, consider the circuit $C = (g - \{0\}) - ((g - \{0\}) \cdot \{2\})$, which is the following circuit



where each node is given by its number and its label. The node 5 is the output gate and it computes the set $\{1\}$ if and only if $I(2; C)$ is a set of the form $\{2^0, 2^1, 2^2, \dots, 2^r\}$ for some $r \in \mathbb{N}$.

The Main Problems Now we define the problems this paper focuses on.

Definition 3. *Let $\mathcal{O} \subseteq \{-, \cup, \cap, +, \cdot\}$ and define*

$$\mathrm{BC}(\mathcal{O}) = \{C \mid C \text{ is an } \mathcal{O}\text{-circuit with } n \text{ unassigned inputs and there exist } X_1, \dots, X_n \in \mathcal{P}_{\text{fin}}(\mathbb{N}) \text{ such that } I(C(X_1, \dots, X_n)) \text{ is balanced}\}.$$

For the rest of the paper we will study the complexity of the problems $\mathrm{BC}(\mathcal{O})$ for $\mathcal{O} \subseteq \{-,\cdot\}$. In order to prove $\mathrm{BC}(\cap)$ to be $\leq_{\mathrm{m}}^{\log}$-hard for NL we need the following NL-complete problem investigated by McKenzie and Wagner [MW07]

$$\mathrm{MC}(\cap) = \{(C,b) \mid C \text{ is an } \cap\text{-circuit whose inputs are all assigned and have labels from } \{X \subseteq \mathbb{N} \mid |X| = 1\},\ b \in I(C)\}.$$

The following lemma follows from the definition.

Lemma 4. *For $\mathcal{O} \subseteq \mathcal{O}' \subseteq \{-,\cdot\}$ it holds $\mathrm{BC}(\mathcal{O}) \leq_{\mathrm{m}}^{\log} \mathrm{BC}(\mathcal{O}')$.*

Therefore, each lower bound for a problem $\mathrm{BC}(\mathcal{O})$ shown in this paper implies the same lower bound for all problems $\mathrm{BC}(\mathcal{O}')$ for arbitrary $\mathcal{O}' \supseteq \mathcal{O}$.
We use the following abbreviations if confusions are impossible: we write g or $I(g)$ for $I(g;C)$, where C is a circuit and g is a gate of C; we write C for $I(C)$, where C is a circuit; we write $\mathrm{BC}(-,\cdot)$ for $\mathrm{BC}(\{-,\cdot\})$ and the like.

3 Set Difference and Multiplication Lead to Undecidability

This section contains our main result: the undecidability of $\mathrm{BC}(-,\cdot)$. According to the Matiyasevich-Robinson-Davis-Putnam theorem [Mat70, DPR61] the problem of determining whether there is a solution for a given Diophantine equation is RE-complete. It can be derived by standard arguments that also the following problem is RE-complete (with regard to $\leq_{\mathrm{m}}$).

$$\mathrm{DE} = \{(p(x_1,\ldots,x_n), q(x_1,\ldots,x_n)) \mid \exists a_1,\ldots,a_n \in \mathbb{N}^+,\ p(a_1,\ldots,a_n) = q(a_1,\ldots,a_n) \text{ for multivariate polynomials } p \text{ and } q \text{ with coefficients from the positive naturals}\}.$$

Reducing this problem to $\mathrm{BC}(-,\cdot)$ shows the following theorem.

Theorem 5. *$\mathrm{BC}(-,\cdot)$ is RE-complete.*

Let for the remainder of this section $\mathcal{O} = \{-,\cdot\}$ unless stated differently. For the sake of brevity, we make use of intersection gates but note that $A \cap B$ is just an abbreviation for $A-(A-B)$. Further abbreviated notations are $A - \bigcup_{i=1}^n B_i$ for $(\ldots((A-B_1)-B_2)-\ldots)-B_n$ and $A - (\bigcup_{i=1}^n B_i - \{1\})$ for $(\ldots((A-(B_1-\{1\}))-(B_2-\{1\}))-\ldots-(B_n-\{1\})$.

In order to prove Theorem 5 we define a slightly different version of the problem $\mathrm{BC}(-,\cdot)$ which can be reduced to the original version in logarithmic space.

Definition 6. *Define*

$$\mathrm{BC}'(\mathcal{O}) = \{(C,Q) \mid C \textit{ is a partially assigned } \mathcal{O}\textit{-circuit, } Q \textit{ is a subset of the nodes of } C, \textit{ and there exist } X_1,\ldots,X_n \in \mathcal{P}_{\mathrm{fin}}(\mathbb{N}^+) \textit{ such that } I(C(X_1,\ldots,X_n)) \textit{ is balanced and } I(K;C(x_1,\ldots,x_n)) = \{1\} \textit{ for all } K \in Q\}.$$

For the sake of simplicity, we call instances of $\mathrm{BC}'(\mathcal{O})$ $\mathcal{O}$-circuits as well.

Lemma 7. *The following hold.*

1. *For $K \in \mathcal{P}_{\mathrm{fin}}(\mathbb{N})$ with $\kappa := \max(K) \geq 3$ it holds $|K\cdot K\cdot K| < \kappa^3/2$.*
2. *$\mathrm{BC}'(\mathcal{O}) \leq_{\mathrm{m}}^{\log} \mathrm{BC}(\mathcal{O})$ for $\mathcal{O} = \{-,\cdot\}$.*

Proof. We argue for statement 1. Due to $K \cdot K \cdot K = \bigcup_{l=1}^{3}\{i \cdot j \cdot k \mid i,j,k \in K, |\{i,j,k\}| = l\}$ we obtain

$$|K \cdot K \cdot K| \le \binom{\kappa}{3} + 2 \cdot \binom{\kappa}{2} + \kappa = \frac{\kappa(\kappa-1)(\kappa-2) + 6\kappa(\kappa-1) + 6\kappa}{6}$$
$$= \frac{\kappa^3 + 3\kappa^2 + 2\kappa}{6} < \frac{\kappa^3 + \frac{3}{2}\kappa^3 + \frac{1}{2}\kappa^3}{6} = \frac{\kappa^3}{2}.$$

Now we argue for statement 2. Let C be a partially assigned $\mathcal{O}$-circuit with output node g_C and let Q be a subset of the nodes of C. Starting with this circuit, we build a new circuit and denote this modified circuit by C':
For all assigned and unassigned input nodes g add a node g' of type $-$ which computes the set $g - \{0\}$ and replace all edges (g, h) with (g', h). Then add a new output node $g_{C'} = g_C \cdot \prod_{K \in Q}(K \cdot K \cdot K)$.
It remains to show that $(C, Q) \in \mathrm{BC}'(\mathcal{O})$ if and only if $C' \in \mathrm{BC}(\mathcal{O})$.
Assume $(C, Q) \in \mathrm{BC}'(\mathcal{O})$. Hence there is an assignment with elements of $\mathcal{P}_{\text{fin}}(\mathbb{N}^+)$ such that under this assignment, g_C is balanced and $K = \{1\}$ for all $K \in Q$. Then by construction of C' there is an assignment under which $g_{C'}$ is balanced.

Conversely, let C' be balanced under some assignment. Then without loss of generality all assigned inputs do not contain 0 and all unassigned inputs are mapped to a set not containing 0 by the mentioned assignemnt. Due to that it suffices to show that $K = \{1\}$ for all $K \in Q$ under this assignment. By construction, $0 \notin K$. Assume $K \neq \{1\}$ for some K. As $K = \emptyset$ leads to an empty output set and due to Lemma 1 also $\max(K) = 2$ does not lead to a balanced output set, we have $\kappa = \max(K) \ge 3$ and statement 1 can be applied.
We show that for an arbitrary finite set M the set $M \cdot K \cdot K \cdot K$ is not balanced, which yields a contradiction. For $M = \emptyset$ and $\max(M) = 0$ this assertion is true. Consider the case $\max(M) \ge 1$ and $0 \notin M$. Here it holds that $M \cdot K \cdot K \cdot K$ contains less than $\frac{\kappa^3 \max(M)}{2}$ elements, the maximum of this set is $\max(M) \cdot \kappa^3$ and thus $M \cdot K \cdot K \cdot K$ is not balanced. ☐

Before proving Theorem 5 we introduce some $\mathcal{O}$-circuits which will be used extensively as components of circuits expressing Diophantine equations.

Lemma 8. *For every finite $P = \{p_1, \dots, p_n\} \subseteq \mathbb{P}$ with $n = |P| \ge 1$ there is an $\mathcal{O}$-circuit (C_P, Q_P) containing gates $g_P^1, \dots, g_P^n$ satisfying the following properties:*

1. *For an arbitrary assignment with values from $\mathcal{P}_{\text{fin}}(\mathbb{N}^+)$ it holds*

$$\forall_{K \in Q_P}\ K = \{1\} \quad \Rightarrow \quad \exists_{m \in \mathbb{N}}\ \forall_{i=1,\dots,n}\ g_P^i = \{1, p_i, \dots, p_i^m\}.$$

2. *For each $m \in \mathbb{N}$ there is an assignment with values from $\mathcal{P}_{\text{fin}}(\mathbb{N}^+)$ under which $g_P^i = \{1, p_i, \dots, p_i^m\}$ and $K = \{1\}$ for all $K \in Q_P$.*

Proof. We construct (C_P, Q_P) as follows:

- For each $p \in P$ insert an input gate X_p and gates $h_p = X_p - (X_p \cdot \{p\})$ and $h'_p = (\{1, p\} \cdot X_p) - (X_p - \{1\})$. Put all the nodes h_p into Q_P.
- Similarly, for $k \in \{p_1 \cdot p_2, p_2 \cdot p_3, \dots, p_{n-1} \cdot p_n\}$ insert an input gate X_k and gates $h_k = X_k - (X_k \cdot \{k\})$ and $h'_k = (\{1, k\} \cdot X_k) - (X_k - \{1\})$. Insert all the nodes h_k into Q_P.
- For each $k = p_i \cdot p_{i+1}$ with $i \in \{1, \dots, n-1\}$ add a node $\gamma_k = h'_k - \big((h'_{p_i} \cdot h'_{p_{i+1}}) - \{1\}\big)$ and let Q_P contain all these nodes.

- Denote $g_P^i = X_{p_i}$.

We now argue that the conditions 1 and 2 are satisfied.

1. Choose an arbitrary assignment with values from $\mathcal{P}_{\text{fin}}(\mathbb{N}^+)$ and assume $K = \{1\}$ for all $K \in Q_P$. Then for $\alpha \in \{p_1, \ldots, p_n, p_1 \cdot p_2, p_2 \cdot p_3, \ldots, p_{n-1} \cdot p_n\}$ it holds

$$X_\alpha - X_\alpha \cdot \{\alpha\} = \{1\} \tag{1}$$

and in particular, $1 \in X_\alpha$.
Assume there is some $\beta \in X_\alpha$ such that β is no power of α. Then there are $l \in \mathbb{N}$ and $\alpha' \geq 2$ with $\beta = \alpha^l \cdot \alpha'$ and $\alpha \nmid \alpha'$. Choose β such that $l = 0$ or $\alpha^{l-1} \cdot \alpha' \notin X_\alpha$. Then due to (1) we obtain $\beta \in X_\alpha \cdot \{\alpha\}$. If $l = 0$, we have $\alpha \mid \alpha'$, a contradiction. Otherwise we obtain $\alpha^{l-1} \cdot \alpha' \in X_\alpha$, which is a contradiction to the choice of β. Thus X_α only contains powers of α.
Now, choose $l \in \mathbb{N}^+$ with $\alpha^l \in X_\alpha$ (if there is none, then $X_\alpha = \{1\}$). Then due to (1) we have $\alpha^l \in X_\alpha \cdot \{\alpha\}$ and thus $\alpha^{l-1} \in X_\alpha$. Hence each X_α is of the form $\{1, \alpha, \ldots, \alpha^{m_\alpha}\}$ for some $m_\alpha \in \mathbb{N}$. As a consequence $h'_\alpha = \{1, \alpha^{m_\alpha + 1}\}$.
Now choose $k = p_i \cdot p_{i+1}$ for some i. As $\gamma_k = \{1\}$ we have

$$k^{m_k+1} = p_i^{m_k+1} \cdot p_{i+1}^{m_k+1} \in (\{1, p_i^{m_{p_i}+1}\} \cdot \{1, p_{i+1}^{m_{p_{i+1}}+1}\}),$$

which yields $m_k = m_{p_i} = m_{p_{i+1}}$. Thus there exists m such that for each $i \in \{1, \ldots, n\}$ it holds $g_P^i = \{1, p_i, \ldots, p_i^m\}$.
2. Let $m \in \mathbb{N}$ and choose the assignment with $X_\alpha = \{1, \alpha, \ldots, \alpha^m\}$ for $\alpha \in \{p_1, \ldots, p_n, p_1 \cdot p_2, p_2 \cdot p_3, \ldots, p_{n-1} \cdot p_n\}$.
It follows immediately that $h_\alpha = \{1\}$ and $h'_\alpha = \{1, \alpha^{m+1}\}$. Consequently, for $k \in \{p_1 \cdot p_2, p_2 \cdot p_3, \ldots, p_{n-1} \cdot p_n\}$ it holds

$$\gamma_k = \{1, p_i^{m+1} \cdot p_{i+1}^{m+1}\} - \{p_i^{m+1}, p_{i+1}^{m+1}, p_i^{m+1} \cdot p_{i+1}^{m+1}\} = \{1\},$$

which proves statement 2. □

Building upon this construction we extend these circuits and receive the following statement.

Lemma 9. *For every finite $P = \{p_1, \ldots, p_n\} \subseteq \mathbb{P}$ with $n = |P| \geq 1$ there is an $\mathcal{O}$-circuit (D_P, Q_P) with gates $g_P^0, g_P^1, \ldots, g_P^n$ satisfying the following properties:*

1. *For an arbitrary assignment with values from $\mathcal{P}_{\text{fin}}(\mathbb{N}^+)$ it holds*

   $\forall_{K \in Q_P}\ K = \{1\} \quad \Rightarrow \quad \exists_{m \in \mathbb{N}^+} \forall_{i=0,\ldots,n}\ |g_P^i| = m^i$, *$1 \in g_P^i$, and the prime divisors of numbers in g_P^i are all in P.*

2. *For each $m \in \mathbb{N}^+$ there is an assignment with values from $\mathcal{P}_{\text{fin}}(\mathbb{N}^+)$ under which $|g_P^i| = m^i$ and $1 \in g_P^i$ for all i, the prime divisors of numbers in g_P^i are all in P, and $K = \{1\}$ for all $K \in Q_P$.*

Proof. The lemma basically follows from Lemma 8: let (C_P, Q_P) be a circuit according to that lemma. As —in case $K = \{1\}$ for all $K \in Q_P$— any two numbers $a \in g_P^i$ and $b \in g_P^j$ for $i \neq j$ are relatively prime, it holds $|g_P^i \cdot g_P^j| = |g_P^i| \cdot |g_P^j|$. Under repeated application of this argument it can be shown that adding nodes computing $\prod_{i=1}^{j} g_P^i$ for $j = 1, \ldots, n$ and a node $g_P^0 = \{1\}$ leads to a circuit which satisfies the statement. □

Proof of Theorem 5. Due to Lemma 7 it suffices to show the reduction

$$\mathrm{DE} \leq_{\mathrm{m}} \mathrm{BC}'(-,\cdot).$$

Instead of showing this reduction directly we define an intermediate problem, the cardinality circuit problem CC given by

$\{(C, Q, s, t) \mid C = (V, E, g_C, \alpha)$ is a $\{-,\cdot\}$-circuit, $Q \subseteq V$, $s, t \in V$, and there exists an assignment with values from $\mathcal{P}_{\mathrm{fin}}(\mathbb{N}^+)$ under which

1. $|I(s)| = |I(t)|$
2. $1 \in I(s) \cap I(t)$
3. $I(K) = \{1\}$ for all $K \in Q$
4. $I(s)$ and $I(t)$ only contain numbers whose prime divisors are all > 3.$\}$

Moreover, define

$\mathcal{C} = \{(C, Q, s, t) \mid C = (V, E, g_C, \alpha)$ is a $\{-,\cdot\}$-circuit, $Q \subseteq V$, $s, t \in V$ such that for all assignments with values from $\mathcal{P}_{\mathrm{fin}}(\mathbb{N}^+)$ satisfying $\forall_{K\in Q}\ K = \{1\}$ it holds that $s \geq t$ and that s and t solely contain numbers whose prime divisors are all greater than 3$\}$,

i.e., for all circuits in $\mathcal{C}$ each relevant assignment maps s to a set with higher cardinality than the set it maps t to and each relevant assignment maps s and t to sets that do not contain any numbers with prime divisors ≤ 3. For the sake of simplicity, we also call tuples (C, Q, s, t) $\{-,\cdot\}$-circuits.
The proof will be given in the two steps

1. $(\mathrm{DE}, \overline{\mathrm{DE}}) \leq_{\mathrm{m}} (\mathrm{CC}, \overline{\mathrm{CC}} \cap \mathcal{C})$
2. $(\mathrm{CC}, \overline{\mathrm{CC}} \cap \mathcal{C}) \leq_{\mathrm{m}} (\mathrm{BC}'(-,\cdot), \overline{\mathrm{BC}'(-,\cdot)})$.

That means that the function composition of the two reduction functions yields a reduction $\mathrm{DE} \leq_{\mathrm{m}} \mathrm{BC}'(-,\cdot)$.

1. Roughly speaking, the first of the two reductions generates a circuit computing two sets whose cardinalities express the results of two multivariate polynomials.
Let q and q' be multivariate polynomials with variables $x_1, \dots, x_n$. Then for any assignment with positive natural numbers $a_1, \dots, a_n$ it holds $q(a_1, \dots, a_n) = q'(a_1, \dots, a_n)$ if and only if $q(a_1, \dots, a_n)^2 + q'(a_1, \dots, a_n)^2 = 2 \cdot q(a_1, \dots, a_n) \cdot q'(a_1, \dots, a_n)$. Observe that here because of $(q(a_1, \dots, a_n) - q'(a_1, \dots, a_n))^2 \geq 0$ we have $q(a_1, \dots, a_n)^2 + q'(a_1, \dots, a_n)^2 \geq 2 \cdot q(a_1, \dots, a_n) \cdot q'(a_1, \dots, a_n)$ for any assignment.
Due to that we may assume that we are given multivariate polynomials q and q' with variables $x_1, \dots, x_n$ such that $q \geq q'$ for all assignments of the variables with values from $\mathbb{N}^+$. Let

$$q = \sum_{i=1}^{m} a_i \cdot \prod_{j=1}^{n} x_j^{d_{i,j}} \quad \text{and} \quad q' = \sum_{i=1}^{m'} a'_i \prod_{j=1}^{n} x_j^{d'_{i,j}}$$

for positive numbers m, m', a_i, and a'_i and natural numbers $d_{i,j}$ and $d'_{i,j}$. Moreover, for each variable x_j define $e_j = \max(\{d_{1,j}, \dots, d_{m,j}, d'_{1,j}, \dots, d'_{m',j}\})$, i.e., e_j denotes the maximum exponent of the variable x_j occurring in a monomial of q or q'.
We now successively build the output circuit (C, Q, s, t). For the single steps we give some intuition which is written italic.

1. For each variable x_j select a set $P_j = \{p_{j,1}, \ldots, p_{j,e_j}\}$ of primes greater than 3 such that $|P_j| = e_j$ and $P_j \cap P_{j'} = \emptyset$ for $j \neq j'$. Then insert a circuit (C_{P_j}, Q_{P_j}) according to Lemma 9 and for all P_j, insert the nodes of Q_{P_j} into Q.

   We will make use of the notation of Lemma 9, in particular of the nodes $g^0_{P_j}, \ldots, g^{e_j}_{P_j}$. That means, for any assignment which satisfies $K = \{1\}$ for all $K \in Q \supseteq Q_{P_j}$, it holds $|g^i_{P_j}| = m^i_j$ for $m_j \in \mathbb{N}^+$ and for all $i \leq e_j$. Moreover, in that case all primes dividing some number of $g^i_{P_j}$ are in P_j.

   For intuition, think of the node $g^i_{P_j}$ as a set whose cardinality describes x^i_j.

2. (a) Choose a prime $p > 3$ not used before and insert gates $h_i = \{1, p, \ldots, p^{a_i - 1}\} \cdot \prod_{j=1}^n g^{d_{i,j}}_{P_j}$ for all $i = 1, \ldots, m$.
   Loosely speaking, the cardinality of h_i describes the value of the i-th monomial of q.

   (b) For each node h_i choose a prime $p_i > 3$ not used before and insert a node $h'_i = (\{1, p_i\} \cdot h_i) - \big(h_i - \{1\}\big)$.
   As addition is supposed to be simulated by union, we need to make sure that the sets standing for distinct monomials are disjoint. Still, for a technical reason we have to keep 1 in each set. So the idea is to let h'_i consist of 1 and a copy of h_i multiplied with an additional prime factor.

   (c) For $i = 1, \ldots, m$ add an unassigned input node z_q. Finally add nodes $z_q - \big(\bigcup_{i=1}^m h'_i - \{1\}\big)$ and $h'_i - (z_q - \{1\})$ (for $i = 1, \ldots, m$) and insert these nodes into Q.
   Roughly speaking, z_q describes the value of $q + 1$ as it is the union of all the h'_i.

3. Do the same as in step 2 but for q'. In particular a node $z_{q'}$ is added.

4. Define $s = z_q$ and $t = z_{q'}$.

First, observe that the function $(q, q') \mapsto (C, Q, s, t)$ is computable. In order to show

$$(q, q') \in \mathrm{DE} \Rightarrow (C, Q, s, t) \in \mathrm{CC} \quad \text{and} \quad (q, q') \notin \mathrm{DE} \Rightarrow (C, Q, s, t) \in \overline{\mathrm{CC}} \cap \mathcal{C}$$

we make the following central observation.

Claim 10. *1. For each $y_1, \ldots, y_n \in \mathbb{N}^+$ there is an assignment of the circuit (C, Q) with values from $\mathcal{P}_{\text{fin}}(\mathbb{N}^+)$ such that s (resp., t) consists of $1 + q(y_1, \ldots, y_n)$ (resp., $1 + q'(y_1 \ldots, y_n)$) numbers whose prime divisors are greater than 3, $1 \in s \cap t$, and $K = \{1\}$ for all $K \in Q$.*

2. If $K = \{1\}$ for all $K \in Q$ under some assignment with values from $\mathcal{P}_{\text{fin}}(\mathbb{N}^+)$, then there are $y_1, \ldots, y_n \in \mathbb{N}^+$ such that $|s| = 1 + q(y_1, \ldots, y_n)$ and $|t| = 1 + q'(y_1, \ldots, y_n)$ and s and t solely contain numbers whose prime divisors are all greater than 3.

Proof of Claim 10. 1. Let $y_1, \ldots, y_n \in \mathbb{N}^+$. Then according to Lemma 9 the inputs of the circuits (C_{P_j}, Q_{P_j}) can be chosen such that

- $K = \{1\}$ for all $K \in Q_{P_j}$,
- $|g^i_{P_j}| = y^i_j$ and $1 \in g^i_{P_j}$ for $i = 1, \ldots, e_j$, and
- all prime divisors of numbers in $g^i_{P_j}$ are in P_j and greater than 3.

As the set of primes chosen for two different variables are disjoint and in step 2b we select primes not used before, the gate h_i associated with the monomial $a_i \cdot \prod_{j=1}^{n} x_j^{d_{i,j}}$ contains $a_i \cdot \prod_{j=1}^{n} y_j^{d_{i,j}}$ elements that only have prime divisors greater than 3. Furthermore, as $1 \in h_i$ for all i, we have $|h'_i| = 2 \cdot |h_i| - (|h_i| - 1) = |h_i| + 1$. Moreover, observe that $h'_i \cap h'_j = \{1\}$ for arbitrary $i \neq j$. For the node z_q choose the assignment $\bigcup_{i=1}^{m} h'_i$. Consequently, $1 \in z_q$ and

$$|z_q| = 1 + \sum_{i=1}^{m} \underbrace{(|h'_i| - 1)}_{=|h_i|} = 1 + \sum_{i=1}^{m} a_i \cdot \prod_{j=1}^{n} x_j^{d_{i,j}} = 1 + q(y_1, \ldots, y_n).$$

Since we do the same for the nodes associated with the polynomial q' we have $|z_{q'}| = 1 + q'(y_1, \ldots, y_n)$ and $1 \in z_{q'}$. Observe that the prime divisors of numbers in z_q and $z_{q'}$ are greater than 3.
It remains to observe that all nodes added into Q in step 2c compute the set $\{1\}$. This holds since z_q was chosen to be $\bigcup_{i=1}^{m} h'_i$.

2. Consider an assignment with $K = \{1\}$ for all $K \in Q$. Then according to Lemma 9 for each variable x_j we have $|g^i_{P_j}| = y^i_j$ for some $y_j \in \mathbb{N}^+$ and $i = 0, \ldots, e_j$ and all numbers in these gates solely have prime divisors in P_j. As the P_j are pairwise disjoint and in step 2b we select primes not used before, we obtain $|h_i| = a_i \cdot \prod_{j=1}^{n} y_j^{d_{i,j}}$ and $|h'_i| = |h_i| + 1$. As $h'_i \cap h'_j = \{1\}$ for $i \neq j$ and each h'_i contains 1, it holds $|z_q| = 1 + \sum_{i=1}^{n} a_i \cdot \prod_{j=1}^{n} y_j^{d_{i,j}} = 1 + q(y_1, \ldots, y_n)$. Similarly we obtain $|z_{q'}| = 1 + q'(y_1, \ldots, y_n)$.
It remains to argue that under the given assignment s and t do not contain any numbers with prime divisors ≤ 3. Obviously, the assigned inputs only compute sets whose elements solely have prime divisors greater than 3. By our construction and Lemma 9 the same holds for all nodes $g^i_{P_j}$. As a consequence, all nodes h_i and h'_i have the same property and due to $z_q - \left(\bigcup_{i=1}^{m} h'_i - \{1\}\right) = \{1\}$ (cf. Step 2c) this also holds for $z_q = s$. An analogous argumentation shows that also t does not contain any numbers with prime divisors ≤ 3. $\square$

Claim 11. *1. If $(q, q') \in \mathrm{DE}$, then $(C, Q, s, t) \in \mathrm{CC}$.*

2. If $(q, q') \notin \mathrm{DE}$, then $(C, Q, s, t) \in \overline{\mathrm{CC}} \cap \mathcal{C}$.

Proof of Claim 11. The first implication follows from Claim 10. For the second implication note that $q \geq q'$ as has been argued above. Due to that and Claim 10 it holds $(C, Q, s, t) \in \mathcal{C}$. Since $(q, q') \notin \mathrm{DE} \Rightarrow (C, Q, s, t) \notin \mathrm{CC}$ by Claim 10, the proof is complete. $\square$

2. Now we show $(\mathrm{CC}, \overline{\mathrm{CC}} \cap \mathcal{C}) \leq_{\mathrm{m}} (\mathrm{BC}'(-,\cdot), \overline{\mathrm{BC}'(-,\cdot)})$. The following algorithm computes the reduction function. The italic comments are supposed to give some intuition.

1. Let a circuit (C, Q, s, t) be given. We construct a circuit (C', Q') by successively updating the given circuit.

2. Add new unassigned input gates X and X'. Insert the following nodes into Q':

$$\{1,2\} \cdot s - (X - \{1\}), \tag{2}$$
$$\{1,2\} \cdot t - (X - \{1\}), \tag{3}$$
$$\{1,2\} \cdot (X - s) - \big((X' \cup (X - s)) - \{1\}\big), \tag{4}$$
$$X' - \{2\} \cdot (X - s). \tag{5}$$

The basic idea is as follows: X is supposed to be an interval containing s and t and X' basically encodes the set $X - s$ where this set is made disjoint to t by multiplying it with $\{2\}$. As $|s| \geq |t|$, the set $X' \cup t$ is subbalanced. But if $|s| = |t|$, then $X' \cup t$ is almost balanced. Adding the element $\max(X')+1$ would make the set balanced. This element is generated in the next step.

3. Let $p_1 = 2$ and $p_2 = 3$. Add a circuit $(C_{\{p_1,p_2\}}, Q_{\{p_1,p_2\}})$ according to Lemma 8. Put all nodes of $Q_{\{p_1,p_2\}}$ into Q'. Add a node $g = \left(g^2_{\{p_1,p_2\}} \cdot \{1,3\}\right) - \left(g^2_{\{p_1,p_2\}} - \{1\}\right)$.

4. Add a new unassigned input node O and the following nodes which are also added to Q':

$$O - \Big(\big(X' \cup t \cup g\big) - \{1\}\Big), \tag{6}$$
$$X' - (O - \{1\}), \tag{7}$$
$$t - (O - \{1\}), \tag{8}$$
$$g - (O - \{1\}). \tag{9}$$

Thus, roughly speaking, the output set O equals $X' \cup t \cup g$ and is only balanced if $|t| \geq |s|$.

5. Let O be the output node of the circuit (C', Q').

Claim 12. *If $(C, Q, s, t) \in \mathrm{CC}$, then $(C', Q') \in \mathrm{BC}'(-, \cdot)$.*

Proof of Claim 12. Let $(C, Q, s, t) \in \mathrm{CC}$. Then there is some assignment with

- $|s| = |t|$,
- $1 \in s \cap t$,
- $K = \{1\}$ for all $K \in Q$, and
- s and t only contain numbers whose prime divisors are all greater than 3.

We now consider the circuit (C', Q') under an assignment satisfying the four conditions just mentioned. Moreover, we choose the input of $C_{\{p_1,p_2\}}$, X, X', and O such that

- $g = \{1, 3^m\}$ for m minimal with $4 \cdot (\max(s \cup t) + 1) < 3^m$ and $4 \mid 3^m - 1$ and all nodes in $Q_{\{p_1,p_2\}}$ compute $\{1\}$ (such an assignment exists by Lemma 8),
- $X = \{x \mid 1 \leq x \leq (3^m - 1)/2\}$,
- $X' = \{1\} \cup \{2\} \cdot (X - s)$, and
- $O = X' \cup t \cup g = \big((\{2\} \cdot (X - s))\big) \cup t \cup \{3^m\}$.

In order to see $K = \{1\}$ for all $K \in Q'$ it remains to consider the nodes added in the steps 2 and 4. Due to the choice of g and X it holds $\max(X) > 2 \cdot \max(s \cup t)$ and thus the nodes defined in (2) and (3) compute $\{1\}$. The choice of X' immediately implies that the node defined in (5) computes $\{1\}$. Now we argue for the node defined in (4): As $X' = \{1\} \cup \{2\} \cdot (X - s)$ we have $\{1,2\} \cdot (X - s) - \big((X' \cup (X - s)) - \{1\}\big) = \{1,2\} \cdot (X - s) - \big((\{1,2\} \cdot (X - s)) - \{1\}\big) = \{1\}$. The nodes defined in (6), (7), (8), and (9) compute $\{1\}$ by the choice of g, X, X', and O.
As s and t only contain numbers whose prime divisors are > 3, the sets $\{2\} \cdot (X - s)$, t, and $\{3^m\}$ are disjoint. Hence,

$$|O| = \max(X) - |s| + |t| + 1 = \max(X) + 1 = \frac{\max(O) - 1}{2} + 1 = \frac{\max(O) + 1}{2}$$

and thus O is balanced. □

Claim 13. *If* $(C, Q, s, t) \in \overline{\mathrm{CC}} \cap \mathcal{C}$*, then* $(C', Q') \in \overline{\mathrm{BC}'(-,\cdot)}$.

Proof of Claim 13. For a contradiction, assume that $(C, Q, s, t) \in \overline{\mathrm{CC}} \cap \mathcal{C}$ and $(C', Q') \in \mathrm{BC}'(-,\cdot)$. As the second circuit is an extended version of the first circuit, both circuits can now be considered under the same assignment. Choose an assignment with values from $\mathcal{P}_{\mathrm{fin}}(\mathbb{N}^+)$ under which O is balanced and all $K \in Q'$ satisfy $K = \{1\}$. As by construction $Q \subseteq Q'$, we have $K = \{1\}$ for $K \in Q$.
As in particular the nodes defined in (2) and (3) compute $\{1\}$, we obtain $1 \in s \cap t$, $X \supseteq \{1,2\} \cdot s \cup \{1,2\} \cdot t$, and in particular $s \subseteq X$ and $\max(X) > \max(s) \geq 1$. As $\{1,2\} \cdot (X - s) - \big((X' \cup (X - s)) - \{1\}\big) = \{1\}$ (cf. (4)), it holds $2 \cdot \max(X) \in X'$. Since the node defined in (5) computes $\{1\}$, we obtain $X' \subseteq \{1\} \cup \{2\} \cdot (X - s)$. In particular, $\max(X') = 2 \cdot \max(X)$.
The fact that the nodes defined in (6), (7), (8), and (9) compute $\{1\}$ implies $1 \in O \cap X' \cap t \cap g$ and $O = X' \cup t \cup g$. Moreover, it follows from Lemma 8 that $g = \{1, 3^m\}$ for some $m \in \mathbb{N}^+$.
Thus, as $1 \in t$,

$$O \subseteq \{1\} \cup \big(\{2\} \cdot (X - s)\big) \cup t \cup g = \big(\{2\} \cdot (X - s)\big) \cup t \cup \{3^m\}. \tag{10}$$

As O is balanced, Lemma 1 implies that $\max(O)$ is odd. Since $X \supseteq t$ and $\max(X') = 2 \cdot \max(X)$ is even, $\max(O) = 3^m > \max(X')$. Due to $(C, Q, s, t) \in \mathcal{C}$ for the given assignment it holds $|s| \geq |t|$ and that s and t do not contain any numbers with prime divisors ≤ 3. Due to that, since we have seen that $1 \in s \cap t$, and as by assumption we have $(C, Q, s, t) \notin \mathrm{CC}$ it even holds $|s| > |t|$.
Putting things together, as we have proven (10), $|s| > |t|$, $1 \in t$, $s \subseteq X$, $\max(X') = 2 \cdot \max(X)$, and $\max(O) > \max(X')$, we now obtain

$$|O| \leq \max(X) - |s| + |t| + 1 < \max(X) + 1 = \frac{\max(X') + 2}{2} \leq \frac{\max(O) + 1}{2},$$

which contradicts the fact that O is balanced. □

This completes the proof of $(\mathrm{CC}, \overline{\mathrm{CC}} \cap \mathcal{C}) \leq_{\mathrm{m}} (\mathrm{BC}'(-,\cdot), \overline{\mathrm{BC}'(-,\cdot)})$ and thus $\mathrm{BC}'(-,\cdot)$ and $\mathrm{BC}(-,\cdot)$ are $\leq_{\mathrm{m}}$-complete for RE. □

4 Smaller Sets of Operations Lead to Problems in NP

In this section it is shown that all problems $\mathrm{BC}(\mathcal{O})$ for $\mathcal{O} \subsetneq \{-,\cdot\}$ are in NP. Each of these problems is proven to be $\leq_{\mathrm{m}}^{\log}$-complete for one of the classes L, NL, and NP.

4.1 The Complexity of the Problem Solely Admitting Multiplication

This section's purpose is to prove the NL-completeness of $\mathrm{BC}(\cdot)$: first, a technical elaboration shows that $A \cdot B$ for sets A and B with sufficiently large maxima is subbalanced. Second, this result is exploited by a non-trivial non-deterministic logarithmic-space algorithm which accepts $\mathrm{BC}(\cdot)$.
In order to prove the first of the two results, we need the following estimation.

Lemma 14. *1. Let* $p_1, \ldots, p_n \in \mathbb{N}^+$ *be relatively prime. Let* A *be an interval. Define* $B_0 = A$ *and* $B_{k+1} = B_k - \{x \in A \mid p_{k+1} \mid x\}$ *for* $k = 0, \ldots, n-1$*. Then for* $k = 0, 1, \ldots, n$

$$|B_k| \geq |A| \cdot \prod_{i=1}^{k} \left(1 - \frac{1}{p_i}\right) - 2^k.$$

2. Let p be some number greater than $\max(p_1,\dots,p_n)$ and relatively prime to $\prod_{i=1}^n p_i$. Moreover, let $A = \{x \in \mathbb{N} \mid p \mid x, a \le x \le b\}$ for naturals $a \le b$. Define $B_0 = A$ and $B_{k+1} = B_k - \{x \in A \mid p_{k+1} \mid x\}$ for $k = 0,\dots,n-1$. Then for $k = 0,1,\dots,n$

$$|B_k| \ge |A| \cdot \prod_{i=1}^{k} \Big(1 - \frac{1}{p_i}\Big) - 2^k.$$

Proof. 1. We prove the inequation by induction over k. For $k = 0$ the statement is true. Assume that the statement is true for some $k \ge 0$. We prove that it also holds for $k+1$. As for arbitrary finite sets $M_1,\dots,M_n$

$$|M_1 \cup \dots \cup M_n| = \sum_{\emptyset \ne T \subseteq \{1,\dots,n\}} (-1)^{|T|-1} \Big| \bigcap_{i \in T} M_i \Big| \tag{11}$$

holds, we obtain

$$\begin{aligned}
|B_{k+1}| &= |B_k| - \Big| \{x \in A \mid p_{k+1} \mid x\} - \bigcup_{i=1}^{k} \{x \in A \mid p_i \mid x, p_{k+1} \mid x\} \Big| \\
&\overset{(11)}{\ge} |B_k| - \bigg(\Big\lceil \frac{|A|}{p_{k+1}} \Big\rceil - \sum_{\emptyset \ne T \subseteq \{1,\dots,k\}} (-1)^{|T|-1} \cdot |\{x \in A \mid \forall i \in T\ p_i \mid x, p_{k+1} \mid x\}| \bigg) \\
&\ge |B_k| - \Bigg(\Big\lceil \frac{|A|}{p_{k+1}} \Big\rceil - \Bigg(\sum_{\emptyset \ne T \subseteq \{1,\dots,k\}, |T| \text{ odd}} (-1)^{|T|-1} \left\lfloor \frac{|A|}{\left(\prod_{i \in T} p_i\right) \cdot p_{k+1}} \right\rfloor + \\
&\quad + \sum_{\emptyset \ne T \subseteq \{1,\dots,k\}, |T| \text{ even}} (-1)^{|T|-1} \left\lceil \frac{|A|}{\left(\prod_{i \in T} p_i\right) \cdot p_{k+1}} \right\rceil \Bigg) \Bigg) \\
&\ge |B_k| - 2^k - \Bigg(\frac{|A|}{p_{k+1}} - \sum_{\emptyset \ne T \subseteq \{1,\dots,k\}} (-1)^{|T|-1} \frac{|A|}{\left(\prod_{i \in T} p_i\right) \cdot p_{k+1}} \Bigg) \\
&= |B_k| - 2^k - \frac{|A|}{p_{k+1}} \cdot \Bigg(1 + \sum_{\emptyset \ne T \subseteq \{1,\dots,k\}} (-1)^{|T|} \frac{1}{\prod_{i \in T} p_i} \Bigg) \\
&= |B_k| - 2^k - \frac{|A|}{p_{k+1}} \cdot \prod_{i=1}^{k} \Big(1 - \frac{1}{p_i}\Big) \overset{\text{ind. hyp.}}{\ge} |A| \cdot \prod_{i=1}^{k} \Big(1 - \frac{1}{p_i}\Big) - \frac{|A|}{p_{k+1}} \cdot \prod_{i=1}^{k} \Big(1 - \frac{1}{p_i}\Big) - 2^{k+1} \\
&= |A| \cdot \Bigg(\prod_{i=1}^{k} \Big(1 - \frac{1}{p_i}\Big) \Bigg) \cdot \Big(1 - \frac{1}{p_{k+1}}\Big) - 2^{k+1} = |A| \cdot \Bigg(\prod_{i=1}^{k+1} \Big(1 - \frac{1}{p_i}\Big) \Bigg) - 2^{k+1}.
\end{aligned}$$

2. Note $A = \{p\} \cdot [c,d]$ for an interval $[c,d]$. Define $B'_0 = [c,d]$ and $B'_{k+1} = B'_k - \{x \in B'_0 \mid p_{k+1} \mid x\}$ for $k = 0,\dots,n-1$. As p and all p_i are relatively prime, it can be seen inductively that $B_k = \{p\} \cdot B'_k$ for all $0 \le k \le n$. In particular, $|B_k| = |B'_k|$ for $0 \le k \le n$. Applying the first statement for the B'_k finishes the proof. □

Theorem 15. *For $A, B \in \mathcal{P}_{\text{fin}}(\mathbb{N})$ with sufficiently large maxima the set $A \cdot B$ is subbalanced.*

Proof. It suffices to prove the statement for the case where $A \subseteq B$ are intervals starting from 0, i.e., $|A| = \max(A) + 1$ and $|B| = \max(B) + 1$. We even prove that $|A \cdot B| < \max(A) \cdot \max(B)/2$. Let $k = (23!)^2$. We show the statement in two steps.

1. First we show that for $|B| \in \{|A|, |A|+1, \ldots, |A|+k-1\}$ the set $A \cdot B$ has less than $\max(A) \cdot \max(B)/2$ elements.
2. Then it is argued that if $|A \cdot B| < \max(A) \cdot \max(B)/2$ for some B with $|B| \geq |A|$, then $|A \cdot (B \cup \{\max(B)+1, \max(B)+2, \ldots, \max(B)+k\})| < \max(A) \cdot (\max(B)+k)/2$.

Then, given finite and sufficiently large intervals $A = [0, a]$ and $B = [0, b]$ for $a \leq b$, if $b \leq a + k - 1$, the statement follows from 1. Otherwise, there is $0 \leq r \leq k-1$ and $s \in \mathbb{N}^+$ with $B = [0, a + r + s \cdot k]$. According to 1, the statement holds for the sets A and $B' = [0, a+r]$. Applying part 2 for s times yields that the statement also holds for A and B.

1. Let $A = [0, \alpha]$ and $B = [0, \beta]$ with $\beta \in [\alpha, \alpha + k - 1]$. We first give an approximation for the size of $A \cdot A$. Let $D = \{(a, b) \mid a, b \in A, 1 \leq a \leq b\} \cup \{(0,0)\}$ and

$$E = D - \{(a, b) \in D \mid a \text{ even}, 0 < a \leq b \leq \alpha/2\}.$$

We have $A \cdot A = \{a \cdot b \mid (a, b) \in D\} = \{a \cdot b \mid (a, b) \in E\}$: we only argue for $\subseteq$ of the second equation. Let $(a, b) \in \{(a, b) \mid a > 0, a, b \in A, a \leq b\} \notin E$. Then, as $(a, b) \in D - E$, a is even and $b \leq \alpha/2$. Consider the pair $(a/2, 2b) \in D$. If this pair is in E, we are done. Otherwise, the pair is in $D - E$ and we can apply the same argument. Thus we finally obtain a pair $(a', b') \in E$ with $a' \cdot b' = a \cdot b$.
It holds

$$|D| = \binom{\alpha}{2} + \alpha + 1 = \frac{\alpha \cdot (\alpha - 1)}{2} + \alpha + 1 = \frac{\alpha^2 + \alpha}{2} + 1$$

and if α is sufficiently large

$$|A \cdot A| \leq |E| \leq |D| - \frac{\alpha}{2 \cdot 5} \cdot \frac{\alpha}{4} - 1 \leq \frac{\frac{19}{20}\alpha^2 + \alpha}{2}.$$

Now assume $\beta = \alpha + b$ for $b \in \{1, 2, \ldots, k-1\}$. Then, with the observations made above, it holds

$$\begin{aligned} |A \cdot B| &\leq \frac{\frac{19}{20}\alpha^2 + \alpha}{2} + \alpha \cdot b = \frac{\alpha \cdot (\alpha + 1 + 2 \cdot b - \alpha/20)}{2} \\ &= \frac{\alpha \cdot \beta - \alpha \cdot \big(\alpha/20 - (b+1)\big)}{2} < \frac{\alpha \cdot \beta}{2} \end{aligned}$$

in case α is sufficiently large.

2. Let $A = [0, \alpha]$ and $B = [0, \beta]$ with $\alpha \leq \beta$ and $A \cdot B < \max(A) \cdot \max(B)/2$. We show that then $|A \cdot [0, \beta + k]| < \alpha \cdot (\beta + k)/2$.
We sketch the basic idea of the proof in a semiformal way. Consider the set $C = [1, \alpha] \times [\beta + 1, \beta + k]$. Clearly,

$$\{a \cdot b \mid (a, b) \in C\} \cup (A \cdot B) \supseteq A \cdot [0, \beta + k]. \tag{12}$$

Thus, C covers the set $(A \cdot [0, \beta + k]) - A \cdot B$. We delete the elements of two sets D and E from C. Thereto, let $P = \{p \in \mathbb{P} \mid p \leq 23\}$,

$$D = \{(a, b) \in C \mid \exists_{p \in P}\, p \mid b, 1 \leq a \leq \alpha/p\},$$

and

$$E = \{(a, b) \in C \mid \exists_{j|a} \exists_{i|b}\, j \leq \min(i-1, 50) \wedge \gcd(i, j) = 1 \wedge \frac{a \cdot i}{j} \leq \alpha\}.$$

Observe that for each pair $(a, b) \in D$ there is a prime $p \in P$ such that $(a \cdot p, b/p)$ is in C. Therefore, roughly speaking, the pairs in D are redundant and can be deleted from C.

Analogously, for each pair $(a, b) \in E$, there are numbers i and j satisfying the mentioned properties such that $(a \cdot i/j, b \cdot j/i)$ is in C. Hence, loosely speaking, also the pairs in E are obsolete and may be deleted from C.

Hence it suffices to show that $C-(D\cup E)$ does not contain too many elements. In other words, if $D \cup E$ is large enough, then the set $(A\cdot[0,\beta+k]) - A\cdot B$ is not too big and as $A\cdot B$ is subbalanced, the set $A\cdot[0,\beta+k] = A\cdot B \cup \big((A\cdot[0,\beta+k]) - A\cdot B\big)$ is subbalanced as well.

We now move to the formal proof. It is sufficient to show

$$\{a\cdot b \mid (a,b)\in C-D\}\cup(A\cdot B) = \{a\cdot b \mid (a,b)\in C\}\cup(A\cdot B), \tag{13}$$

$$\{a\cdot b \mid (a,b)\in C-(D\cup E)\}\cup A\cdot B = \{a\cdot b \mid (a,b)\in C-D\}\cup A\cdot B, \tag{14}$$

and

$$|D\cup E| \geq k\cdot\alpha/2 \tag{15}$$

because then it follows

$$\begin{aligned}
|A\cdot[0,\beta+k]| &\overset{(12)}{\leq} |\{a\cdot b \mid (a,b)\in C\}\cup(A\cdot B)| \\
&\overset{(13)}{=} |\{a\cdot b \mid (a,b)\in C-D\}\cup(A\cdot B)| \\
&\overset{(14)}{=} \{a\cdot b \mid (a,b)\in C-(D\cup E)\}\cup(A\cdot B)| \\
&\leq |C-(D\cup E)| + |A\cdot B| = |C| - |D\cup E| + |A\cdot B| \\
&\overset{(15)}{\leq} k\cdot\alpha - \frac{k\alpha}{2} + |A\cdot B| < \frac{k\alpha}{2} + \frac{\alpha\cdot\beta}{2} \\
&= \frac{\alpha\cdot(\beta+k)}{2} = \frac{\max(A)\cdot\max([0,\beta+k])}{2}.
\end{aligned}$$

Define $C' = C - D$ and $C'' = C-(D\cup E)$. We argue for (13). It suffices to prove $\supseteq$. Let $(a,b)\in C$. If $(a,b)\notin C'$, then there is a prime $p\in P$ with $p \mid b$ and $a \leq \alpha/p$. Consider the pair $(a\cdot p, b/p)$. There are three cases.

1. $(a\cdot p, b/p) \in A\times B$
2. $(a\cdot p, b/p) \in C\cap C'$
3. $(a\cdot p, b/p) \in C - C'$

In the first two cases we are done. In the third case we argue in the same way as we did for the pair (a,b). As $b/p < b$, repeating this argument finally leads to a pair which is in $A\cdot B$ or in C'.

Now we prove (14). It suffices to argue for $\supseteq$. Let $(a,b)\in C'$ and assume $(a,b)\notin C''$. Then $(a,b)\in E$. Let i and j be numbers according to the definition of E, i.e., $j \mid a$, $i \mid b$, $\gcd(a,b)=1$, $j<i$, $j\leq 50$, and $a\cdot i/j \leq \alpha$. We consider the pair $(a',b') = (a\cdot i/j, b\cdot j/i)$. We analyze all possible cases.

1. $(a',b')\in A\times B$.
2. $(a',b')\notin A\times B$. Then $b' > \max(B)$ and $(a',b')\in C$. Thus we have the following cases.

(a) $(a', b') \in C - C'$

(b) $(a', b') \in C' - C''$

(c) $(a', b') \in C' \cap C''$

In the first case and in the case 2(c) we are immediately done. In the case 2(a), according to the proof of Equation 13 there is a pair $(a'', b'') \in C' \cup A \times B$ with $a' < a''$, $b'' < b'$, and $a'' \cdot b'' = a' \cdot b'$. If $(a'', b'') \in \{(x, y) \mid x \in A, y \in B\}$, we are done. Otherwise $(a'', b'') \in C'$. Here, if $(a'', b'') \in C''$, we are done. Otherwise $(a'', b'') \in C' - C''$ and we can argue as in the case 2(b), which is the last case to consider and in which we can argue for the given pair in the same way as we did for the pair (a, b).
Whenever we consider a new pair, then this pair's first (resp., second) component is greater (resp., lower) than the first (resp., second) component of the pair before. Hence, we do not have an endless recursion and at some point in time, we will reach one of the base cases 1 and 2(c), in which we are immediately done.

For the remainder of the proof we argue for (15), i.e., we show $|D \cup E| \geq k \cdot \alpha/2$.
For $P' \subseteq P$ non-empty let $b_{P'}$ denote the least number greater than α such that each prime $p \in P - P'$ does not divide $b_{P'}$ and each prime $p \in P'$ divides $b_{P'}$ (note that due to the choice of k there always is such a number $b_{P'} \leq \alpha + k$).
Observe that in $[\beta + 1, \beta + k]$ there are

$$k \cdot \prod_{p \in P} \begin{cases} \frac{1}{p} & p \in P' \\ \frac{p-1}{p} & p \notin P' \end{cases}$$

numbers y that equal $b_{P'}$ regarding the primes in P, i.e., all $p \in P - P'$ do not divide y and all $p \in P'$ divide y. Moreover, note that a pair (a, y) for y with the properties just mentioned is in D if and only if the pair $(a, b_{P'})$ is in D. Finally $(a, b_{P'})$ is in D if and only if $1 \leq a$ and $a \cdot \min(P') \leq \alpha$.
Thus it holds

$$|D| = \sum_{\substack{P' \in \mathcal{P}(P), \\ P' \neq \emptyset}} k \cdot \left(\prod_{p \in P} \begin{cases} \frac{1}{p} & p \in P' \\ \frac{p-1}{p} & p \notin P' \end{cases} \right) \cdot \underbrace{\left| \left\{ (a, b_{P'}) \in C \mid 1 \leq a \leq \left\lfloor \frac{\alpha}{\min(P')} \right\rfloor \right\} \right|}_{\geq \frac{\alpha}{\min(P')} - 1}$$

$$\geq k \cdot \alpha \cdot \left(\sum_{\substack{P' \in \mathcal{P}(P), \\ P' \neq \emptyset}} \left(\prod_{p \in P} \begin{cases} \frac{1}{p} & p \in P' \\ \frac{p-1}{p} & p \notin P' \end{cases} \right) \cdot \left(\frac{1}{\min(P')} - \frac{1}{\alpha} \right) \right). \tag{16}$$

Now we consider lower bounds for the size of $E - D$.
Let

$$y \in \{x \mid \exists_{p \in P} p \mid x\} \cap [\beta + 1, \beta + k] =: P'$$

and $j \in \{2, \ldots, 50\}$. Moreover, let

$$i_{y,j} = \min(\{x \mid x > j, \gcd(x, j) = 1, \forall_{p \in \mathbb{P}, p | x} p \in P \wedge p^3 \nmid x \wedge x \mid y\},$$

where $\min(\emptyset)$ is defined to be -1. In other words, $i_{y,j}$ is the least number greater j and coprime to j that divides y and is solely built by primes ≤ 23 occurring with exponent at most 2 in the prime factor decomposition. We denote the least prime divisor of y by p_y. Define $E_{y,j} = \emptyset$ if $p_y \notin P$ and otherwise

$$E_{y,j} = \left\{ (x, y) \in E \mid \left\lfloor \frac{\max(A)}{p_y} \right\rfloor + 1 \leq x \leq \left\lfloor \frac{\max(A) \cdot j}{i_{y,j}} \right\rfloor, j \mid x \right\},$$

i.e., $E_{y,j}$ contains such pairs $(x, y) \in E - D$ for which j and $i_{y,j}$ witness the membership in E. Roughly speaking, as $i_{y,j}$ is selected to be as low as possible, j and $i_{y,j}$ witness the membership in E for as many pairs (x, y) as possible.
Note that by definition $E_{y,j}$ and $E_{y',j'}$ for $y' \neq y$ and arbitrary j' are disjoint.
As $E_{y,j}$ only contains pairs (a, b) with $a > \max(A)/p_y$ and D solely contains pairs (a, b) with $a \leq \max(A)/p_y$,

$$\bigcup_{y \in P'} \bigcup_{2 \leq j \leq 50} E_{y,j} \subseteq E - D.$$

It follows that $E - D$ can be written as a superset of the union of pairwise disjoint sets in the following way.

$$E - D \supseteq \bigcup_{y \in P'} \bigcup_{j=2}^{50} \underbrace{\left(E_{y,j} - \Big(\bigcup_{j'=2}^{j-1} E_{y,j'} \Big) \right)}_{=: E'_{y,j}}.$$

This shows

$$|E - D| \geq \sum_{y \in P'} \sum_{j=2,\ldots,50} |E'_{y,j}|. \tag{17}$$

The next step is to observe that, roughly speaking, the size of $E'_{y,j}$ does not really depend on y but only on the primes in P occurring in the prime factor decomposition of y:
Let y and y' satisfy the following condition: for all $p \in P$ it holds

- $p \mid y \Leftrightarrow p \mid y'$
- $p^2 \mid y \Leftrightarrow p^2 \mid y'$.

Observe that then for all $2 \leq j \leq 50$ it follows from the definitions that $i_{y,j} = i_{y',j}$ and thus $\{x \mid (x, y) \in E_{y,j}\} = \{x \mid (x, y') \in E_{y',j}\}$.
Therefore, the following definition is well-defined, i.e., independent of the choice of y. Let $P_1, P_2 \subseteq P$ with $P_1 \cup P_2 \neq \emptyset$ be disjoint. Choose $y \in \{\max(B) + 1, \ldots, \max(B) + k\}$ such that each $p \in P - (P_1 \cup P_2)$ does not divide y, for each $p \in P_1$ it holds $p \mid y$ and $p^2 \nmid y$, and for each $p \in P_2$ we have $p^2 \mid y$ (note that such a number y exists by the choice of k). Define $i_{P_1,P_2,j} := i_{y,j}$, $E_{P_1,P_2,j} := \{x \mid (x, y) \in E_{y,j}\}$, and $E'_{P_1,P_2,j} = \{x \mid (x, y) \in E'_{y,j}\}$. Note that then

$$i_{P_1,P_2,j} = \min\left(\left\{ x \mid x > j, \gcd(x, j) = 1, x \mid \Big(\prod_{p \in P_1} p \Big) \cdot \Big(\prod_{p \in P_2} p^2 \Big) \right\} \right),$$

where by definition $\min(\emptyset) = -1$, and observe that $i_{P_1,P_2,j} \leq j \cdot \max(P_1 \cup P_2)$.
Observe that in $[\beta + 1, \beta + k]$ there are

$$k \cdot \left(\prod_{p \in P} \begin{cases} \frac{p-1}{p^2} & p \in P_1 \\ \frac{1}{p^2} & p \in P_2 \\ \frac{p-1}{p} & p \notin (P_1 \cup P_2) \end{cases} \right)$$

numbers y that equal b_{P_1,P_2} regarding the primes in $P_1 \cup P_2$, i.e., all $p \in P - (P_1 \cup P_2)$ do not divide y, for all $p \in P_1$ it holds $p \mid y$ and $p^2 \nmid y$, and for all $p \in P_2$ it holds $p^2 \mid y$.
Thus, due to (17) we obtain

$$|E - D| \geq \sum_{P_1 \in \mathcal{P}(P)} \sum_{\substack{P_2 \in \mathcal{P}(P-P_1),\\ P_1 \cup P_2 \neq \emptyset}} \left[k \cdot \left(\prod_{p \in P} \begin{cases} \frac{p-1}{p^2} & p \in P_1 \\ \frac{1}{p^2} & p \in P_2 \\ \frac{p-1}{p} & p \notin (P_1 \cup P_2) \end{cases} \right) \cdot \sum_{j=2}^{50} \cdot |E'_{P_1,P_2,j}| \right]. \tag{18}$$

Now we estimate the size of the sets of the form $E'_{P_1,P_2,j}$.
The set $E_{P_1,P_2,j}$ consists of all numbers in the interval $I(E_{P_1,P_2,j}) = [\lfloor \alpha/p \rfloor + 1, \lfloor \alpha \cdot j / i_{P_1,P_2,j} \rfloor]$ dividable by j. Hence, for each $j' \in [2, 50]$ there is an interval $I(E_{P_1,P_2,j'})$ associated with it. To simplify the estimation, we do not want the intervals associated with j and j' to overlap if $j' \neq j$ and $\gcd(j, j') > 1$. So, we shrink the intervals in the following way. For that purpose fix $P_1, P_2 \subseteq P$ with $P_1 \cup P_2 \neq \emptyset$ disjoint, $j \in \{2, \ldots, 50\}$ and let $p = \min(P_1 \cup P_2)$.
Define

$$\gamma_{P_1,P_2,j} = \max\left(\{\delta_{P_1,P_2,j'} \mid 2 \leq j' < j, \gcd(j, j') \neq 1, \gamma_{P_1,P_2,j'} < \delta_{P_1,P_2,j'}\} \cup \left\{\frac{1}{p}\right\} \right),$$

$$\delta_{P_1,P_2,j} = \frac{j}{i_{P_1,P_2,j}},$$

and

$$J_{P_1,P_2,j} = \left[\lfloor \alpha \cdot \gamma_{P_1,P_2,j} \rfloor + 1, \lfloor \alpha \cdot \delta_{P_1,P_2,j} \rfloor\right].$$

Note that $\gamma_{P_1,P_2,j}$ is positive whereas $\delta_{P_1,P_2,j}$ is possibly negative and thus $i_{P_1,P_2,j} = -1$ implies $J_{P_1,P_2,j} = \emptyset$.
It follows immediately from the definition that $J_{P_1,P_2,j} \subseteq I(E_{P_1,P_2,j})$ and thus the set $E_{P_1,P_2,j}$ contains all elements in $J_{P_1,P_2,j}$ that are dividable by j. Moreover, observe that for numbers $j' < j$ with $\gcd(j', j) > 1$ the intervals $J_{P_1,P_2,j}$ and $J_{P_1,P_2,j'}$ do not overlap: this holds as $\gamma_{P_1,P_2,j} > \delta_{P_1,P_2,j'}$ by definition of $\gamma_{P_1,P_2,j}$. In other words, the interval $J_{P_1,P_2,j}$ lies above of all intervals $J_{P_1,P_2,j'}$ for $j' < j$ with $\gcd(j', j) > 1$ and if there is some number occurring in two intervals $J_{P_1,P_2,j}$ and $J_{P_1,P_2,j'}$, then j' and j are relatively prime.
For our estimation we will consider $E'_{P_1,P_2,j}$ only inside the interval $J_{P_1,P_2,j}$.
As a next step, loosely speaking, we partition the interval $J_{P_1,P_2,j}$ into a set of intervals depending on which intervals $J_{P_1,P_2,j'}$ with $j' < j$ overlap with the current part of $J_{P_1,P_2,j}$. More precisely, we define a partition of $J_{P_1,P_2,j'}$ into intervals such that for each interval I of the partition and each $j' < j$ either $J_{P_1,P_2,j'}$ contains I or the two intervals are disjoint. Then for each such I we can estimate the size of $E'_{P_1,P_2,j} \cap I$ with Lemma 14.
Note that $\gamma_{P_1,P_2,j}$ equals either $1/p$ or $\delta_{P_1,P_2,j'} + 1$ for the upper bound $\delta_{P_1,P_2,j'}$ of an interval $J_{P_1,P_2,j'}$, where $j' < j$ and $\gcd(j', j)$.
Thus the list defined in the following contains all "relevant" points. Define

$$S_{P_1,P_2,j} = \{\delta_{P_1,P_2,j'} \mid 2 \leq j' \leq j, \gamma_{P_1,P_2,j} < \delta_{P_1,P_2,j'} \leq \delta_{P_1,P_2,j}, \gamma_{P_1,P_2,j'} < \delta_{P_1,P_2,j'}\} \cup \{\gamma_{P_1,P_2,j}\}$$

and let $\Gamma_{P_1,P_2,j}$ be the empty list if $|S_{P_1,P_2,j}| = 1$ and an increasingly sorted list containing the numbers in $S_{P_1,P_2,j}$ otherwise.
Let $n_{P_1,P_2,j} \leq j$ be the number of elements in $\Gamma_{P_1,P_2,j}$.
The interval $J_{P_1,P_2,j} = \Big[\lfloor \alpha \cdot \gamma_{P_1,P_2,j} \rfloor + 1, \lfloor \alpha \cdot \delta_{P_1,P_2,j} \rfloor\Big]$ is either empty or can be partitioned into the intervals $I_{P_1,P_2,j,\sigma} = \Big[\lfloor \alpha \cdot \Gamma_{P_1,P_2,j}[\sigma] \rfloor + 1, \lfloor \alpha \cdot \Gamma_{P_1,P_2,j}[\sigma + 1] \rfloor\Big]$ for $\sigma = 1, \ldots, n_{P_1,P_2,j} - 1$, where $\Gamma_{P_1,P_2,j}[1] = \gamma_{P_1,P_2,j}$ and $\Gamma_{P_1,P_2,j}[n_{P_1,P_2,j}] = \delta_{P_1,P_2,j}$.

Observe that by construction for arbitrary $r < s$ and $1 \leq \zeta < n_{P_1,P_2,s}$ either $J_{P_1,P_2,r}$ contains $I_{P_1,P_2,s,\zeta}$ or the two intervals are disjoint.

Claim 16.

$$|E'_{P_1,P_2,j}| \geq \alpha \cdot \Bigg(\sum_{r=2}^{n_{P_1,P_2,j}} \frac{(\Gamma_{P_1,P_2,j}[r] - \Gamma_{P_1,P_2,j}[r-1])}{j} \cdot \prod_{\substack{2 \leq j' < j, \\ I_{P_1,P_2,j',r-1} \subseteq J_{P_1,P_2,j'}}} \Big(1 - \frac{1}{j'}\Big)\Bigg) - 4^j.$$

Proof of Claim 16. As P_1 and P_2 are fixed throughout this proof, we may omit corresponding indices.
Since α can be chosen so large that $\lfloor \alpha \cdot \Gamma_j[r+1] \rfloor > \lfloor \alpha \cdot \Gamma_j[r] \rfloor$, the size of the interval $I_{j,r} = [\lfloor \alpha \cdot \Gamma_j[r] \rfloor + 1, \lfloor \alpha \cdot \Gamma_j[r+1] \rfloor]$ for $1 \leq r \leq n_j - 1$ is at least

$$\lfloor \alpha \cdot \Gamma_j[r+1] \rfloor - \lfloor \alpha \cdot \Gamma_j[r] \rfloor \geq \max\big(0, \alpha \cdot (\Gamma_j[r+1] - \Gamma_j[r]) - 1\big). \tag{19}$$

As for all $r < s$ and arbitrary ζ either $J_r \supseteq I_{s,\zeta}$ or the two intervals are disjoint,

$$\begin{aligned}
E'_j &= E_j - \bigcup_{j'=2}^{j-1} E_{j'} \supseteq (E_j \cap J_j) - \bigcup_{j'=2}^{j-1} (E_{j'} \cap J_j) \\
&= \Big(E_j \cap \bigcup_{r=2}^{n_j} I_{j,r-1}\Big) - \bigcup_{j'=2}^{j-1} (E_{j'} \cap \bigcup_{r=2}^{n_j} I_{j,r-1}) \\
&= \Big(\bigcup_{r=2}^{n_j} (E_j \cap I_{j,r-1})\Big) - \bigcup_{r=2}^{n_j} \bigcup_{j'=2}^{j-1} (E_{j'} \cap I_{j,r-1}) \\
&= \Big(\bigcup_{r=2}^{n_j} (E_j \cap I_{j,r-1})\Big) - \bigcup_{r=2}^{n_j} \bigcup_{2 \leq j' < j, I_{j,r-1} \subseteq J_{j'}} (E_{j'} \cap I_{j,r-1}) \\
&\supseteq \bigcup_{r=2}^{n_j} \Big(\Big(E_j \cap I_{j,r-1}\Big) - \bigcup_{2 \leq j' < j, J_{j'} \supseteq I_{j,r-1}} (E_{j'} \cap I_{j,r-1})\Big).
\end{aligned}$$

Now Lemma 14 can be applied: each $E_j \cap I_{j,r}$ consists of those numbers in the interval $I_{j,r}$ that are dividable by j. Beginning with this set we successively remove numbers which are dividable by lower numbers j' that are relatively prime to j (cf. the choice of the bounds of the interval J_j, in particular the choice of γ_j).

$$\begin{aligned}
|E'_j| &\geq \sum_{r=2}^{n_j} |(E_j \cap I_{j,r-1})| \cdot \prod_{2\leq j'<j, I_{j,r-1}\subseteq J_{j'}} \left(1-\frac{1}{j'}\right) - n_j \cdot 2^j \\
&\overset{(19)}{\geq} \sum_{r=2}^{n_j} \left\lfloor \frac{\max(0, \alpha\cdot(\Gamma_j[r]-\Gamma_j[r-1])-1)}{j} \right\rfloor \cdot \prod_{2\leq j'<j, I_{j,r-1}\subseteq J_{j'}} \left(1-\frac{1}{j'}\right) - n_j \cdot 2^j \\
&\overset{(*)}{\geq} \sum_{r=2}^{n_j} \left\lfloor \frac{\alpha\cdot(\Gamma_j[r]-\Gamma_j[r-1])}{j} \right\rfloor \cdot \prod_{2\leq j'<j, I_{j,r-1}\subseteq J_{j'}} \left(1-\frac{1}{j'}\right) - n_j \cdot (2^j+1) \\
&\overset{(*)}{\geq} \alpha \cdot \sum_{r=2}^{n_j} \frac{(\Gamma_j[r]-\Gamma_j[r-1])}{j} \cdot \prod_{2\leq j'<j, I_{j,r-1}\subseteq J_{j'}} \left(1-\frac{1}{j'}\right) - n_j \cdot (2^j+2) \\
&\geq \alpha \cdot \sum_{r=2}^{n_j} \frac{(\Gamma_j[r]-\Gamma_j[r-1])}{j} \cdot \prod_{2\leq j'<j, I_{j,r-1}\subseteq J_{j'}} \left(1-\frac{1}{j'}\right) - 4^j,
\end{aligned}$$

where $(*)$ holds as all factors behind the Gauß-brackets are in the rational interval $[0,1]$. □

The Estimations (16) and (18) together with Claim 16 yield

$$\begin{aligned}
&|D\cup E| = |D| + |E-D| \\
&\geq k\cdot\alpha\cdot\left[\sum_{\substack{P'\in\mathcal{P}(P),\\ P'\neq\emptyset}} \left(\prod_{p\in P}\begin{cases}\frac{1}{p} & p\in P'\\ \frac{p-1}{p} & p\notin P'\end{cases}\right)\cdot\left(\frac{1}{\min(P')}-\frac{1}{\alpha}\right)+\right. \\
&\sum_{P_1\in\mathcal{P}(P)} \sum_{\substack{P_2\in\mathcal{P}(P-P_1),\\ P_1\cup P_2\neq\emptyset}} \left(\prod_{p\in P}\begin{cases}\frac{p-1}{p^2} & p\in P_1\\ \frac{1}{p^2} & p\in P_2\\ \frac{p-1}{p} & p\notin (P_1\cup P_2)\end{cases}\right)\cdot \\
&\left.\left(\sum_{j=2}^{50}\left(\sum_{r=2}^{n_{P_1,P_2,j}} \frac{(\Gamma_{P_1,P_2,j}[r]-\Gamma_{P_1,P_2,j}[r-1])}{j}\cdot\prod_{\substack{2\leq j'<j,\\ I_{P_1,P_2,j,r-1}\subseteq J_{P_1,P_2,j'}}}\left(1-\frac{1}{j'}\right)\right) - \underbrace{\frac{\sum_{\xi=2}^{50}4^\xi}{\alpha}}_{\leq\frac{4^{51}}{\alpha}}\right)\right]
\end{aligned}$$

Thus, in order to complete the proof of Theorem 15 it suffices to observe that

$$\begin{aligned}
&\sum_{\substack{P'\in\mathcal{P}(P),\\ P'\neq\emptyset}} \left(\prod_{p\in P}\begin{cases}\frac{1}{p} & p\in P'\\ \frac{p-1}{p} & p\notin P'\end{cases}\right)\cdot\frac{1}{\min(P')} + \sum_{P_1\in\mathcal{P}(P)} \sum_{\substack{P_2\in\mathcal{P}(P-P_1),\\ P_1\cup P_2\neq\emptyset}} \left(\prod_{p\in P}\begin{cases}\frac{p-1}{p^2} & p\in P_1\\ \frac{1}{p^2} & p\in P_2\\ \frac{p-1}{p} & p\notin (P_1\cup P_2)\end{cases}\right)\cdot \\
&\sum_{j=2}^{50}\left(\sum_{r=2}^{n_{P_1,P_2,j}} \frac{(\Gamma_{P_1,P_2,j}[r]-\Gamma_{P_1,P_2,j}[r-1])}{j}\cdot\prod_{\substack{2\leq j'<j,\\ I_{P_1,P_2,j,r-1}\subseteq J_{P_1,P_2,j'}}}\left(1-\frac{1}{j'}\right)\right) > \frac{1}{2}. \qquad (20)
\end{aligned}$$

Claim 17. *Inequation 20 holds.*

In order to prove Claim 17 it suffices to determine the value of the expression on the left-hand side of Inequation 20. Appendix A contains a Python program computing this value showing that it equals

$$\frac{4119837652873172969493088543163503414370874600434 69}{8180551369271618253482794367092562532109496310000 00},$$

which is greater than 1/2. □

Theorem 18. $\mathrm{BC}(\cdot) \in \mathrm{NL}$.

Proof. Consider the problem

$$\mathrm{BC}'(\cdot) \overset{df}{=} \{C \in \mathrm{BC}(\cdot) \mid \text{all gates in } C \text{ have a path to the output gate, there exists an assigned input, no assigned input computes the empty set or a set whose maximum is even or } \leq 1, \text{ there is an assignment of the unassigned inputs with values from } \mathcal{P}_{\mathrm{fin}}(\mathbb{N}^+) \text{ such that the output set is balanced.}\}.$$

Claim 19. $\mathrm{BC}(\cdot) \in \mathrm{NL}^{\mathrm{BC}'(\cdot)}$.

Proof of Claim 19. The following NL-algorithm with oracle $\mathrm{BC}'(\cdot)$ accepts $\mathrm{BC}(\cdot)$.

1. Let C be the input circuit. If C has no assigned input gates, then accept.
2. For each node, test whether there is a path from it to the output gate. If there is none, then delete the node and all its incident edges.
   This step can be implemented as a non-deterministic logarithmic-space subroutine (recall that the graph accessibility problem for directed graphs is in NL).
3. If there is some assigned input computing the empty set or a set with an even maximum, then reject.
4. If there is no assigned input computing a set with maximum > 1:
   (a) In case there is an unassigned input, accept.
   (b) In case there is no unassigned input, accept if all assigned inputs compute $\{1\}$. Otherwise reject.
5. Now there is some assigned input gate g with $\max(g) > 1$. If there is some assigned input gate computing a set containing 0, then add 0 to the set computed by g and delete 0 from all sets computed by another assigned input.
6. If there is some assigned input gate h computing the set $\{1\}$, then for each successor h'' of h do the following.
   (a) Let h' be the node such that h and h' are the predecessors of h'' (possibly it holds $h = h'$). Delete h'' and its ingoing edges and replace each outgoing edge (h'', u) with (h', u).
   (b) If h'' was the output gate, then let h' be the new output gate.

   If h has no outgoing edge, delete h. If there still is an assigned input computing $\{1\}$, repeat Step 6.
7. If the current circuit is in $\mathrm{BC}'(\cdot)$, then accept. Add 0 into the set computed by the gate g. If the current circuit is in $\mathrm{BC}'(\cdot)$, then accept, otherwise reject.

Observe that if in Step 6b a new output node is chosen, then g is the new output node: before the first execution of Step 6 g is connected to the output node. Observe that if g is connected to the output node before some execution of this step, then it still is after the execution. In particular, g is never deleted as only nodes computing $\{1\}$ are deleted in Step 6. Hence, when a new output node is chosen, g is still connected to the old output node, thus is a predecessor of it, and becomes the new output.
This shows that, in case the circuit has an output node before an execution of Step 6, then it still has afterwards.
In order to see that the algorithm is correct, it is important to note that Step 6 is no endless loop. When this step is executed for some node h the first time, then h is connected to the output gate. If h is not connected to the output node after some execution of Step 6, then it is deleted. Thus, Step 6 is only executed for nodes connected to the output node. Consequently, in each execution of it one node of the circuit is deleted. Due to that Step 6 can only be executed finitely (indeed linearly) many times.
It follows that the algorithm is indeed an NL-algorithm.

The following observations complete the proof.

- With the observations made above, it can be observed that at any point in time in which the algorithm executes one of the Steps 1 to 6 the current circuit is in $\mathrm{BC}(\cdot)$ if and only if the original circuit is in $\mathrm{BC}(\cdot)$.
- If the algorithm accepts in one of the Steps 1 to 6, then the input circuit is in $\mathrm{BC}(\cdot)$.
- If the algorithm rejects in one of the Steps 1 to 6, then the input circuit is not in $\mathrm{BC}(\cdot)$.
- When Step 7 is executed, then in the current circuit all gates have a path to the output gate, there exists an assigned input, and no assigned input computes the empty set or a set whose maximum is even or ≤ 1.
- If and only if the circuit at the beginning of the execution of Step 7 is in $\mathrm{BC}(\cdot)$, then one of the two circuits the oracle is asked in this step is in $\mathrm{BC}'(\cdot)$.

□

Due to $\mathrm{NL}^{\mathrm{NL}} = \mathrm{NL}$ it suffices to prove $\mathrm{BC}'(\cdot) \in \mathrm{NL}$.

According to Theorem 15 there is a number $\mu \in \mathbb{N}$ such that for $A, B \in \mathcal{P}_{\mathrm{fin}}(\mathbb{N}^+)$ with $\max(A) \geq \mu \leq \max(B)$ the set $A \cdot (B \cup \{0\})$ is subbalanced.
Observe that the set

$$\begin{aligned}\mathrm{GAP}' = \{((V,E), s, t, k) \mid\ & (V,E) \text{ is a directed multigraph, } s,t \in V,\ k \in \{1,2,3\},\\ & \text{there are at least } k \text{ paths from } s \text{ to } t\}\end{aligned}$$

is in NL.
We show that the following deterministic logarithmic-space algorithm with oracle GAP' accepts $\mathrm{BC}'(\cdot)$ (then due to $\mathrm{L}^{\mathrm{NL}} = \mathrm{NL}$ it follows that $\mathrm{BC}'(\cdot) \in \mathrm{NL}$). As input a $\{\cdot\}$-circuit C with the set of nodes V, set of edges E and output node g_C is given. We may assume that all gates in C have a path to the output gate, there is an assigned input a, and no assigned input computes the empty set or a set whose maximum is even or ≤ 1.

1. If there are two assigned input gates each containing an element $\geq \mu$, reject.
   If there is an assigned input gate with two paths to g_C containing an element $\geq \mu$, reject.

2. If there are at least μ assigned input gates, reject.

3. In case there is an assigned input gate with at least three paths to the output, reject.

4. We denote the set of unassigned input gates that have precisely one path (resp., two paths) to the output gate by U (resp., V).

   While there are more than μ nodes in U, there are more than μ nodes in V, or there is an unassigned input with at least three paths to the output, do the following.

   (a) If there is an unassigned input with at least three paths to the output, denote this gate by g.
   Otherwise:
      - If there are more than μ nodes in U, let g denote a node of U. Otherwise let g denote a node of V.

   (b) For each successor g'' of g do the following.
      i. Let g and g' be the predecessors of g'' (possibly $g = g'$). Delete g'' and replace each outgoing edge (g'', u) with (g', u).
      ii. If g'' was the output gate, then let g' be the new output gate.

      If g has no outgoing edges, delete g.

   Note that the sets U and V are never computed explicitly and stored on a working tape.

5. We have the following differentiation of cases.

   (a) **In case there is no assigned input gate with an element $\geq \mu$:**
   If the current circuit is in $\mathrm{BC}'(\cdot)$, then accept. Otherwise reject.

   (b) **In case there is one assigned input gate g with an element $\geq \mu$:**
      i. For each assignment for the remaining $\leq 2\mu$ unassigned input gates with values from $\mathcal{P}(\{1, \ldots, \mu\})$ do the following
         - Compute the constant-size set
         $$M = \prod_{u \in U} u \cdot \prod_{v \in V} (v \cdot v) \cdot \prod_{h \text{ is assigned input } \neq g} h.$$
         Then $g_C = M \cdot g$ where M is of constant-size and g is part of the input.
         - Test whether $g \in \mathrm{Bal}_M$ and accept in case the answer is “yes”.
      ii. Reject.

Now we consider each step explicitly. From this consideration it will follow that the algorithm works correctly and only requires logarithmic space.

1. If the algorithm rejects in Step 1, then for any assignment
$$g_C = M_1 \cdot M_2 \cdot M_3 \subseteq (M_1 \cup \{0\}) \cdot M_2 \cdot (M_3 - \{0\}),$$
where M_1 and M_2 contain an element $\geq \mu$ each. Lemma 15 yields that $|(M_1 \cup \{0\}) \cdot M_2| \leq \max(M_1) \cdot \max(M_2)/2$. Consequently, $|g_C| \leq \max(M_1) \cdot \max(M_2) \cdot \max(M_3)/2 = \max(g_C)/2$. So, $C \notin \mathrm{BC}'(\cdot)$.

2. If the algorithm rejects in Step 2, then there are μ assigned inputs which by assumption compute sets with maxima ≥ 2. These μ sets can also be seen as two sets each containing an element $\geq \mu$. Then it follows from the case before that $C \notin \mathrm{BC}'(\cdot)$.

3. Assume the algorithm rejects in Step 3 and let g be an assigned input with at least three paths to the output. Here $g_C = g^3 \cdot M \subseteq (g \cup \{0\})^3 \cdot (M - \{0\})$ for some set M depending on the assignment of the unassigned inputs. By assumption $\max(g)$ is odd and $\max(g) \geq 3$. According to Lemma 7 the set $(g \cup \{0\})^3$ has at most $\max(g)^3/2$ elements. Therefore $|g_C| \leq \max(g)^3/2 \cdot \max(M) = \max(g_C)/2$ and hence g_C is not balanced under any assignment. Therefore, $C \notin \mathrm{BC}'(\cdot)$.

4. Assigning an input with at least three paths to the output gate with a set with maximum ≥ 2 does not lead to a balanced output (cf. Lemma 1 and the argumentation for Step 3).

   Assigning μ inputs with sets with maximum ≥ 2 does not lead to a balanced output as well (cf. the argumentation for Step 2).

   Assigning any input with the empty set leads to an empty output.

   Therefore, only assignments mapping all gates g treated in Step 4b to $\{1\}$ might generate a balanced output. But then in the notation of this step, each successor of g'' computes the same set as g'. Together with the observation that, if a new output node is chosen, then a (and thus no node which might be deleted at the end of the step) becomes the new output (cf. the argumentation for Step 6 of the algorithm in Claim 19), this justifies the modifications of each execution of the loop 4b, i.e., if the circuit before some execution of this loop is in $\mathrm{BC}'(\cdot)$, then it is in $\mathrm{BC}'(\cdot)$ afterwards as well.

   Observe that each execution of this step is possible in logarithmic space and that with the same argumentation as for Step 6 of the algorithm in Claim 19 it is only executed finitely (indeed linearly) many times.

5. Due to the previous steps, at the beginning of this step the circuit has an assigned input, it only has constantly many inputs, each gate in it is connected to the output, and at most one assigned input contains an element $> \mu$.

   Therefore, in Step 5a the circuit is of constant size. Hence it remains to argue for Step 5b. Let g be an assigned input with an element greater than μ. Clearly, the algorithm only accepts if the input circuit is in $\mathrm{BC}'(\cdot)$. We argue that it only rejects if the input circuit is not in $\mathrm{BC}'(\cdot)$.

   If an unassigned input is assigned a set whose maximum is greater than μ, then we basically have the same situation as in Step 1 and an analogous argumentation yields that the output is not balanced under such an assignment. Hence, it suffices to consider those assignments which are tested by the algorithm. As the algorithm only rejects if for each such assignment and its corresponding set M it holds $g \notin \mathrm{Bal}_M$, it only rejects if the input circuit is not in $\mathrm{BC}'(\cdot)$.

   Finally observe that due to Propostion 2 Step 5a only requires logarithmic space.

□

Theorem 20. $\mathrm{BC}(\cdot)$ *is* $\leq_{\mathrm{m}}^{\log}$*-hard for* NL.

Proof. McKenzie and Wagner [MW07] proved the problem $\mathrm{MC}(\cap)$ to be $\leq_{\mathrm{m}}^{\log}$-complete for NL. The following algorithm computes a function for $\mathrm{MC}(\cap) \leq_{\mathrm{m}}^{\log} \mathrm{BC}(\cdot)$.

- input: (C, b)
- Replace all gates computing $\cap$ with gates computing $\cdot$.

- Let all input gates computing a set containing b compute the set $\{1\}$.
  Let all other input gates compute the set $\{0\}$.
- Return the modified circuit C'.

This can be done in logarithmic space. Moreover, $(C, b) \in \mathrm{MC}(\cap)$ if and only if there is an input computing a set containing b that is connected to the output node and no input computing a set not containing b is connected to the output node. In the output circuit, the output set is balanced if and only if there is an input computing the set $\{1\}$ that is connected to the output node and no input computing another set is connected to the output node. Hence $(C, b) \in \mathrm{MC}(\cap) \Leftrightarrow C' \in \mathrm{BC}(\cdot)$. □

Corollary 21. $\mathrm{BC}(\cdot)$ *is* $\leq_{\mathrm{m}}^{\log}$*-complete for* NL.

Proof. The statement follows from the Theorems 18 and 20. □

4.2 The Complexity of the Problems Not Admitting Multiplication

We consider the two remaining problems and prove that $\mathrm{BC}(-)$ is $\leq_{\mathrm{m}}^{\log}$-complete for NP and $\mathrm{BC}(\emptyset)$ is in L.

Theorem 22. $\mathrm{BC}(-)$ *is* $\leq_{\mathrm{m}}^{\log}$*-hard for* NP.

Proof. We show the hardness by a reduction from CSAT.

- input: a Boolean circuit C without assigned inputs (assigned inputs can be simulated by $X \vee \neg X$ or $X \wedge \neg X$ for an unassigned input X).
- We convert the Boolean circuit into a $\{-\}$-circuit: For each unassigned input gate X add a node $Y = \{1\} - (\{1\} - X)$ and let all outgoing edges of X start in Y.
- Let Z be an inner gate of the circuit. Replace it as follows.
  - In case Z is a $\neg$-gate with predecessor Z', let $Z = \{1\} - Z'$.
  - In case Z is a $\vee$-gate with predecessors Z_1 and Z_2, let $Z = \{1\} - \big((\{1\} - Z_1) - Z_2\big)$.
  - In case Z is a $\wedge$-gate with predecessors Z_1 and Z_2, let $Z = \big(\{1\} - (\{1\} - Z_1)\big) - (\{1\} - Z_2)$.
- Return the circuit.

Observe that this function is logarithmic-space computable. Moreover, by construction, the output circuit is in $\mathrm{BC}(-)$ if and only if the input circuit is in CSAT. □

Theorem 23. $\mathrm{BC}(-)$ *is in* NP.

Proof. We sketch an NP-algorithm that accepts $\mathrm{BC}(-)$.

1. input: a circuit C with output node g_C and labeling function α.
2. Go from g_C upwards always taking the left predecessor. Denote the input gate finally reached by g.
3. If g is assigned, then:
   guess all assignments with values from $\mathcal{P}(\alpha(g))$ and accept if the ouput set is balanced for one of these assignments, otherwise reject.

4. Note that now g is unassigned. Let M be the union of all sets computed by assigned inputs. Let $m = \max(M) + 1$. Guess an assignment such that $I(g) = \{m\}$ and each unassigned input either computes $\{m\}$ or $\emptyset$. If under this assignment g_C contains m, then accept.

5. Guess an assignment of the unassigned inputs such that each of them computes a subset of M. In case g_C is balanced, accept. Otherwise reject.

Claim 24. *If the algorithm accepts, then $C \in \mathrm{BC}(-)$.*

Proof of Claim 24. If the algorithm accepts in the 3-rd step, then $C \in \mathrm{BC}(-)$. If it accepts in the 4-th step, then there is an assignment that maps each unassigned input either to $\{m\}$ or to $\emptyset$ such that m is in the output set. Now change this assignment such that the sets mapped to $\{m\}$ are now mapped to $\{m+1, m+2, \ldots, 2m+1\}$. Then $I(C) = \{m+1, m+2, \ldots, 2m+1\}$ is balanced and $C \in \mathrm{BC}(-)$. Trivially, in case C is accepted in the 5-th step, $C \in \mathrm{BC}(-)$. $\square$

Claim 25. *If the algorithm rejects, then $C \notin \mathrm{BC}(-)$.*

Proof of Claim 25. If the algorithm rejects, then this happens in step 3 or step 5. We argue for the first case. Here g is an assigned input gate. As the output set always is a subset of the set computed by g, it holds $g_c \subseteq \alpha(g)$ for any assignment and hence it suffices to consider assignments that map all unassigned inputs to subsets of $\alpha(g)$. As the algorithm rejects, g_C is not balanced under any of these assignments and thus $C \notin \mathrm{BC}(-)$.
It remains to argue for the case where the algorithm rejects in step 5. In this case, g is an unassigned input and as step 4 did not accept, there is no assignment putting elements outside of M into the circuit's ouput set. Hence, it is sufficient to consider assignments that solely map to subsets of M. As the algorithm rejects, none of these assignments yields a balanced output set and hence there is no assignment at all under which the output set is balanced. Therefore, $C \notin \mathrm{BC}(-)$. $\square$

The observation that the algorithm can be computed in polynomial time by a non-deterministic Turing machine completes the proof. $\square$

Corollary 26. $\mathrm{BC}(-)$ *is* $\leq_{\mathrm{m}}^{\log}$*-complete for* NP.

Proof. The assertion follows from the Theorems 22 and 23. $\square$

Theorem 27. $\mathrm{BC}(\emptyset) \in \mathrm{L}$.

Proof. In this situation the output node is also an input node. Hence, the following algorithm decides $\mathrm{BC}(\emptyset)$. If the output node is an unassigned input, accept. Otherwise, test whether the set M computed by the ouput node is balanced which is possible in deterministic logarithmic space (cf. Proposition 2). Accept or reject correspondingly. $\square$

5 Conclusion and Open Questions

The following table summarizes our results, namely the lower and upper complexity bounds for the complexity of $\mathrm{BC}(\mathcal{O})$ with $\mathcal{O} \subseteq \{-, \cdot\}$. As each non-trivial problem is $\leq_{\mathrm{m}}^{\log}$-complete for L, it is not necessary to prove the mentioned lower bound for $\mathrm{BC}(\emptyset)$.

BC($\mathcal{O}$) for $\mathcal{O} =$	$\leq_m^{\log}$-hard for	contained in
$\emptyset$	L	L, Theorem 27
$\{-\}$	NP, Theorem 22	NP, Theorem 23
$\{\cdot\}$	NL, Theorem 20	NL, Theorem 18
$\{\cdot, -\}$	undecidable, Theorem 5	

To our knowledge, in contrast to all results from previous papers on complexty issues concerning decision problems for integer circuits (e.g., [MW07, Tra06, Bre07, GHR+10, GRTW10, BBD+17]) or related constraint satisfaction problems ([GJM17, Dos16]), a problem *admitting only one arithmetic operation* is shown to be undecidable. Beginning with this problem, namely BC($-, \cdot$), the problems BC($\mathcal{O}$) for $\mathcal{O} \subseteq \{-, \cdot\}$ are systematically investigated and for each of these problems the complexity is precisely characterized. It turns out that decreasing the size of the set of allowed operations yields problems that are in NP. In particular, all these problems are $\leq_m^{\log}$-complete for one of the classes L, NL, and NP.
Hence, in some sense the questions of these paper are completely answered. Nevertheless, there arise new questions from our results: Is there a set $\mathcal{O} \subseteq \{-, \cup, \cap\}$ such that BC($\mathcal{O} \cup \{+\}$) is undecidable? And if so, for which of the sets this is the case and for which it is not?

References

[BBD+17] D. Barth, M. Beck, T. Dose, C. Glaßer, L. Michler, and M. Technau. Emptiness problems for integer circuits. In *42nd International Symposium on Mathematical Foundations of Computer Science, MFCS 2017, August 21-25, 2017 - Aalborg, Denmark*, pages 33:1–33:14, 2017.

[Bre07] H.-G. Breunig. The complexity of membership problems for circuits over sets of positive numbers. In *Fundamentals of Computation Theory, 16th International Symposium, FCT 2007, Budapest, Hungary, August 27-30, 2007, Proceedings*, pages 125–136, 2007.

[Coo71] S. A. Cook. The complexity of theorem-proving procedures. In *Proceedings of the 3rd Annual ACM Symposium on Theory of Computing, May 3-5, 1971, Shaker Heights, Ohio, USA*, pages 151–158, 1971.

[Dos16] T. Dose. Complexity of constraint satisfaction problems over finite subsets of natural numbers. In *41st International Symposium on Mathematical Foundations of Computer Science, MFCS 2016, August 22-26, 2016 - Kraków, Poland*, pages 32:1–32:13, 2016.

[DPR61] M. Davis, H. Putnam, and J. Robinson. The decision problem for exponential Diophantine equations. *Annals of Mathematics*, 74(2):425–436, 1961.

[GHR+10] C. Glaßer, K. Herr, C. Reitwießner, S. D. Travers, and M. Waldherr. Equivalence problems for circuits over sets of natural numbers. *Theory Comput. Syst.*, 46(1):80–103, 2010.

[GJM17] C. Glaßer, P. Jonsson, and B. Martin. Circuit satisfiability and constraint satisfaction around skolem arithmetic. *Theor. Comput. Sci.*, 703:18–36, 2017.

[GRTW10] C. Glaßer, C. Reitwießner, S. D. Travers, and M. Waldherr. Satisfiability of algebraic circuits over sets of natural numbers. *Discrete Applied Mathematics*, 158(13):1394–1403, 2010.

[Mat70] Y. V. Matiyasevich. Enumerable sets are Diophantine. *Doklady Akad. Nauk SSSR*, 191:279–282, 1970. Translation in Soviet Math. Doklady, 11:354–357, 1970.

[MW07] P. McKenzie and K. W. Wagner. The complexity of membership problems for circuits over sets of natural numbers. *Computational Complexity*, 16(3):211–244, 2007.

[Pap94] C. M. Papadimitriou. *Computational complexity.* Addison-Wesley, Reading, Massachusetts, 1994.

[PD09] I. Pratt-Hartmann and I. Düntsch. Functions definable by arithmetic circuits. In *Conference on Mathematical Theory and Computational Practice*, volume 5635 of *Lecture Notes in Computer Science*, pages 409–418. Springer, 2009.

[SM73] 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.

[Tra06] S. D. Travers. The complexity of membership problems for circuits over sets of integers. *Theor. Comput. Sci.*, 369(1-3):211–229, 2006.

[Wag84] K. Wagner. The complexity of problems concerning graphs with regularities (extended abstract). In *Proceedings of the Mathematical Foundations of Computer Science 1984*, pages 544–552, London, UK, UK, 1984. Springer-Verlag.

[Yan01] K. Yang. Integer circuit evaluation is PSPACE-complete. *Journal of Computer and System Sciences*, 63(2):288–303, 2001. An extended abstract of appeared at CCC 2000.

A A Program for Claim 17

Claim 17 states

$$\sum_{\substack{P'\in\mathcal{P}(P),\\ P'\neq\emptyset}} \left(\prod_{p\in P}\begin{cases}\frac{1}{p} & p\in P'\\ \frac{p-1}{p} & p\notin P'\end{cases}\right)\cdot\frac{1}{\min(P')}+\sum_{P_1\in\mathcal{P}(P)}\sum_{\substack{P_2\in\mathcal{P}(P-P_1),\\ P_1\cup P_2\neq\emptyset}}\left(\prod_{p\in P}\begin{cases}\frac{p-1}{p^2} & p\in P_1\\ \frac{1}{p^2} & p\in P_2\\ \frac{p-1}{p} & p\notin(P_1\cup P_2)\end{cases}\right)\cdot$$
$$\sum_{j=2}^{50}\left(\sum_{r=2}^{n_{P_1,P_2,j}}\frac{(\Gamma_{P_1,P_2,j}[r]-\Gamma_{P_1,P_2,j}[r-1])}{j}\cdot\prod_{\substack{2\leq j'<j,\\ I_{P_1,P_2,j,r-1}\subseteq J_{P_1,P_2,j'}}}\left(1-\frac{1}{j'}\right)\right)>\frac{1}{2}.$$

We divide the left-hand side into two parts and at the same time introduce some abbreviations that will be used by the program. Note that by definition $I_{P_1,P_2,j,r-1}\subseteq J_{P_1,P_2,j'}$ if and only if $\gamma_{P_1,P_2,j'}\leq\Gamma_{P_1,P_2,j}[r-1]\wedge\Gamma_{P_1,P_2,j}[r]\leq\delta_{P_1,P_2,j'}$.

$$\sum_{\substack{P'\in\mathcal{P}(P),\\ P'\neq\emptyset}} \left(\prod_{p\in P}\begin{cases}\frac{1}{p} & p\in P'\\ \frac{p-1}{p} & p\notin P'\end{cases}\right)\cdot\frac{1}{\min(P')}+ \tag{21}$$

$$\sum_{P_1\in\mathcal{P}(P)}\sum_{\substack{P_2\in\mathcal{P}(P-P_1)\\ P_1\cup P_2\neq\emptyset}}\underbrace{\left(\prod_{p\in P}\begin{cases}\frac{p-1}{p^2} & p\in P_1\\ \frac{1}{p^2} & p\in P_2\\ \frac{p-1}{p} & p\notin(P_1\cup P_2)\end{cases}\right)}_{=:W_{P_1,P_2}}\cdot$$
$$\underbrace{\left[\underbrace{\sum_{j=2}^{50}\underbrace{\sum_{r=2}^{n_{P_1,P_2,j}}\underbrace{\frac{(\Gamma_{P_1,P_2,j}[r]-\Gamma_{P_1,P_2,j}[r-1])}{j}\cdot\prod_{\substack{2\leq j'<j,\\ \gamma_{P_1,P_2,j'}\leq\Gamma_{P_1,P_2,j}[r-1],\\ \Gamma_{P_1,P_2,j}[r]\leq\delta_{P_1,P_2,j'}}}\left(1-\frac{1}{j'}\right)}_{=:X_{P1,P2,j,r}}}_{=:Y_{P1,P2,j}}}\right]}_{Z_{P1,P2}} \tag{22}$$

In the following a Python program is given that computes the values of the Expressions 21 and 22. The values are

$$\frac{16371319996435847}{49770428644836900}$$

and

$$\frac{142895417326061807270595835531719117019879078513469}{818055136927161825348279436709256253210949631000000}.$$

The program represents rational numbers as 2-tuples of integers, where the first entry represents the numerator and the second represents the denominator. It never occurs the situation that the numerator is 0. Moreover, we assume that functions for the powerset of a set, sorting a list of rationals, the greatest common divisor of two integers, as well as multiplication, addition, subtraction, and the relations $=$, $<$, $>$, $\leq$, and $\geq$ over the rationals are given. The names of the corresponding functions are supposed to be “powerset, sort, gcd, mul, add, sub, eq, lower, greater, leq, geq”.
The program follows the Expressions 21 and 22 and broadly uses the same notation.

```python
def computeSetOfPairsP1P2(P):
    pP = powerset(P)
    S = set()
    for P1 in pP:
        S |= {(P1, P2) for P2 in powerset(P - P1)}
    S -= {(frozenset(),frozenset())}
    return S

def computeW_P1P2(P,P1,P2):
    ret = (1,1)
    for p in P:
        if p in P1:
            ret = mul(ret, (p-1, p**2))
        elif p in P2:
            ret = mul(ret, (1, p**2))
        else:
            ret = mul(ret, (p-1, p))
    return ret

def compute__i_P1P2j(j, P1, P2):
    for x in range(j + 1, max(P1|P2)*j+1):
    #recall i_P1P2j <= j * max(P1|P2)
        if gcd(x,j) == 1:
            k = x
            for p in P1|P2:
                if k%p == 0:
                    k = k//p
                    if k%p == 0 and p in P2:
                        k = k//p
            if k == 1:
                return x
    return -1

def compute__delta_P1P2(j,i):
    delta_P1P2 = {}
    for k in range(2, j + 1):
        delta_P1P2[k] = (k, i[k])
    return delta_P1P2

def compute__gamma_P1P2(j,i,delta_P1P2,pmin):
    gamma_P1P2 = {}
    for k in range(2,j+1):
        m = (1, pmin)
        for j_ in range(2, k):
            if gcd(k,j_) > 1 and lower(gamma_P1P2[j_], delta_P1P2[j_]):
                if greater(delta_P1P2[j_], m):
                    m = delta_P1P2[j_]
        gamma_P1P2[k] = m
    return gamma_P1P2
```

```python

def computeGamma_P1P2j(gamma_P1P2, delta_P1P2, j):
    if geq(gamma_P1P2[j], delta_P1P2[j]):
        return []
    Gamma = {gamma_P1P2[j]}
    for j_ in range(2, j + 1):
        d = delta_P1P2[j_]
        if lower(gamma_P1P2[j],d) and leq(d,delta_P1P2[j]):
            if lower(gamma_P1P2[j_], d):
                Gamma |= {d}
    Gamma = {(a//gcd(a,b), b//gcd(a,b)) for (a,b) in Gamma}
    return sort(list(Gamma))

def computeY_P1P2j(j, Gamma_P1P2j, gamma_P1P2, delta_P1P2):
    Y_P1P2j = (0,1)
    for r in range(1, len(Gamma_P1P2j)):
        X_P1P2jr = mul(sub(Gamma_P1P2j[r],Gamma_P1P2j[r-1]), (1,j))
        for j_ in range(2, j):
            if leq(gamma_P1P2[j_], Gamma_P1P2j[r-1]):
                if leq(Gamma_P1P2j[r], delta_P1P2[j_]):
                    X_P1P2jr = mul(X_P1P2jr, (j_-1, j_))
        Y_P1P2j = add(X_P1P2jr, Y_P1P2j)
    return Y_P1P2j

def computeValueOfExpression21():
    P = [2,3,5,7,11,13,17,19,23]
    Q = powerset(set(P)) - {frozenset()}
    ret = (0, 1)
    for P_ in Q:
        f = (1,1)
        for p in P:
                if p in P_:
                   f = mul(f, (1,p))
                else:
                   f = mul(f, (p-1,p))
        ret = add(ret, mul(f, (1, min(P_))))
    return ret

def computeValueOfExpression22(jlim = 50):
  P = [2,3,5,7,11,13,17,19,23]
  Q = computeSetOfPairsP1P2(set(P))
  ret = (0, 1)
  for (P1, P2) in Q:
    W_P1P2 = computeW_P1P2(P, P1, P2)
    i = [0 for i in range(0, jlim + 1)]
    Z_P1P2 = (0,1)
    for j in range(2, jlim + 1):
      i[j] = compute__i_P1P2j(j, P1, P2)
      delta_P1P2 = compute__delta_P1P2(j,i)
```

```
            gamma_P1P2 = compute__gamma_P1P2(j,i,delta_P1P2,min(P1|P2))
            Gamma_P1P2j = computeGamma_P1P2j(gamma_P1P2,delta_P1P2,j)
            Y_P1P2j = computeY_P1P2j(j,Gamma_P1P2j,gamma_P1P2,delta_P1P2)
            Z_P1P2 = add(Z_P1P2, Y_P1P2j)
        ret = add(ret, mul(W_P1P2,Z_P1P2))
    return ret
```