Functions Definable by Arithmetic Circuits

Ian Pratt-Hartmann[1] and Ivo Düntsch[2]

[1] School of Computer Science, University of Manchester, Manchester M13 9PL, U.K.
ipratt@cs.man.ac.uk
[2] Department of Computer Science, Brock University, St. Catherines, ON, L2S 3A1, Canada. duentsch@brocku.ca

Abstract. An *arithmetic circuit* (McKenzie and Wagner [6]) is a labelled, directed graph specifying a cascade of arithmetic and logical operations to be performed on sets of non-negative integers. In this paper, we consider the definability of functions by means of arithmetic circuits. We prove two negative results: the first shows, roughly, that a function is not circuit-definable if it has an infinite range and sub-linear growth; the second shows, roughly, that a function is not circuit-definable if it has a finite range and fails to converge on certain 'sparse' chains under inclusion. We observe that various functions of interest fall under these descriptions.

Keywords. Arithmetic circuit, integer expression, complex algebra, expressive power.

1 Introduction

An *arithmetic circuit* (McKenzie and Wagner [6]) is a labelled, directed graph specifying a cascade of arithmetic and logical operations to be performed on sets of non-negative integers. (Henceforth, we refer to non-negative integers simply as *numbers*). Each node in this graph evaluates to a set of numbers, representing a stage of the computation performed by the circuit. Nodes without predecessors in the graph are called *input nodes*, and their labels indicate the sets of numbers to which they evaluate. Nodes with predecessors in the graph are called *arithmetic gates*, and their labels indicate operations to be performed on the values of their immediate predecessors; the results of these operations are then taken to be the values of the arithmetic gates in question. Multiple edges are allowed. One of the nodes in the graph (usually, a node with no successors) is designated as the *circuit output*; the set of numbers to which it evaluates is taken to be the value of the circuit as a whole.

We allow input nodes to be labelled with any of the symbols $\{1\}$, $\{0\}$, $\emptyset$ or $\mathbb{N}$, denoting a set of numbers in the conventional way, or, alternatively, with a variable ranging over sets of numbers. Similarly, we allow arithmetic gates to be labelled with any of the symbols $+$, $\bullet$, $^-$, $\cap$ or $\cup$, denoting an operation on sets. The symbols $^-$, $\cap$, $\cup$ have the obvious Boolean interpretations (with $^-$ denoting

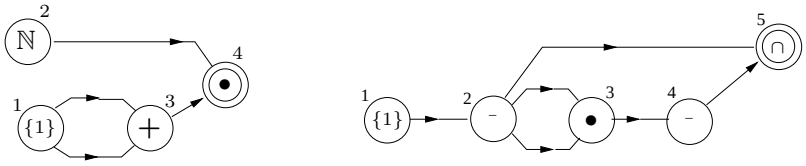
complementation in $\mathbb{N}$), while $+$ and $\bullet$ denote the result of lifting addition and multiplication to the algebra of sets, thus:

$$s + t = \{i + j | i \in s \text{ and } j \in t\}; \qquad s \bullet t = \{i \cdot j | i \in s \text{ and } j \in t\}.$$

We assume that arithmetic gates have the appropriate number of immediate predecessors (1 or 2) for their labels.

Fig. 1 shows two examples of arithmetic circuits, where the output gate is indicated by the double circle. In Fig. 1a) Node 1 evaluates to $\{1\}$, and Node 2 to $\mathbb{N}$; hence, Node 3 evaluates to $\{1\} + \{1\} = \{2\}$, and Node 4, the output of the circuit, to $\{2\} \bullet \mathbb{N}$, i.e. the set of even numbers. The circuit of Fig. 1b) functions similarly: Node 2 evaluates to $\{0\} \cup \{n \in \mathbb{N} \mid n \geq 2\}$, and Node 3 to $\{0\} \cup \{n \in \mathbb{N} \mid n \text{ is composite}\}$; hence, Node 4 evaluates to the set numbers which are either prime or equal to 1, and Node 5, the output of the circuit, to the set of primes. We say that the circuits of Fig. 1a) and b) *define*, respectively, the set of even numbers and the set of primes. Any arithmetic circuit with no variable inputs defines a set of numbers in this way. Likewise, any arithmetic circuit with one or more variable inputs defines a function from (tuples of) sets of numbers to sets of numbers. The question naturally arises as to which sets and functions are definable by arithmetic circuits.

Fig. 1. Arithmetic circuits defining: a) the set of even numbers; b) the set of primes. The integers next to the nodes are for reference only.



The study of arithmetic circuits originates in Stockmeyer and Meyer [7], who studied *integer expressions*—in effect, arithmetic circuits with no variables and no multiplication gates. (Integer expressions are essentially the same as star-free regular expressions over a 1-element alphabet, where the integer n stands for the string of length n). The *membership* problem for integer expressions is as follows: given a number and an integer expression, determine whether that number is in the set defined by that integer expression. The *non-emptiness problem* is as follows: given an integer expression, determine whether the set of numbers it defines is non-empty. Stockmeyer and Meyer showed that both these problems are PSPACE-complete.

The membership and non-emptiness problems for variable-free arithmetic circuits—i.e., circuits featuring, additionally, the $\bullet$-operator, but with no variable inputs—are discussed by McKenzie and Wagner [6]. It is easy to see that these problems are reducible to one another; however, the question of their decidability

is currently still open. The complexity of the membership problem (and related problems) for variable-free arithmetic circuits with operators chosen from various proper subsets of $\{\cup, \cap, ^-, +, \bullet\}$ are studied in Glaßer *et al.* [1, 2], building on the work of Meyer and Stockmeyer, *op. cit.*, McKenzie and Wagner, *op. cit.* and Yang [9]. A somewhat different complexity-theoretic profile is obtained if arithmetic circuits are taken to compute over sets of integers (possibly negative), as shown in Travers [8].

Arithmetic circuits with variable inputs give rise to different complexity-theoretic problems. Jeż and Okhotin [5] consider systems of equations $\phi_i(\boldsymbol{x}) = \psi_i(\boldsymbol{x})$ $(1 \leq i \leq n)$ where the ϕ_i and ψ_i are arithmetic circuits with a tuple of variable inputs $\boldsymbol{x}$, but without the multiplication or complementation operators. It is shown, *inter alia*, that all recursively enumerable sets are representable as the least solutions of such systems of equations. Further, it is shown in Jeż and Okhotin [4] that, when consideration is restricted to so-called resolved sets of equations (i.e. ϕ_i is the variable x_i, where $\boldsymbol{x} = x_1, \ldots, x_n$), the family of sets representable as least solutions of such equations is included in ExpTime, and moreover contains some ExpTime-hard sets.

In the present paper, we consider the expressive power of arithmetic circuits. In particular, we ask: which functions (from tuples of sets of numbers to sets of numbers) are definable by arithmetic circuits? We obtain two negative results: the first shows, roughly, that a function is not circuit-definable if it has an infinite range and sub-linear growth; the second shows, roughly, that a function is not circuit-definable if it has a finite range and fails to converge on certain 'sparse' chains under inclusion. We observe that various functions of interest fall under these descriptions.

2 Preliminaries

The *full complex algebra* of $\mathbb{N}$ is the structure $\mathfrak{Cm} = \{2^{\mathbb{N}}, \cup, ^-, \emptyset, \mathbb{N}, +, \{0\}, \bullet, \{1\}\}$, with $+$ and $\bullet$ as defined above. Let V be an infinite set of *variables*. A *complex arithmetic term* (or simply *term*) is the smallest set C of expressions satisfying the conditions: (*i*) $V \subseteq C$; (*ii*) the constants $\{0\}$, $\{1\}$, $\emptyset$ and $\mathbb{N}$ are in C; and (*iii*) if σ, τ are in C, then so are $\overline{\sigma}$, $\sigma \cap \tau$, $\sigma \cup \tau$ $\sigma + \tau$, $\sigma \bullet \tau$. If $\boldsymbol{x} = x_1, \ldots, x_n$ is a non-empty tuple of variables, and τ a term all of whose variables are among the $\boldsymbol{x}$, we optionally write τ as $\tau(\boldsymbol{x})$ to denote the order of variables. The algebra of terms $\tau(\boldsymbol{x})$ is denoted $C[\boldsymbol{x}]$. Likewise, the algebra of variable-free terms is denoted $C[\,]$.

An *interpretation* is a function $\iota : V \to 2^{\mathbb{N}}$ mapping variables to sets of numbers. For any tuple of variables $\boldsymbol{x}$ (possibly empty), ι is extended homomorphically to a function $C[\boldsymbol{x}] \to \mathfrak{Cm}$. For the sake of readability, if τ is a term in $C[\boldsymbol{x}]$, and ι an interpretation mapping the tuple of variables $\boldsymbol{x}$ to the tuple of sets of numbers $\boldsymbol{s}$, then we denote $\iota(\tau)$ by $\tau(\boldsymbol{s})$. In particular, if τ is variable-free, the set $\iota(\tau)$ (which is independent of ι) is denoted by $\tau(\,)$. If $\boldsymbol{x}$ is an n-tuple of variables $(n > 0)$, the function *defined by* a term $\tau \in C[\boldsymbol{x}]$ is the function $\boldsymbol{s} \mapsto \tau(\boldsymbol{s})$. Any function $f : (2^{\mathbb{N}})^n \to 2^{\mathbb{N}}$ which can be written in this way is said

to be *term-definable*. Likewise, if τ is a variable-free term, the set *defined by* τ is the set of numbers $\tau(\,)$. Any set $s \subseteq \mathbb{N}$ which can be written in this way is said to be *term-definable*.

For the purposes of investigating term-definability, there is no difference between complex arithmetic terms and arithmetic circuits, as described in Section 1. Thus, for example, the circuits of Fig. 1a) and b) correspond to the respective terms

$$\tau_e = (\{1\} + \{1\}) \bullet \mathbb{N} \qquad\qquad \tau_p = \overline{\{1\}} \cap \overline{(\overline{\{1\}} \bullet \overline{\{1\}})}. \tag{1}$$

For the sake of familiarity, therefore, we speak of *circuits* in preference to *terms*, referring to term-definable functions as *circuit-definable* functions, and similarly for term-definable sets.

3 Circuit-definable sets

We observed in Section 1 that it is not known whether the membership problem for arithmetic circuits is decidable. However, for any *fixed* arithmetic circuit $\tau \in C[\,]$, the problem of determining membership in the set $\tau(\,)$ is easily seen to be decidable, and in fact to have relatively low computational complexity.

We introduce a device which will prove useful when we come to examine circuit-definable functions below. Define the binary operation $\circ$ on sets of numbers by

$$s \circ t = \{i \cdot j \mid i \in s \setminus \{0\}, j \in t \setminus \{0\}\}.$$

Thus, $\circ$ is a modified form of set multiplication. Let us define the the set of *modified circuits* C° in exactly the same way as C, except that the multiplication operator $\bullet$ is replaced by the modified multiplication operator $\circ$.

If $\boldsymbol{x}$ is a tuple of variables (possibly empty), the algebra of modified circuits $\tau(\boldsymbol{x})$ will be denoted $C^\circ[\boldsymbol{x}]$. The concept of a circuit's *defining* a function or set is carried over to modified circuits in the obvious way.

Theorem 1. *A set is definable by an arithmetic circuit if and only if it is definable by a modified circuit.*

Proof. Define $g : C[\,] \to C^\circ[\,]$ recursively as follows: (*i*) if $\tau \in \{\{0\}, \{1\}, \emptyset, \mathbb{N}\}$, then $g(\tau) = \tau$; (*ii*) if $\star \in \{\cap, \cup, +\}$, then $g(\sigma \star \tau) = g(\sigma) \star g(\tau)$; (*iii*)

$$g(\sigma \bullet \tau) = \begin{cases} (g(\sigma) \circ g(\tau)) \cup \{0\} & \text{if } 0 \in \tau(\,) \text{ and } \sigma(\,) \neq \emptyset \\ (g(\sigma) \circ g(\tau)) \cup \{0\} & \text{if } 0 \in \sigma(\,) \text{ and } \tau(\,) \neq \emptyset \\ g(\sigma) \circ g(\tau) & \text{otherwise.} \end{cases}$$

Routine induction shows that, for all $\tau \in C[\,]$, $\tau(\,) = g(\tau)(\,)$. Hence, every set definable by an arithmetic circuit is definable by a modified circuit. The converse is proved similarly.

We remark that, since the conditions $0 \in \tau(\ )$ or $\tau(\ ) = \emptyset$ are not known to be decidable, g is not known to be computable. That is: we know that every circuit-definable set is definable by some modified circuit or other, but we do not know which one.

If m is a number, and s, t sets of numbers, to determine whether $m \in s \circ t$, it suffices check whether m can be factored as $m = m_1 \cdot m_2$, *with* m_1, m_2 *bounded by* m, such that $m_1 \in s$ and $m_2 \in t$. (Notice that this is not true for $s \bullet t$.) We remind the reader that the *polynomial hierarchy*, PH, is defined as follows:

$$\Sigma^p_0 = \mathrm{P}\textsc{Time}; \qquad \Sigma^p_{k+1} = \mathrm{NP}\textsc{Time}^{\Sigma^p_k} \quad (0 \leq k); \qquad \mathrm{PH} = \bigcup_{k \in \mathbb{N}} \Sigma^p_k.$$

It is not known whether this hierarchy is strict. We have $\mathrm{PH} \subseteq \mathrm{P}\textsc{Space}$; moreover, if $\mathrm{PH} = \mathrm{P}\textsc{Space}$, then the polynomial hierarchy collapses at some point.

Corollary 1. *All circuit-definable sets are in the polynomial hierarchy. In particular, unless the polynomial hierarchy collapses, no* PSPACE*-complete set is circuit-definable.*

Proof. By Theorem 1, it suffices to show that all sets definable by circuits in $C^0[\ ]$ are in PH. Using a simple induction, every set definable by a modified circuit is evidently definable by a formula of *bounded arithmetic* (see e.g. Harrow [3]), and hence is in PH.

4 Circuit-definable functions

Many natural functions involving sets of numbers turn out to be circuit-definable. For example, the function

$$d(x) = \{n \in \mathbb{N} \mid \forall m \in x, n < m\} \tag{2}$$

is defined by the circuit $\tau_d(x) = \overline{x + \mathbb{N}}$ (but cf. Corollary 4 below). Likewise, the function

$$\mu(x) = \begin{cases} \{\min(x)\} & \text{if } x \neq \emptyset \\ \emptyset & \text{otherwise.} \end{cases}$$

is defined by the circuit $\tau_\mu(x) = (\tau_d(x + \{1\})) \cap x$ (but cf. Corollary 3 below).

Boolean-valued functions may be defined by arithmetic circuits, employing some encoding of truth values as sets of numbers. The precise encoding chosen is not important: we write $\top$ for $\{0\}$ and $\bot$ for $\emptyset$. For example, the Boolean-valued function

$$S(x) = \begin{cases} \top & \text{if } x \text{ is a singleton} \\ \bot & \text{otherwise} \end{cases} \tag{3}$$

is defined by the circuit $\tau_S(x) = (x \bullet \{0\}) \cap \overline{((x \cap \overline{\tau_\mu(x)}) \bullet \{0\})}$.

Definition by cases is also possible: if the functions $F, G, H : (2^{\mathbb{N}})^n \to 2^{\mathbb{N}}$ are defined by the circuits $\rho(\boldsymbol{x})$, $\sigma(\boldsymbol{x})$, $\tau(\boldsymbol{x})$, respectively, then the function

$$\boldsymbol{x} \mapsto \begin{cases} G(\boldsymbol{x}) & \text{if } F(\boldsymbol{x}) = \emptyset \\ H(\boldsymbol{x}) & \text{otherwise} \end{cases}$$

is defined by the circuit $(((\rho(\boldsymbol{x}) \bullet \{0\}) + \mathbb{N}) \cap \sigma(\boldsymbol{x})) \cup (\overline{((\rho(\boldsymbol{x}) \bullet \{0\}) + \mathbb{N})} \cap \tau(\boldsymbol{x}))$.

If $f : \mathbb{N}^n \to \mathbb{N}$ is a function from tuples of numbers to numbers, we say that f is *circuit-definable* if there exists a circuit-definable function $F : (2^{\mathbb{N}})^n \to 2^{\mathbb{N}}$ such that, for all $m_1, \ldots, m_n$, $F(\{m_1\}, \ldots, \{m_n\}) = \{f(m_1, \ldots, m_n)\}$. For example, given a fixed number $p > 1$, the function $n \mapsto (n \mod p)$ is defined, in this sense, by the circuit

$$\tau_{\mathrm{mod}\ p}(x) = \bigcup_{0 \leq k < p} \big(((x \cap (((\{p\} \bullet \mathbb{N}) + \{k\})) \bullet \{0\}) + \{k\}\big)$$

where, for $m > 1$, $\{m\}$ abbreviates $\{1\} + \cdots + \{1\}$ (m times). Notice that, when discussing the circuit-definability of a function $f : \mathbb{N}^n \to \mathbb{N}$, we do not care what values the defining circuit takes on non-singleton inputs.

We now proceed to establish some simple results on functions which are not circuit-definable. We employ the following notation. If $s \subseteq \mathbb{N}$ and $m \in \mathbb{N}$, denote by $s_{|m}$ the set $s \cap [0, m]$. If $\boldsymbol{s} = s_1, \ldots, s_n$ is a tuple of elements of $2^{\mathbb{N}}$, write $\boldsymbol{s}_{|m}$ for $(s_1)_{|m}, \ldots, (s_n)_{|m}$.

Lemma 1. *Let $\tau \in C^\circ[\boldsymbol{x}]$ with $\boldsymbol{x}$ of arity n, let $m \in \mathbb{N}$, and let $\boldsymbol{s}, \boldsymbol{t} \in (2^{\mathbb{N}})^n$. If $\boldsymbol{s}_{|m} = \boldsymbol{t}_{|m}$, then, $\tau(\boldsymbol{s})_{|m} = \tau(\boldsymbol{t})_{|m}$.*

Proof. Straightforward induction on the structure of $\tau(\boldsymbol{x})$.

Note that Lemma 1 fails if the condition $\tau \in C^\circ[\boldsymbol{x}]$ is replaced by $\tau \in C[\boldsymbol{x}]$.

Let $\boldsymbol{x}$ be an n-tuple of variables ($n \geq 0$). A *circuit-condition* (*in* $\boldsymbol{x}$) is an expression α of the form

$$\bigwedge_{1 \leq i \leq a} \sigma_i(\boldsymbol{x}) = \emptyset \wedge \bigwedge_{1 \leq i \leq b} \tau_i(\boldsymbol{x}) \neq \emptyset \wedge \bigwedge_{1 \leq i \leq c} 0 \in \upsilon_i(\boldsymbol{x}) \wedge \bigwedge_{1 \leq i \leq d} 0 \notin \varphi_i(\boldsymbol{x}), \qquad (4)$$

where $a, b, c, d \geq 0$, and the σ_i, τ_i, υ_i and φ_i are circuits in $C[\boldsymbol{x}]$. (The empty conjunct with $a = b = c = d = 0$ is allowed.) When writing circuit-conditions, we silently re-order conjuncts and remove duplicates as necessary. If we wish to specify a particular order of the variables in α, we write $\alpha(\boldsymbol{x})$. If $\boldsymbol{s} \in (2^{\mathbb{N}})^n$, we take $\alpha(\boldsymbol{s})$ to be the Boolean value (either $\top$ or $\bot$) computed by substituting $\boldsymbol{s}$ for $\boldsymbol{x}$ in (4), in the obvious way. Two circuit-conditions $\alpha(\boldsymbol{x})$ and $\alpha'(\boldsymbol{x})$ are *disjoint* if, for all $\boldsymbol{s}$, one of $\alpha(\boldsymbol{s})$ or $\alpha'(\boldsymbol{s})$ is $\bot$, and a k-tuple of circuit-conditions $\alpha_1(\boldsymbol{x}), \ldots, \alpha_k(\boldsymbol{x})$ is *jointly exhaustive* if, for all $\boldsymbol{s}$, one of $\alpha_1(\boldsymbol{s}), \ldots, \alpha_k(\boldsymbol{s})$ is $\top$.

A *circuit-clause* (*in* $\boldsymbol{x}$) is an expression γ of the form $\alpha \to \tau$, where α is a circuit-condition in $\boldsymbol{x}$ and τ a circuit in $\boldsymbol{x}$; in this case, α is called the *condition of* γ. A *circuit-ensemble* (in $\boldsymbol{x}$) is a finite set $\varepsilon = \{\gamma_1, \ldots, \gamma_k\}$ of circuit-clauses

in $\boldsymbol{x}$ such that the respective conditions $\alpha_1(\boldsymbol{x}), \ldots, \alpha_k(\boldsymbol{x})$ are pairwise disjoint and jointly exhaustive. Again, we write $\gamma(\boldsymbol{x})$, $\varepsilon(\boldsymbol{x})$ *etc.* to indicate the order of the variables; and if $\boldsymbol{s}$ is an n-tuple of elements of $2^{\mathbb{N}}$, we take $\varepsilon(\boldsymbol{s})$ to be the set $\tau(\boldsymbol{s})$, where $\alpha \to \tau$ is the unique clause of ε such that $\alpha(\boldsymbol{s}) = \top$. If all the circuits mentioned in ε (including those occurring in the conditions of the clauses), are modified circuits (i.e. are in $C^\circ[\boldsymbol{x}]$), then we say that ε is a *modified* circuit-ensemble. We denote the set of all modified circuit-ensembles in $\boldsymbol{x}$ by $E^\circ[\boldsymbol{x}]$. Let $\tau(\boldsymbol{x})$ be a circuit and $\varepsilon(\boldsymbol{x})$ a circuit-ensemble: we say $\tau(\boldsymbol{x})$ is *equivalent* to $\varepsilon(\boldsymbol{x})$ if, for all $\boldsymbol{s}$, $\tau(\boldsymbol{s}) = \varepsilon(\boldsymbol{s})$.

Lemma 2. *For every $\tau \in C[\boldsymbol{x}]$, there exists an $\varepsilon \in E^\circ[\boldsymbol{x}]$ such that $\tau(\boldsymbol{x})$ is equivalent to $\varepsilon(\boldsymbol{x})$.*

Proof. We construct an equivalent modified circuit-ensemble ε by induction on the structure of τ. We illustrate with the case $\sigma(\boldsymbol{x}) \bullet \tau(\boldsymbol{x})$. Suppose that $\sigma(\boldsymbol{x})$ is equivalent to a modified circuit-ensemble $\{\alpha_i \to \sigma_i \mid 1 \leq i \leq l\}$ and $\tau(\boldsymbol{x})$ is equivalent to a modified circuit-ensemble $\{\beta_j \to \tau_j \mid 1 \leq j \leq m\}$. Then it is easy to see that the set of clauses of the forms

$$\begin{aligned}
&\alpha_i \wedge \beta_j \wedge \sigma_i \neq \emptyset \wedge 0 \in \tau_j \to (\sigma_i \circ \tau_j) \cup \{0\} \\
&\alpha_i \wedge \beta_j \wedge 0 \in \sigma_i \wedge \tau_j \neq \emptyset \wedge 0 \notin \tau_j \to (\sigma_i \circ \tau_j) \cup \{0\} \\
&\alpha_i \wedge \beta_j \wedge \sigma_i \neq \emptyset \wedge 0 \notin \sigma_i \wedge \tau_j \neq \emptyset \wedge 0 \notin \tau_j \to \sigma_i \circ \tau_j \\
&\alpha_i \wedge \beta_j \wedge \sigma_i = \emptyset \wedge \tau_j \neq \emptyset \to \emptyset \\
&\alpha_i \wedge \beta_j \wedge \tau_j = \emptyset \to \emptyset,
\end{aligned}$$

where i and j take all values in the ranges $0 \leq i \leq l$ and $0 \leq j \leq m$, is a modified circuit-ensemble equivalent to $\sigma(\boldsymbol{x}) \bullet \tau(\boldsymbol{x})$. The other cases are similar.

In the sequel, we use the notation $\boldsymbol{\emptyset}$ to denote the tuple $\emptyset, \ldots, \emptyset$ whose arity will be clear from context.

Lemma 3. *For any $\tau \in C[\boldsymbol{x}]$, with $\boldsymbol{x}$ of arity n, there exist $k > 0$ and $\tau_1, \ldots, \tau_k \in C[\,]$ with the following property: for all $m \in \mathbb{N}$ and all $\boldsymbol{s} \in (2^{\mathbb{N}})^n$ with $\boldsymbol{s}_{|m} = \boldsymbol{\emptyset}$, there exists i $(1 \leq i \leq k)$ such that $\tau(\boldsymbol{s})_{|m} = \tau_i(\,)_{|m}$.*

Proof. By Lemma 2, let $\tau(\boldsymbol{x})$ be equivalent to $\varepsilon(\boldsymbol{x}) \in E^\circ[\boldsymbol{x}]$, and let $\varepsilon(\boldsymbol{x}) = \{\alpha_i(\boldsymbol{x}) \to \tau'_i(\boldsymbol{x}) \mid 1 \leq i \leq k\}$. Let $\tau_1, \ldots, \tau_k$ be obtained by replacing all variables in $\tau'_1, \ldots, \tau'_k$ with the constant $\emptyset$. Fix any $m \in \mathbb{N}$ and $\boldsymbol{s} \in (2^{\mathbb{N}})^n$ with $\boldsymbol{s}_{|m} = \boldsymbol{\emptyset}$. Let i be the unique integer $(1 \leq i \leq k)$ such that $\alpha_i(\boldsymbol{s}) = \top$. Then, $\tau(\boldsymbol{s})_{|m} = \tau'_i(\boldsymbol{s})_{|m} = \tau'_i(\boldsymbol{\emptyset})_{|m}$, by Lemma 1 (since $\boldsymbol{s}_{|m} = \boldsymbol{\emptyset}$). Thus, $\tau(\boldsymbol{s})_{|m} = \tau_i(\,)_{|m}$, as required.

Lemma 4. *For any $\tau \in C[\boldsymbol{x}]$, with $\boldsymbol{x}$ of arity n, the family of sets $\{\tau(\boldsymbol{s})_{|m} \mid m \geq 0,\ \boldsymbol{s} \in (2^{\mathbb{N}})^n$ s.t. $\boldsymbol{s}_{|m} = \boldsymbol{\emptyset}\}$ is the union of finitely many chains under inclusion.*

Proof. Let $\tau_1, \ldots, \tau_k$ be as in Lemma 3. The collection $\{\tau_i(\,)_{|m}\}_{m \geq 0}$ is a chain under inclusion.

This shows that a function (from numbers to numbers) is not circuit-definable if it has an infinite range and sub-linear growth:

Theorem 2. *Let $f : \mathbb{N} \to \mathbb{N}$ be a function. If $\{f(n) \mid n \in \mathbb{N},\ f(n) < n\}$ is infinite, then f is not circuit-definable.*

Proof. Suppose, for contradiction, that $\tau(x)$ defines f. Consider an infinite collection M of numbers m such that the values $f(m+1)$ are pairwise distinct, with $f(m+1) \leq m$. Then

$$\begin{aligned}\{\{f(m+1)\} \mid m \in M\} &\subseteq \{\tau(\{n\})_{|m} \mid 0 \leq m < n\} \\ &\subseteq \{\tau(s)_{|m} \mid m \geq 0,\ s \in 2^{\mathbb{N}} \text{ s.t. } s_{|m} = \emptyset\}.\end{aligned}$$

But the infinite set of singletons $\{\{f(m+1)\} \mid m \in M\}$ is certainly not included in the union of any finite collection of chains, contrary to Lemma 4.

Corollary 2. *If $0 < \alpha < 1$, then the functions $f(n) = \lceil \alpha n \rceil$ and $f(n) = \lceil n^{\alpha} \rceil$ are not circuit-definable. If $1 < \beta$, then neither are:*

$$f(n) = \begin{cases} n-1 & \text{if } n > 0 \\ 0 & \text{otherwise} \end{cases} \qquad f(n) = \begin{cases} \lceil \log_{\beta} n \rceil & \text{if } n > 0 \\ 0 & \text{otherwise.} \end{cases}$$

Note that Theorem 2 fails if the condition that $\{f(n) \mid n \in \mathbb{N},\ f(n) < n\}$ is infinite is replaced by the condition that $\{n \in \mathbb{N} \mid f(n) < n\}$ is infinite. For example, we have already seen that the function $n \mapsto (n \mod p)$ is circuit-definable, for all $p > 1$. Thus, arithmetic circuits can compute remainders (for fixed divisors), but not quotients.

As a further corollary, we see that, while the circuit τ_{μ} given above computes minima, no arithmetic circuit computes maxima:

Corollary 3. *If $F : 2^{\mathbb{N}} \to 2^{\mathbb{N}}$ has the property that, for all finite, non-empty $s \subseteq \mathbb{N}$, $F(s) = \{\max(s)\}$, then F is not circuit-definable.*

Proof. Suppose that F is defined by a circuit $\tau(x)$. We observed above that the function d given in (2) is defined by the circuit τ_d. Then the circuit $\tau(\tau_d(x))$ maps any singleton $\{n\}$ $(n > 0)$ to $\{n-1\}$, contradicting Theorem 2.

We remark on another consequence of Lemma 2. This contrasts with the circuit-definability of the function $d(x)$ given in (2). Let the function $\downarrow\colon 2^{\mathbb{N}} \to 2^{\mathbb{N}}$ be given by $\downarrow(x) = \{n \in \mathbb{N} \mid \exists m \in x, n < m\}$.

Corollary 4. *There is no arithmetic circuit $\tau(x)$ such that, for all finite s, $\tau(s) = \downarrow(s)$.*

Proof. Suppose otherwise. By Lemma 2, let $\varepsilon(x)$ be a modified circuit-ensemble such that $\varepsilon(s) = \downarrow(s)$ for all finite s. For all even numbers i, define $s_i = \{m \in \mathbb{N} \mid m \text{ odd and } m < i\}$. Note that, for i, j even with $i < j$, we have $(s_i)_{|i} = (s_j)_{|i}$, but $(\downarrow(s_i))_{|i} = [0, i-1] \neq [0, i] = (\downarrow(s_j))_{|i}$. Since the number of clauses of ε is finite, we can find even numbers i, j, with $i < j$, such that s_i and s_j satisfy (the condition of) the same clause of ε. But then, by Lemma 1, $\varepsilon(s_i)_{|i} = \varepsilon(s_j)_{|i}$, contradicting the fact that $(\downarrow(s_i))_{|i} \neq (\downarrow(s_j))_{|i}$.

Thus, $d(x)$ is circuit-definable, but $\downarrow(x)$ is not, even for finite inputs.

Continuing with functions of sets of numbers, we now show that no such function is circuit-definable if it has a finite range and fails to converge on certain 'sparse' chains under inclusion. Let s be a finite set of numbers, t a set of numbers, and m a number. We write $s \sqsubseteq_m t$ if $m \geq \max(s)$ and $s_{|m} = t_{|m}$. In that case, we may think of the (possibly empty) interval $[\max(s)+1, m]$ as a 'buffer' following the largest element of s, in which no element of t occurs.

Theorem 3. *Let $F : 2^{\mathbb{N}} \to 2^{\mathbb{N}}$ be a function with finite range. And suppose that, for all finite $s \subseteq \mathbb{N}$ and all $m \geq \max(s)$, there exists $t \subseteq \mathbb{N}$ for which $s \sqsubseteq_m t$ and $F(t) \neq F(s)$. Then F is not circuit-definable.*

Proof. First, since the range of F is finite, fix a number q such that, if $F(s) \neq F(s')$, then $F(s)$ and $F(s')$ differ on some number less than or equal to q.

Suppose, for contradiction, that F is defined by a circuit $\tau(x)$; and, by Lemma 2, let $\varepsilon(x) = \{\alpha_i(x) \to \tau'_i(x) \mid 1 \leq i \leq k\}$ be a modified circuit-ensemble equivalent to $\tau(x)$. Let $\sigma_1(x), \ldots, \sigma_p(x)$ be a list, in some (arbitrary) order, of all the (modified) circuits $\sigma(x)$ such that $\sigma(x)$ occurs in a conjunct $\sigma = \emptyset$, $\sigma \neq \emptyset$, $0 \in \sigma$ or $0 \notin \sigma$ of any of the α_i. If $1 \leq j \leq p$, we write $\beta_j(x)$ for the condition $\sigma_j(x) \neq \emptyset$. If $s \subseteq \mathbb{N}$, let us say that the *profile* of s is the tuple of Boolean values $\pi(s) = \langle \beta_1(s), \ldots, \beta_p(s) \rangle$. We shall write $s \preceq t$ if (i) $s \cap \{0\} = t \cap \{0\}$, and (ii) for all j $(1 \leq j \leq p)$, $\beta_j(s) = \top$ implies $\beta_j(t) = \top$.

Now select a $\preceq$-maximal finite set s. That is: if u is finite with $s \preceq u$, then $\pi(s) = \pi(u)$. This must be possible, since the number of β_j is finite. For all j $(1 \leq j \leq p)$, let m_j be some element of $\sigma_j(s)$ if $\beta_j(s) = \top$, and 0 otherwise. Let $m = \max(s \cup \{m_j \mid 1 \leq j \leq p\} \cup \{q\})$. By the hypothesis of the lemma, let t be a subset of $\mathbb{N}$ such that $s \sqsubseteq_m t$ and $F(s) \neq F(t)$. Since $s \sqsubseteq_m t$, we have $s_{|m} = t_{|m}$. Then $\beta_j(s) = \top \Rightarrow m_j \in \sigma_j(s)_{|m} \Rightarrow m_j \in \sigma_j(t)_{|m}$ (by Lemma 1) $\Rightarrow \beta_j(t) = \top$. Moreover, we certainly have $s \cap \{0\} = t \cap \{0\}$. On the other hand, since $F(s) \neq F(t)$, it follows that s and t have different profiles. For otherwise, s and t would satisfy the same $\alpha_j(x)$ $(1 \leq j \leq k)$, whence $F(s) = \tau'_j(s)$ and $F(t) = \tau'_j(t)$. But $s_{|m} = t_{|m}$ would then imply $\tau'_j(s)_{|m} = \tau'_j(t)_{|m}$ (by Lemma 1), contradicting the fact that $F(s)$ and $F(t)$ disagree on some number less than or equal to $q \leq m$. Thus, there exists h $(1 \leq h \leq p)$ such that $\beta_h(t) = \top$ and $\beta_h(s) = \bot$. Let m_h be a number in $\beta_h(t)$, let $m^* = \max(m, m_h)$, and let $u = t_{|m^*}$. Then u is finite, $s \preceq u$ (by the same argument as for $s \preceq t$), and $\beta_h(u) = \top \neq \beta_h(s)$. This contradicts the $\preceq$-maximality of s among finite sets.

Corollary 5. *The function $F_{\text{fin}} : 2^{\mathbb{N}} \to 2^{\mathbb{N}}$ such that $F_{\text{fin}}(s) = \top$ if s is finite, and $F_{\text{fin}}(s) = \bot$ otherwise, is not circuit-definable. Further, no circuit-definable function $F : 2^{\mathbb{N}} \to 2^{\mathbb{N}}$ satisfies any of the following conditions for all finite $s \subseteq \mathbb{N}$:*

$$F(s) = \begin{cases} \top & \text{if } |s| \text{ is even} \\ \bot & \text{otherwise;} \end{cases} \qquad F(s) = \begin{cases} \top & \text{if } \max(s) \text{ even} \\ \bot & \text{otherwise;} \end{cases} \qquad F(s) = \begin{cases} \top & \text{if } \sum s \text{ even} \\ \bot & \text{otherwise.} \end{cases}$$

Proof. By considering the function $s \mapsto F(s) \bullet \{0\}$ if necessary, we may assume without loss of generality that the range of F is finite. Now apply Theorem 3.

Observe that the circuit $(\tau_e \cap x) \bullet \{0\}$, where τ_e is given in (1), defines a function F such that, for all finite s, $F(s) = \top$ if $\prod s$ is even, and $F(s) = \bot$ otherwise!

Corollary 6. *No circuit-definable function $F : 2^{\mathbb{N}} \to 2^{\mathbb{N}}$ satisfies the condition $F(s) = \{\sum s\}$ for all finite $s \subseteq \mathbb{N}$. No circuit-definable function $F : 2^{\mathbb{N}} \to 2^{\mathbb{N}}$ satisfies the condition $F(s) = \{|s|\}$ for all finite $s \subseteq \mathbb{N}$.*

Proof. Suppose, for contradiction, that the circuit $\tau(x)$ computes the sum of any finite argument s. Then the circuit $(\tau_e \cap \tau(x)) \bullet \{0\}$, where τ_e is given in (1), violates Corollary 5. The case of $|s|$ is handled similarly.

Acknowledgement

The authors gratefully acknowledge the support of the EPSRC, grant number EP/F069154/1.

References

1. C. Glaßer, K. Herr, C. Reitwießner, S. Travers, and M. Waldherr. Equivalence problems for circuits over sets of natural numbers. In *Computer Science Theory and Applications*, volume 4649 of *LNCS*, pages 127–138, Berlin, 2007. Springer Verlag.
2. C. Glaßer, C. Reitwießner, S. Travers, and M. Waldherr. Satisfiability of algebraic circuits over sets of natural numbers. In *Proceedings of FSTTCS 2007*, volume 4855 of *LNCS*, pages 253–264, Berlin, 2007. Springer Verlag.
3. K. Harrow. The bounded arithmetic hierarchy. *Information and Control*, 36:102–117, 1978.
4. A. Jeż and A. Okhotin. Complexity of solutions of equations over sets of natural numbers. In S. Albers and P. Weil, editors, *Proceedings of STACS 2008*, volume 08001 of *Dagstuhl Seminar Proceedings*, pages 373–384. Internationales Begegnungs- und Forschungszentrum für Informatik (IBFI), Schloss Dagstuhl, 2008.
5. A. Jeż and A. Okhotin. On the computational completeness of equations over sets of natural numbers. In L. Aceto, I. Damgård, L. Goldberg M. Halldórsson, A. Ingólfsdóttir, and I. Walukiewicz, editors, *Proceedings of ICALP 2008, Part II*, volume 5126 of *LNCS*, pages 63–74. Springer, 2008.
6. P. McKenzie and K. Wagner. The complexity of membership problems for circuits over sets of natural numbers. In H. Alt and M. Habib, editors, *Proceedings of STACS 2003*, volume 2607 of *LNCS*, pages 571–582. Springer–Verlag, 2003.
7. L. Stockmeyer and A. Meyer. Word problems requiring exponential time (preliminary report). In *Proceedings of the Fifth Annual ACM Symposium on Theory of Computing*, pages 1 – 9. ACM Digital Library, 1973.
8. S. Travers. The complexity of membership problems for circuits over sets of integers. *Theoretical Computer Science*, 369:211–229, 2006.
9. K. Yang. Integer circuit evaluation is PSPACE-complete. In *Proceedings, 15th IEEE Conference on Computational Complexity*, pages 204–211. IEEE, 2000.