Stronger Separation of Analog Neuron Hierarchy by Deterministic Context-Free Languages

Jiří Šíma[a]

[a]*Institute of Computer Science of the Czech Academy of Sciences, P. O. Box 5, 18207 Prague 8, Czech Republic,*

Abstract

The computational power of discrete-time recurrent neural networks (NNs) with the saturated-linear activation function depends on the descriptive complexity of their weight parameters encoding the NN program. In order to study the power of increasing analogicity in NNs between integer (finite automata) and arbitrary rational weights (Turing machines), we have established the analog neuron hierarchy 0ANNs $\subset$ 1ANNs $\subset$ 2ANNs $\subseteq$ 3ANNs where αANN is a binary-state NN that is extended with $\alpha \geq 0$ extra analog-state neurons with rational weights. In our previous work, we have compared it to the traditional Chomsky hierarchy and separated its first two levels. The separation 1ANNs $\subsetneqq$ 2ANNs has been witnessed by the non-regular deterministic context-free language (DCFL) $L_\# = \{0^n 1^n \mid n \geq 1\}$ which cannot be recognized by any 1ANN even with real weights, while any DCFL is accepted by a 2ANN with rational weights. In this paper, we strengthen this separation by showing that any non-regular DCFL (DFCL$'$) cannot be recognized by 1ANNs with real weights, which means DCFL$'$s $\subset$ (2ANNs $\setminus$ 1ANNs), implying 1ANNs $\cap$ DCFLs = 0ANNs. For this purpose, we show that any 1ANN that would recognize a DFCL$'$ can be augmented to a larger 1ANN that would recognize $L_\#$, which does not exists.

Keywords: recurrent neural network, analog neuron hierarchy, deterministic context-free language, Chomsky hierarchy

Email address: sima@cs.cas.cz (Jiří Šíma)

1. Analog Neuron Hierarchy

In our previous work [1], we have established a neural network (NN) version of the traditional Chomsky hierarchy which is the so-called *analog neuron hierarchy* (ANH) 0ANNs $\subset$ 1ANNs $\subset$ 2ANNs $\subseteq$ 3ANNs where αANN is a recurrent binary-state NN that is extended with $\alpha \geq 0$ extra analog-state neurons with rational weights. This hierarchy fills the gap between binary-state NNs with integer weights (0ANNs), implementing finite automata (FA) [2, 3, 4, 5, 6, 7], and analog-state NNs with arbitrary rational weights which can simulate Turing machines (TMs) [4, 8] by using only three analog units (3ANNs) [1].

This study has primarily been motivated by theoretical issues of how the computational power of NNs [9] increases with enlarging analogicity when we change step by step from binary to analog states, or equivalently, from integer to arbitrary rational weights. In particular, the weights are mainly assumed to be just fixed fractions with a finite representation (i.e. a quotient of two integer constants) avoiding real numbers with infinite precision[1]. Hence, the states of added α analog units can thus be only rationals although the number of digits in the representation of analog values may increase (linearly) along a computation. Nevertheless, by bounding the precision of analog states, we would reduce the computational power of NNs to that of finite automata which could be implemented by binary states. This would not allow the study of analogicity phenomena such as the transition from integer to rational weights in NNs whose functionality (program) is after all encoded in numerical weights.

There is nothing suspicious about the fact that the precision of analog states in αANNs is not limited by a fixed constant in advance. The same is true in conventional abstract models of computation such as pushdown automata or Turing machines with unlimited (potentially infinite) size of stack or tape, respectively, whose limitation would lead to the collapse of Chomsky hierarchy to finite automata. Thus, the proposed abstract model of αANNs itself has been intended for measuring the expressive power of a binary-state NN to which analog neurons are added one by one, rather than for solving special-purpose practical tasks or biological modeling. These theoretical motivations are further discussed in detail within the context of

[1]Nevertheless, we formulate the present lower-bound results for arbitrary real weights which hold all the more so for rationals.

relevant literature in [1].

Nevertheless, as a secondary use, our theoretical analysis may potentially be relevant to practical hybrid NNs that combine binary and analog neurons in deep networks employing the LSTM, GRU or ReLU units [10], which have recently been studied [11, 12, 13]. Moreover, various models of memory-augmented recurrent NNs [14, 15, 16, 17] have been explored for their empirical as well as theoretical capability to learn context-free languages (CFLs) which require some stack-based mechanism. Modeling CFLs in recurrent NNs is of great interest for natural language processing [18], machine translation, and speech recognition, because natural languages exhibit the hierarchical structure including long-distance dependencies, counting, repetition etc. Since the analog-state neurons in αANNs can naturally be viewed as a stack memory augmented to binary-state NNs [1], the placement of CFLs in the ANH appears to be an important issue which is completely answered for the deterministic CFLs (DCFLs) in this paper.

In order to compare the ANH to the classical Chomsky hierarchy, we have characterized syntactically the class 1ANNs in terms of so-called cut languages [19], which has been exploited for deriving a sufficient condition when a 1ANN accepts only a regular language (REG). This condition defines the subclass QP-1ANNs = 0ANNs of so-called *quasi-periodic* 1ANNs whose parameters, depending on weights, are quasi-periodic in a numeral system with non-integer radix [20]. For example, this class includes the 1ANNs with the self-loop weight w of the only analog neuron such as $w = 1/n$ for some integer $n > 1$, or $w = 1/\varphi$ where $\varphi = (1+\sqrt{5})/2$ is the golden ratio. In addition, we have proven that any language accepted by 1ANNs online[2], is context-sensitive (CSL), which implies 1ANNs $\subset$ CSLs. On the other hand, there are examples of languages accepted by 1ANNs such as

$$L_1 = \left\{ x_1 \ldots x_n \in \{0,1\}^* \;\middle|\; \sum_{k=1}^{n} x_{n-k+1} \left(\tfrac{27}{8}\right)^{-k} < \tfrac{1}{4} \right\} \tag{1}$$

that are not context-free [19], which provides the separation 0ANNs $\subsetneqq$ 1ANNs.

Furthermore, we have proven that the (non-regular) deterministic context-

[2]In *online* input/output protocols, the time between reading two consecutive input symbols as well as the delay in outputting the result after an input has been read, is bounded by a constant, while in *offline* protocols these time intervals are not bounded.

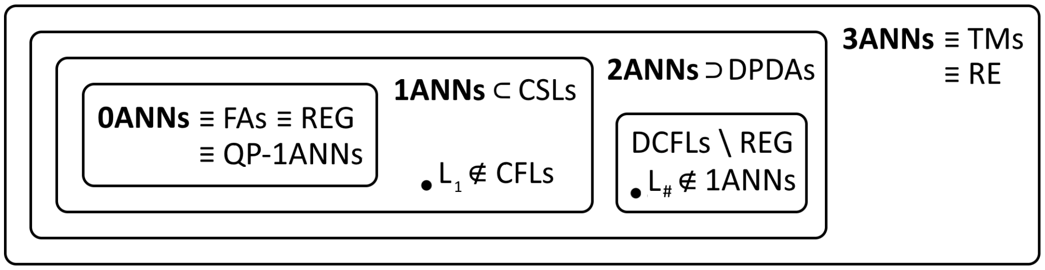

Figure 1: The analog neuron hierarchy

free language

$$L_{\#} = \{0^n 1^n \,|\, n \geq 1\}\,, \tag{2}$$

which contains the words of n zeros followed by n ones, cannot be recognized even offline[2] by any 1ANN with arbitrary real weights. Nevertheless, any DCFL can be recognized by a 2ANN by simulating a corresponding deterministic pushdown automaton (DPDA) [1]. This provides the separation 1ANNs $\subsetneqq$ 2ANNs. In addition, we have shown that any TM accepting a recursively enumerable language (RE) can be simulated by a 3ANN with a linear-time overhead [1], which proves the collapse of ANH at its third level. It appears that the ANH which is schematically depicted in Figure 1, is only partially comparable to that of Chomsky.

In this paper, we further study the relation between the ANH and the Chomsky hierarchy. We show that any non-regular DCFL (DCFL′) cannot be recognized online by 1ANNs with real weights, which provides the stronger separation

$$\text{DCFL}'\text{s} = (\text{DCFLs} \,\setminus\, \text{REG}) \subset (\text{2ANNs} \,\setminus\, \text{1ANNs})\,,$$

implying REG = 0ANNs = QP-1ANNs = 1ANNs ∩ DCFLs. Thus, the class of DCFL′s is contained in 2ANNs, having the empty intersection with 1ANNs, as depicted in Figure 1.

In order to prove this stronger lower bound on the computational power of 1ANNs, we have shown in another paper [21] that the language $L_{\#}$ defined in (2) is in some sense the simplest DCFL′ (a so-called DCFL′-simple problem), by reducing $L_{\#}$ to any language in DCFL′s. Namely, given any DCFL′ L, we can recognize the language $L_{\#}$ by a Mealy machine (a deterministic finite-state transducer) that is allowed to call a subroutine for deciding L (oracle) on its output extended with a few suffixes of constant length. In computability theory, this is a truth-table (a special case of Turing) reduction by a Mealy machine with oracle L. In this paper, we prove that such a reduction can be implemented by an online 1ANN. Thus, if the DCFL′ L were accepted by an online 1ANN, then we could recognize $L_{\#}$ by a 1ANN, which is a contradiction, implying that L cannot be accepted by any online 1ANN even with real weights.

The definition of DCFL′-simple problems which any DCFL′ language must include, that have been introduced in [21], appears to be an interesting achievement in formal language theory. A DCFL′-simple problem can be reduced to all the DCFL′ problems by the truth-table reduction using oracle Mealy machines, which is somewhat methodologically opposite to the usual hardness results in computational complexity theory where all problems in a class are reduced to its hardest problem such as in NP-completeness proofs. The concept of DCFL′-simple problems has been motivated by our analysis of the computational power of 1ANNs and this paper represents its first non-trivial application to proving the lower bounds. Our result can thus open a new direction of research in computability theory aiming towards the existence of the simplest problems in traditional complexity classes which could provide new proof techniques of extending a lower-bound result known for one problem to the whole class of problems.

The paper is organized as follows. In Section 2, we introduce basic definitions of a language acceptor based on 1ANNs. In Section 3, we prove two technical lemmas concerning the properties of 1ANNs which are used in Section 4 for the reduction of $L_{\#}$ to any DCFL′ by a 1ANN, implying that one extra analog neuron even with real weights is not sufficient for recognizing any DCFL′ online. Finally, we summarize the results and list some open problems in Section 5.

A preliminary version of this paper [22] contains only a sketch of the proof exploiting the representation of DCFLs by so-called deterministic monotonic restarting automata [23], while the complete argument for $L_{\#}$ to be the DCFL′-simple problem has eventually been achieved by using DPDAs [21].

2. Neural Language Acceptors with One Analog Unit

We specify the computational model of a *discrete-time binary-state recurrent neural network with one extra analog unit* (shortly, 1ANN), $\mathcal{N}$, which will be used as a formal language acceptor. The network $\mathcal{N}$ consists of $s \geq 1$ *units (neurons)*, indexed as $V = \{1, \ldots, s\}$. All the units in $\mathcal{N}$ are assumed to be binary-state (shortly *binary*) neurons (i.e. perceptrons, threshold gates) except for the last sth neuron which is an analog-state (shortly *analog*) unit. The neurons are connected into a directed graph representing an *architecture* of $\mathcal{N}$, in which each edge $(i,j) \in V^2$ leading from unit i to j is labeled with a real *weight* $w_{ji} \in \mathbb{R}$. The absence of a connection within the architecture corresponds to a zero weight between the respective neurons, and vice versa.

The computational dynamics of $\mathcal{N}$ determines for each unit $j \in V$ its *state (output)* $y_j^{(t)}$ at discrete time instants $t = 0, 1, 2, \ldots$. The states $y_j^{(t)}$ of the first $s-1$ binary neurons $j \in \tilde{V} = V \setminus \{s\}$ are Boolean values 0 or 1, whereas the output $y_s^{(t)}$ from the analog unit s is a real number from the unit interval $\mathbb{I} = [0,1]$. This establishes the *network state* $\mathbf{y}^{(t)} = \left(y_1^{(t)}, \ldots, y_{s-1}^{(t)}, y_s^{(t)}\right) \in \{0,1\}^{s-1} \times \mathbb{I}$ at each discrete time instant $t \geq 0$.

For notational simplicity, we assume the synchronous fully parallel mode without loss of efficiency [24]. At the beginning of a computation, the 1ANN $\mathcal{N}$ is placed in a predefined *initial state* $\mathbf{y}^{(0)} \in \{0,1\}^{s-1} \times \mathbb{I}$. At discrete time instant $t \geq 0$, an *excitation* of any neuron $j \in V$ is defined as

$$\xi_j^{(t)} = \sum_{i=0}^{s} w_{ji} y_i^{(t)} , \tag{3}$$

including a real *bias* value $w_{j0} \in \mathbb{R}$ which, as usually, can be viewed as the weight from a formal constant unit input $y_0^{(t)} \equiv 1$ for every $t \geq 0$. At the next instant $t+1$, all the neurons $j \in V$ compute their new outputs $y_j^{(t+1)}$ in parallel by applying an *activation function* $\sigma_j : \mathbb{R} \longrightarrow \mathbb{I}$ to $\xi_j^{(t)}$, that is,

$$y_j^{(t+1)} = \sigma_j\left(\xi_j^{(t)}\right) \quad \text{for every } j \in V \, . \tag{4}$$

For the neurons $j \in \tilde{V}$ with binary states $y_j \in \{0,1\}$, the *Heaviside* activation function $\sigma_j(\xi) = H(\xi)$ is used where

$$H(\xi) = \begin{cases} 1 & \text{for } \xi \geq 0 \\ 0 & \text{for } \xi < 0 \, , \end{cases} \tag{5}$$

while the analog unit $s \in V$ with real output $y_s \in \mathbb{I}$ employs the *saturated-linear* function $\sigma_s(\xi) = \sigma(\xi)$ where

$$\sigma(\xi) = \begin{cases} 1 & \text{for } \xi \geq 1 \\ \xi & \text{for } 0 < \xi < 1 \\ 0 & \text{for } \xi \leq 0 \, , \end{cases} \tag{6}$$

In this way, the new network state $\mathbf{y}^{(t+1)} \in \{0,1\}^{s-1} \times \mathbb{I}$ is determined at time $t+1$.

The computational power of NNs has been studied analogously to the traditional models of computations [9] so that the network is exploited as an acceptor of formal language $L \subseteq \Sigma^*$ over a finite alphabet $\Sigma = \{\lambda_1, \ldots \lambda_q\}$ composed of q letters (symbols). For the finite 1ANN $\mathcal{N}$, we use the following *online* input/output protocol employing its special binary neurons $X \subset \tilde{V}$ and $\text{nxt}, \text{out} \in \tilde{V}$. An input word (string) $\mathbf{x} = x_1 \ldots x_n \in \Sigma^n$ of arbitrary length $n \geq 0$, is sequentially presented to the network, symbol after symbol, via the first $q < s$ so-called *input neurons* $X = \{1, \ldots, q\} \subset \tilde{V}$, at the time instants $0 < \tau_1 < \tau_2 < \cdots < \tau_n$ after queried by $\mathcal{N}$. The neuron $\text{nxt} \in \tilde{V}$ is used by $\mathcal{N}$ to prompt a user to enter the next input symbol. Thus, once the prefix $x_1, \ldots, x_{k-1}$ of $\mathbf{x}$ for $1 \leq k \leq n$, has been read, the next input symbol $x_k \in \Sigma$ is presented to $\mathcal{N}$ at the time instant τ_k that is one computational step after $\mathcal{N}$ activates the neuron $\text{nxt} \in \tilde{V}$. This means that $\mathcal{N}$ signals

$$y_{\text{nxt}}^{(t-1)} = \begin{cases} 1 & \text{if } t = \tau_k \\ 0 & \text{otherwise} \end{cases} \quad \text{for } k = 1, \ldots, n\,. \tag{7}$$

We employ the popular *one-hot encoding* of alphabet Σ where each letter $\lambda_i \in \Sigma$ is represented by one input neuron $i \in X$ which is activated when the symbol λ_i is being read while, at the same time, the remaining input neurons $j \in X \setminus \{i\}$ do not fire. Namely, the states of input neurons $i \in X$, which represent a current input symbol $x_k \in \Sigma$ at the time instant τ_k, are thus externally set as

$$y_i^{(t)} = \begin{cases} 1 & \text{if } x_k = \lambda_i \text{ and } t = \tau_k \\ 0 & \text{otherwise} \end{cases} \quad \text{for } i \in X \text{ and } k = 1, \ldots, n\,. \tag{8}$$

At the same time, $\mathcal{N}$ carries out its computation deciding about each prefix of the input word $\mathbf{x}$ whether it belongs to L, which is indicated by the output neuron $\text{out} \in \tilde{V}$ when the next input symbol is presented which is one step after the neuron nxt is active according to (7):

$$y_{\text{out}}^{(\tau_{k+1})} = \begin{cases} 1 & \text{if } x_1 \ldots x_k \in L \\ 0 & \text{if } x_1 \ldots x_k \notin L \end{cases} \quad \text{for } k = 0, \ldots, n\,, \tag{9}$$

where $\tau_{n+1} > \tau_n$ is the time instant when the input word $\mathbf{x}$ is decided (e.g. formally define x_{n+1} to be any symbol from Σ to ensure the consistency with the input protocol (8) for $k = n + 1$). For instance, $y_{\text{out}}^{(\tau_1)} = 1$ iff the empty word ε belongs to L. We assume the online protocol where $\tau_{k+1} - \tau_k \leq \delta$ for every $k = 0, \ldots, n$ (formally $\tau_0 = 0$), is bounded by some integer constant $\delta > 0$, which ensures $\mathcal{N}$ halts on every input word $\mathbf{x} \in \Sigma^*$. We say that a language $L \subseteq \Sigma^*$ is *accepted (recognized)* by 1ANN $\mathcal{N}$, which is denoted as $L = \mathcal{L}(\mathcal{N})$, if for any input word $\mathbf{x} \in \Sigma^*$, $\mathcal{N}$ accepts $\mathbf{x}$ iff $\mathbf{x} \in L$.

3. Technical Properties of 1ANNs

In this section, we will prove two lemmas about technical properties of 1ANNs that will be used in Section 4 for implementing the reduction of $L_\#$ to any DCFL$'$ by a 1ANN. Namely, Lemma 1 shows that for any time constant $T > 0$, the state domain $\mathbb{I}$ of the only analog unit of a 1ANN $\mathcal{N}$ can be partitioned into finitely many subintervals so that the binary states during T consecutive computational steps by $\mathcal{N}$ are invariant to any initial analog state within each subinterval of this partition. Thus, one can extrapolate any computation by $\mathcal{N}$ for the next T computational steps only on the basis of information to which subinterval the initial analog state belongs. Lemma 2 then shows that such an extrapolation can be evaluated by a binary neural network, which ensures that the class of 1ANNs is in fact closed under the (right) quotient with a word.[3]

Lemma 1 *Let $\mathcal{N}$ be a 1ANN of size s neurons, which can be exploited as an acceptor of languages over an alphabet Σ for different initial states of $\mathcal{N}$. Then for every integer $T > 0$, there exists a partition $I_1 \cup I_2 \cup \cdots \cup I_p = \mathbb{I}$ of the unit interval $\mathbb{I} = [0,1]$ into $p = O\left(s2^{sT}\right)$ intervals such that for any input word $\mathbf{u} \in \Sigma^*$ of length $n = |\mathbf{u}|$ that meets $\tau_{n+1} \leq T$ according to the input/output protocol (7)–(9) for $\mathcal{N}$, the binary states $\tilde{\mathbf{y}}^{(t)} = \left(y_1^{(t)}, \ldots, y_{s-1}^{(t)}\right) = \tilde{\mathbf{y}}'^{(t)} \in \{0,1\}^{s-1}$ at any time instant $t \in \{0, 1, \ldots, \tau_{n+1}\}$ coincide for any two initial states $\mathbf{y}^{(0)}, \mathbf{y}'^{(0)} \in \{0,1\}^{s-1} \times \mathbb{I}$ with the same binary values $\tilde{\mathbf{y}}^{(0)} = \tilde{\mathbf{y}}'^{(0)} \in \{0,1\}^{s-1}$ and with analog values $y_s^{(0)}, y_s'^{(0)} \in I_r$ from the same interval I_r for some index $r \in \{1, \ldots, p\}$.*

Proof. Let $T > 0$ be an integer, $\mathbf{y}^{(0)} \in \{0,1\}^{s-1} \times \mathbb{I}$ be an initial state of $\mathcal{N}$, and $\mathbf{u} \in \Sigma^*$ of length $n = |\mathbf{u}|$ be an input word that meets $\tau_{n+1} \leq T$

[3]The (right) quotient of language L with a word $\mathbf{u}$ is the language $L/\mathbf{u} = \{\mathbf{x} \mid \mathbf{x}\cdot\mathbf{u} \in L\}$.

according to the input/output protocol (7)–(9) for $\mathcal{N}$. Assume that

$$0 < \xi_s^{(t-1)} < 1 \quad \text{for every } t = 1, \ldots, \tau - 1 \tag{10}$$

for some τ such that $0 \leq \tau < \tau_{n+1}$, which implies $y_s^{(t)} = \xi_s^{(t-1)}$ for every $t = 1, \ldots, \tau - 1$, according to (4) and (6), and hence, for $\tau > 0$,

$$\begin{aligned} \xi_s^{(\tau-1)} &= \sum_{i=0}^{s-1} w_{si} y_i^{(\tau-1)} + w_{ss} y_s^{(\tau-1)} \\ &= \sum_{i=0}^{s-1} w_{si} y_i^{(\tau-1)} + w_{ss} \left(\sum_{i=0}^{s-1} w_{si} y_i^{(\tau-2)} + w_{ss} y_s^{(\tau-2)} \right) \\ \ldots &= \sum_{t=0}^{\tau-1} \left(\sum_{i=0}^{s-1} w_{si} y_i^{(t)} \right) w_{ss}^{\tau-t-1} + w_{ss}^{\tau} y_s^{(0)} \,. \end{aligned} \tag{11}$$

Note that formula (11) reduces to

$$\xi_s^{(\tau-1)} = \sum_{i=0}^{s-1} w_{si} y_i^{(\tau-1)} \,, \tag{12}$$

when $w_{ss} = 0$.

First assume $0 < \xi_s^{(\tau-1)} < 1$ when $\tau > 0$, which implies

$$y_s^{(\tau)} = \xi_s^{(\tau-1)} = \sum_{t=0}^{\tau-1} \left(\sum_{i=0}^{s-1} w_{si} y_i^{(t)} \right) w_{ss}^{\tau-t-1} + w_{ss}^{\tau} y_s^{(0)} \tag{13}$$

according to (4), (6), and (11). For any binary neuron $j \in \tilde{V}$, we have

$$y_j^{(\tau+1)} = 1 \quad \text{iff} \quad \xi_j^{(\tau)} = \sum_{i=0}^{s-1} w_{ji} y_i^{(\tau)} + w_{js} y_s^{(\tau)} \geq 0 \tag{14}$$

according to (4) and (5). By plugging (13) into (14), we obtain

$$y_j^{(\tau+1)} = 1 \quad \text{iff} \quad \sum_{i=0}^{s-1} w_{ji} y_i^{(\tau)} + w_{js} \sum_{t=0}^{\tau-1} \left(\sum_{i=0}^{s-1} w_{si} y_i^{(t)} \right) w_{ss}^{\tau-t-1} + w_{js} w_{ss}^{\tau} y_s^{(0)} \geq 0 \,, \tag{15}$$

which can be rewritten for $w_{ss} \neq 0$ and $w_{js} \neq 0$ as

$$y_j^{(\tau+1)} = 1 \quad \text{iff}$$

$$\sum_{t=0}^{\tau-1} \left(-\sum_{i=0}^{s-1} \frac{w_{si}}{w_{ss}} y_i^{(t)} \right) w_{ss}^{-t} - \sum_{i=0}^{s-1} \frac{w_{ji}}{w_{js}} y_i^{(\tau)} w_{ss}^{-\tau} \begin{cases} \geq y_s^{(0)} & \text{if } w_{js} w_{ss}^{\tau} < 0 \\ \leq y_s^{(0)} & \text{if } w_{js} w_{ss}^{\tau} > 0 \,. \end{cases} \tag{16}$$

For $w_{ss} = 0$ and $\tau > 0$, condition (15) reduces to

$$y_j^{(\tau+1)} = 1 \quad \text{iff} \quad \sum_{i=0}^{s-1} w_{ji} y_i^{(\tau)} + w_{js} \left(\sum_{i=0}^{s-1} w_{si} y_i^{(\tau-1)} \right) \geq 0 \tag{17}$$

which means the state $y_j^{(\tau+1)}$ depends in fact only on the binary states $\tilde{\mathbf{y}}^{(\tau)}$ and $\tilde{\mathbf{y}}^{(\tau-1)}$ where $\tilde{\mathbf{y}}^{(t)} = \left(y_1^{(t)}, \ldots, y_{s-1}^{(t)} \right) \in \{0,1\}^{s-1}$. Similarly, for $w_{js} = 0$, we have

$$y_j^{(\tau+1)} = 1 \quad \text{iff} \quad \sum_{i=0}^{s-1} w_{ji} y_i^{(\tau)} \geq 0 \tag{18}$$

when the state $y_j^{(\tau+1)}$ depends only on the binary states $\tilde{\mathbf{y}}^{(\tau)}$.

For the case when either $\xi_s^{(\tau-1)} \leq 0$ or $\xi_s^{(\tau-1)} \geq 1$ for $w_{ss} \neq 0$ and $\tau > 0$, we have

$$y_s^{(\tau)} = 0 \quad \text{iff} \quad \sum_{t=0}^{\tau-1} \left(-\sum_{i=0}^{s-1} \frac{w_{si}}{w_{ss}} y_i^{(t)} \right) w_{ss}^{-t} \begin{cases} \geq y_s^{(0)} & \text{if } w_{ss}^{\tau} > 0 \\ \leq y_s^{(0)} & \text{if } w_{ss}^{\tau} < 0 \end{cases} \tag{19}$$

$$y_s^{(\tau)} = 1 \quad \text{iff} \quad \frac{1}{w_{ss}^{\tau}} + \sum_{t=0}^{\tau-1} \left(-\sum_{i=0}^{s-1} \frac{w_{si}}{w_{ss}} y_i^{(t)} \right) w_{ss}^{-t} \begin{cases} \geq y_s^{(0)} & \text{if } w_{ss}^{\tau} < 0 \\ \leq y_s^{(0)} & \text{if } w_{ss}^{\tau} > 0 \,, \end{cases} \tag{20}$$

respectively, according to (4), (6), and (11).

Altogether, for any $\ell \in V$ such that $w_{\ell s} \neq 0$, and $\tilde{\mathbf{y}} = (y_1, \ldots, y_{s-1}) \in \{0,1\}^{s-1}$, we denote

$$\zeta_\ell \left(\tilde{\mathbf{y}} \right) = -\sum_{i=0}^{s-1} \frac{w_{\ell i}}{w_{\ell s}} y_i \,, \tag{21}$$

and

$$z_\ell \left(\tilde{\mathbf{y}}_0, \tilde{\mathbf{y}}_1, \ldots, \tilde{\mathbf{y}}_m \right) = \sum_{t=0}^{m-1} \zeta_s \left(\tilde{\mathbf{y}}_t \right) w_{ss}^{-t} + \zeta_\ell \left(\tilde{\mathbf{y}}_m \right) w_{ss}^{-\tau} \,, \tag{22}$$

for any $\tilde{\mathbf{y}}_0, \tilde{\mathbf{y}}_1, \ldots, \tilde{\mathbf{y}}_m \in \{0,1\}^{s-1}$ where $m \geq 0$, which reduces conditions (16) and (19), (20) with $w_{ss} \neq 0$ to

$$y_j^{(\tau+1)} = 1 \quad \text{iff} \quad z_j\left(\tilde{\mathbf{y}}^{(0)}, \tilde{\mathbf{y}}^{(1)}, \ldots, \tilde{\mathbf{y}}^{(\tau)}\right) \begin{cases} \geq y_s^{(0)} & \text{if } w_{js}w_{ss}^{\tau} < 0 \\ \leq y_s^{(0)} & \text{if } w_{js}w_{ss}^{\tau} > 0 \end{cases} \tag{23}$$

for $j \in \tilde{V}$ such that $w_{js} \neq 0$, and

$$y_s^{(\tau)} = 0 \quad \text{iff} \quad z_s\left(\tilde{\mathbf{y}}^{(0)}, \tilde{\mathbf{y}}^{(1)}, \ldots, \tilde{\mathbf{y}}^{(\tau-1)}\right) \begin{cases} \geq y_s^{(0)} & \text{if } w_{ss}^{\tau} > 0 \\ \leq y_s^{(0)} & \text{if } w_{ss}^{\tau} < 0 \end{cases} \tag{24}$$

$$y_s^{(\tau)} = 1 \quad \text{iff} \quad \frac{1}{w_{ss}^{\tau}} + z_s\left(\tilde{\mathbf{y}}^{(0)}, \tilde{\mathbf{y}}^{(1)}, \ldots, \tilde{\mathbf{y}}^{(\tau-1)}\right) \begin{cases} \geq y_s^{(0)} & \text{if } w_{ss}^{\tau} < 0 \\ \leq y_s^{(0)} & \text{if } w_{ss}^{\tau} > 0\,, \end{cases} \tag{25}$$

for $\tau > 0$, respectively.

We define the set

$$\begin{aligned} Z &= \left(Z' \cap (\mathbb{I} \times \{-1,1\})\right) \cup \left\{(0,-1), (0,1), (1,-1), (1,1)\right\} \\ &= \left\{(a_1,b_1), (a_2,b_2), \ldots, (a_{p+1}, b_{p+1})\right\} \subset \mathbb{I} \times \{-1,1\} \end{aligned} \tag{26}$$

where

$$\begin{aligned} Z' &= \left\{ \left(z_j\left(\tilde{\mathbf{y}}_0, \ldots, \tilde{\mathbf{y}}_\tau\right), -\text{sgn}\left(w_{js}w_{ss}^{\tau}\right)\right) \,\middle|\, \begin{array}{c} j \in \tilde{V} \text{ s.t. } w_{js} \neq 0 \\ \tilde{\mathbf{y}}_0 \ldots, \tilde{\mathbf{y}}_\tau \in \{0,1\}^{s-1} \\ 0 \leq \tau < T \end{array} \right\} \\ &\bigcup \left\{ \left(z_s\left(\tilde{\mathbf{y}}_0, \ldots, \tilde{\mathbf{y}}_{\tau-1}\right), \text{sgn}\left(w_{ss}^{\tau}\right)\right) \,\middle|\, \begin{array}{c} \tilde{\mathbf{y}}_0 \ldots, \tilde{\mathbf{y}}_{\tau-1} \in \{0,1\}^{s-1} \\ 0 < \tau < T \end{array} \right\} \\ &\bigcup \left\{ \left(\frac{1}{w_{ss}} + z_s\left(\tilde{\mathbf{y}}_0, \ldots, \tilde{\mathbf{y}}_{\tau-1}\right), -\text{sgn}\left(w_{ss}^{\tau}\right)\right) \,\middle|\, \begin{array}{c} \tilde{\mathbf{y}}_0 \ldots, \tilde{\mathbf{y}}_{\tau-1} \in \{0,1\}^{s-1} \\ 0 < \tau < T \end{array} \right\} \end{aligned} \tag{27}$$

and sgn : $\mathbb{R} \to \{-1,0,1\}$ is the signum function. The set Z includes the $p+1$ pairs $(a_r, b_r) \in \mathbb{I} \times \{-1,1\}$ for $r = 1, \ldots, p+1$, which encode all the possible closed half-lines with the finite endpoints $a_r \in \mathbb{I} = [0,1]$, either $[a_r, +\infty)$ if $b_r = -1$, or $(-\infty, a_r]$ if $b_r = 1$, that may occur in conditions (23)–(25) determining the binary outputs $y_j^{(\tau+1)}, y_s^{(\tau)} \in \{0,1\}$ for the analog state $y_s^{(0)} \in \mathbb{I}$. Clearly, the number $|Z| = p+1$ of these half-lines can be bounded as

$$\begin{aligned} p+1 \;\leq\; & (s-1)\left(2^{s-1} + \left(2^{s-1}\right)^2 + \cdots + \left(2^{s-1}\right)^T\right) \\ & + 2\left(\left(2^{s-1}\right)^2 + \cdots + \left(2^{s-1}\right)^{T-1}\right) + 4 = O\left(s2^{sT}\right)\,. \end{aligned} \tag{28}$$

We also assume that the elements of Z are lexicographically sorted as

$$(a_1, b_1) < (a_2, b_2) < \cdots < (a_{p+1}, b_{p+1}) \tag{29}$$

which is used in the definition of the partition of the unit interval $\mathbb{I} = [0, 1] = I_1 \cup I_2 \cup \ldots \cup I_p$ into p intervals:

$$I_r = \begin{cases} [a_r, a_{r+1}) & \text{if } b_r = -1 \ \& \ b_{r+1} = -1 \\ [a_r, a_{r+1}] & \text{if } b_r = -1 \ \& \ b_{r+1} = 1 \\ (a_r, a_{r+1}) & \text{if } b_r = 1 \ \& \ b_{r+1} = -1 \\ (a_r, a_{r+1}] & \text{if } b_r = 1 \ \& \ b_{r+1} = 1 \end{cases} \quad \text{for } r = 1, \ldots, p\,. \tag{30}$$

Note that if $a_r = a_{r+1}$ for some $r \in \{1, \ldots, p\}$, then we know $-1 = b_r < b_{r+1} = 1$ due to Z is lexicographically sorted, which produces the degenerate interval $I_r = [a_r, a_r]$. Thus, $I_1 = [0, 0]$ and $I_p = [1, 1]$ because $(0, -1), (0, 1), (1, -1), (1, 1) \in Z$ according to (26).

We will show that for any initial binary states $\tilde{\mathbf{y}}^{(0)} \in \{0, 1\}^{s-1}$, the binary output $y_j^{(\tau)} \in \{0, 1\}$ from any neuron $j \in \tilde{V}$ after the next τ computational steps of $\mathcal{N}$ where $0 \leq \tau \leq \tau_{n+1} \leq T$, is the same for all initial analog values $y_s^{(0)}$ within the whole interval I_r, which means $\tilde{\mathbf{y}}^{(\tau)}$ depends only on $\tilde{\mathbf{y}}^{(0)}$ and $r \in \{1, \ldots, p\}$ such that $y_s^{(0)} \in I_r$. We proceed by induction on $\tau = 0, \ldots, \tau_{n+1}$ satisfying (10). The base case is trivial since $\tilde{\mathbf{y}}^{(0)}$ does not depend on $y_s^{(0)}$ at all. Thus assume in the induction step that the statement holds for $\tilde{\mathbf{y}}^{(0)}, \tilde{\mathbf{y}}^{(1)}, \ldots, \tilde{\mathbf{y}}^{(\tau)}$ that meet (10), where $0 \leq \tau < \tau_{n+1}$.

Consider first the case when either $\tau = 0$ or $0 < \xi_s^{(\tau-1)} < 1$ for $\tau > 0$ which ensures (13) and extends the validity of condition (10) for τ replaced by $\tau + 1$ in the next inductive step. Further assume $w_{ss} \neq 0$ and let $j \in \tilde{V}$ be any binary neuron. For $w_{js} = 0$, the state $y_j^{(\tau+1)}$ is clearly determined only by $\tilde{\mathbf{y}}^{(\tau)}$ according to (18). For $w_{js} \neq 0$, the binary state $y_j^{(\tau+1)} \in \{0, 1\}$ depends on whether the initial analog output $y_s^{(0)} \in \mathbb{I}$ lies on the corresponding half-line from Z with the endpoint $z_j(\tilde{\mathbf{y}}^{(0)}, \tilde{\mathbf{y}}^{(1)}, \ldots, \tilde{\mathbf{y}}^{(\tau)})$, according to (23), which holds within the whole interval $I_r \ni y_s^{(0)}$, since the endpoints $z_j\left(\tilde{\mathbf{y}}_0, \tilde{\mathbf{y}}_1, \ldots, \tilde{\mathbf{y}}_\tau\right)$ of all the possible half-lines in condition (23) for $0 \leq \tau < T$, are taken into account in the definition (26), (27) determining the partition (30) of the analog state domain $\mathbb{I}$. Thus, $\tilde{\mathbf{y}}^{(\tau+1)}$ depends only on $\tilde{\mathbf{y}}^{(\tau)}$ and I_r containing $y_s^{(0)}$, and hence, only on $\tilde{\mathbf{y}}^{(0)}$ and $r \in \{1, \ldots, p\}$ such that $y_s^{(0)} \in I_r$, by induction hypothesis. For $w_{ss} = 0$, we know that $\tilde{\mathbf{y}}^{(\tau+1)}$ depends only on $\tilde{\mathbf{y}}^{(\tau)}$ and $\tilde{\mathbf{y}}^{(\tau-1)}$ according to (17), which proves the assertion for $\tau > 0$ by induction hypothesis, while for $\tau = 0$ the argument is the same as for $w_{ss} \neq 0$ since condition (23) makes still sense for $\tau = 0$. This completes the induction step for $\tau = 0$ or $0 < \xi_s^{(\tau-1)} < 1$ for $\tau > 0$.

In the case when either $\xi_s^{(\tau-1)} \leq 0$ or $\xi_s^{(\tau-1)} \geq 1$ for $\tau > 0$, we know the analog output $y_s^{(\tau)} \in \{0,1\}$ is, in fact, binary, satisfying (24) or (25) when $w_{ss} \neq 0$, respectively, which means $y_s^{(0)} \in \mathbb{I}$ lies on the corresponding half-line from Z with the endpoint $z_s(\tilde{\mathbf{y}}^{(0)}, \tilde{\mathbf{y}}^{(1)}, \ldots, \tilde{\mathbf{y}}^{(\tau-1)})$. This holds within the whole interval $I_r \ni y_s^{(0)}$, since the endpoints $z_s\left(\tilde{\mathbf{y}}_0, \tilde{\mathbf{y}}_1, \ldots, \tilde{\mathbf{y}}_{\tau-1}\right)$ of all the possible half-lines in conditions (24) and (25) for $0 < \tau < T$, are taken into account in the definition (26), (27) determining the partition (30). For $w_{ss} = 0$, the state $y_s^{(\tau)} \in \{0,1\}$ depends only on $\tilde{\mathbf{y}}^{(\tau-1)}$ according to (12). Thus, $\tilde{\mathbf{y}}^{(\tau+1)}$ is determined by the binary state $\mathbf{y}^{(\tau)} \in \{0,1\}^s$ that is guaranteed for the whole interval I_r containing $y_s^{(0)}$, and hence, $\tilde{\mathbf{y}}^{(\tau+1)}$ depends only on $\tilde{\mathbf{y}}^{(0)}$ and $r \in \{1, \ldots, p\}$ such that $y_s^{(0)} \in I_r$, by induction hypothesis. In addition, the same holds for the subsequent binary states $\tilde{\mathbf{y}}^{(\tau+2)}, \tilde{\mathbf{y}}^{(\tau+3)}, \ldots, \tilde{\mathbf{y}}^{(\tau_{n+1})}$ which are also determined by the binary state $\mathbf{y}^{(\tau)} \in \{0,1\}^s$ at the time instant τ, which completes the proof of Lemma 1. $\square$

Lemma 2 *Let $\mathcal{N}$ be a 1ANN which recognizes the language $L = \mathcal{L}(\mathcal{N}) \subseteq \Sigma^*$ over an alphabet Σ by using the online input/output protocol (7)–(9) satisfying $\tau_{k+1} - \tau_k \leq \delta$ for every $k \geq 0$ and some integer constant $\delta > 0$. Let $\mathbf{u}_1, \mathbf{u}_2 \in \Sigma^+$ be two nonempty strings which define the (right) quotients $L_1 = L/\mathbf{u}_1$ and $L_2 = L/(\mathbf{u}_2 \cdot \mathbf{u}_1)$ of L with $\mathbf{u}_1$ and $\mathbf{u}_2 \cdot \mathbf{u}_1$, respectively, where $L/\mathbf{u} = \{\mathbf{x} \in \Sigma^* \mid \mathbf{x} \cdot \mathbf{u} \in L\}$. Then there exists a 1ANN $\mathcal{N}'$ that accepts $\mathcal{L}(\mathcal{N}') = L_2 \setminus L_1$ respectively $\mathcal{L}(\mathcal{N}') = L_1 \setminus L_2$, with the delay of 3 computational steps, that is, the output protocol (9) is modified for $\mathcal{N}'$ as $y_{out'}^{(\tau_{k+1}+3)} = 1$ iff $x_1 \ldots x_k \in \mathcal{L}(\mathcal{N}')$, where $out' \in \tilde{V}'$ is the binary output neuron of $\mathcal{N}'$.*

Proof. We will construct the 1ANN $\mathcal{N}'$ such that $\mathcal{L}(\mathcal{N}') = L_2 \setminus L_1$ respectively $\mathcal{L}(\mathcal{N}') = L_1 \setminus L_2$ for the delayed output protocol, which contains $\mathcal{N}$ with s neurons as its subnetwork including the analog unit $s \in V$ shared by $\mathcal{N}'$, that is, $V \subset V' = \tilde{V}' \cup \{s\}$ for the corresponding sets of (binary) neurons. The architecture of $\mathcal{N}'$ is schematically depicted in Figure 2. Let $I_1 \cup I_2 \cup \cdots \cup I_p = \mathbb{I}$ be the partition of the state domain $\mathbb{I} = [0,1]$ of the analog unit $s \in V$ in $\mathcal{N}$ into p intervals according to Lemma 1 for $T = \delta \cdot (|\mathbf{u}_2\mathbf{u}_1| + 1)$. We encode these intervals by the $p+1$ pairs $(a_r, b_r) \in \mathbb{I} \times \{-1, 1\}$ for $r = 1, \ldots, p+1$, according to (30) where $a_r \in \mathbb{I}$ is the left endpoint of

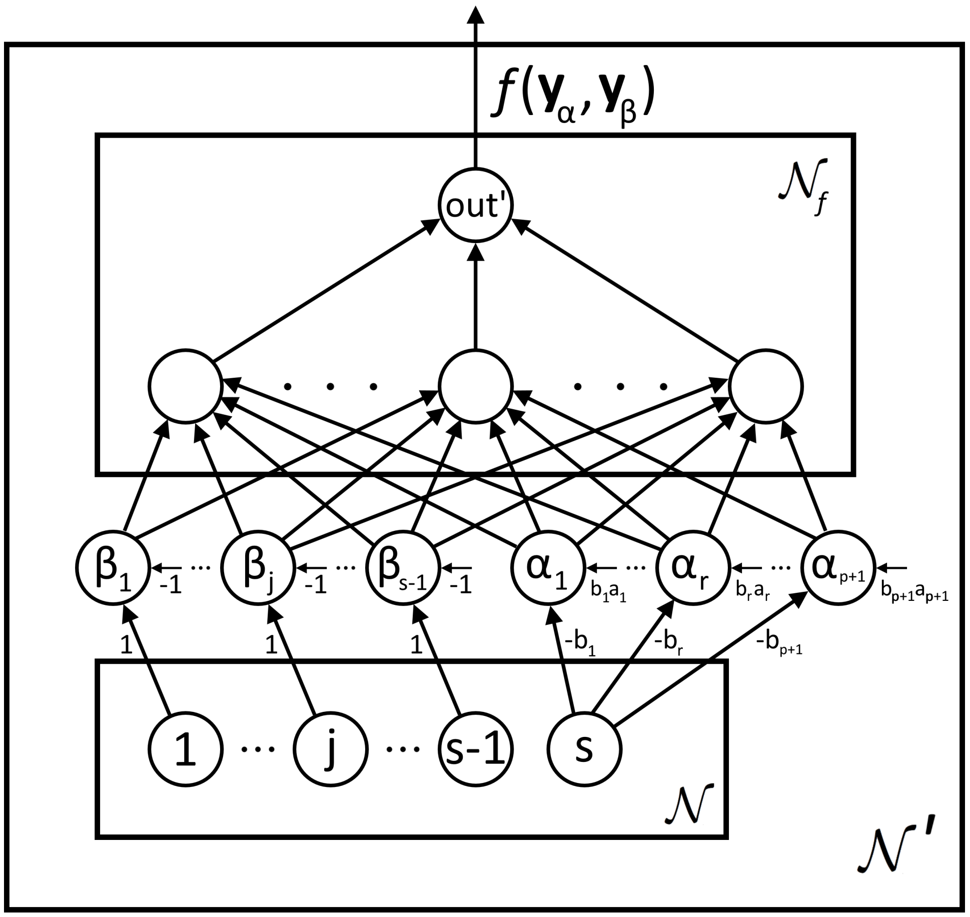

Figure 2: The 1ANN $\mathcal{N}'$ that, with the delay of 3 steps, accepts $\mathcal{L}(\mathcal{N}') = L_2 \setminus L_1$ respectively $\mathcal{L}(\mathcal{N}') = L_1 \setminus L_2$, where $L_1 = \mathcal{L}(\mathcal{N})/\mathbf{u}_1$ and $L_2 = \mathcal{L}(\mathcal{N})/(\mathbf{u}_2 \cdot \mathbf{u}_1)$.

I_r and $b_r = 1$ if I_r is left-open, while $b_r = -1$ if I_r is left-closed, which are lexicographically sorted according to (29).

For each pair (a_r, b_r) where $r \in \{1, \ldots, p+1\}$, we introduce one binary neuron $\alpha_r \in \tilde{V}'$ in $\mathcal{N}'$ to which the analog unit $s \in V$ is connected so that $y_{\alpha_r}^{(t_0+1)} = 1$ iff

$$y_s^{(t_0)} \begin{cases} \geq a_r & \text{for } b_r = -1 \\ \leq a_r & \text{for } b_r = 1 \end{cases} \tag{31}$$

iff $b_r a_r - b_r y_s^{(t_0)} \geq 0$, for any time instant $t_0 \geq 0$. According to (3)–(5), the bias and the corresponding weight of $\alpha_r \in \tilde{V}'$ from s are thus defined as $w'_{\alpha_r,0} = b_r a_r$ and $w'_{\alpha_r,s} = -b_r$, respectively (see Figure 2). Clearly, the binary states $\mathbf{y}_\alpha^{(t_0+1)} = \left(y_{\alpha_1}^{(t_0+1)}, \ldots, y_{\alpha_{p+1}}^{(t_0+1)}\right) \in \{0,1\}^{p+1}$ of neurons in $\alpha = \{\alpha_1, \ldots, \alpha_{p+1}\} \subset \tilde{V}'$ at time $t_0 + 1$ determine uniquely the index $r \in \{1, \ldots, p+1\}$ such that $y_s^{(t_0)} \in I_r$. In addition, for the synchronization purpose, we introduce the set $\beta = \{\beta_1, \ldots, \beta_{s-1}\} \subset \tilde{V}'$ of $s-1$ binary neurons in $\mathcal{N}'$ that, at the time instant t_0+1, copy the binary states $\tilde{\mathbf{y}}^{(t_0)} = \left(y_1^{(t_0)}, \ldots, y_{s-1}^{(t_0)}\right) \in \{0,1\}^{s-1}$ of $\mathcal{N}$ from the time instant t_0, which means $\mathbf{y}_\beta^{(t_0+1)} = \left(y_{\beta_1}^{(t_0+1)}, \ldots, y_{\beta_{s-1}}^{(t_0+1)}\right) = \tilde{\mathbf{y}}^{(t_0)}$. This can implemented by the biases $w'_{\beta_i,0} = -1$ and weights $w'_{\beta_i,i} = 1$ for every $i = 1, \ldots, s-1$, according to (3)–(5).

For any input word $\mathbf{x} \in \Sigma^*$ of length $n = |\mathbf{x}|$, let $t_0 \geq 0$ be a time instant when $\mathbf{x}$ has been read and still not decided by $\mathcal{N}$, that is, $\tau_n \leq t_0 < \tau_{n+1}$ according to the input protocol (7)–(8). According to Lemma 1, for the state $\mathbf{y}^{(t_0)} \in \{0,1\}^{s-1} \times \mathbb{I}$ that is considered as an initial state of $\mathcal{N}$ and for any nonempty suffix string $\mathbf{u} \in \Sigma^+$ added to $\mathbf{x}$ such that $\delta(|\mathbf{u}|+1) \leq T$, which is presented to $\mathcal{N}$ as an input since the time instant t_0, the binary states $\tilde{\mathbf{y}}^{(t_0+\tau)} = \left(y_1^{(t_0+\tau)}, \ldots, y_{s-1}^{(t_0+\tau)}\right) \in \{0,1\}^{s-1}$ at any time instant $t_0 + \tau \geq t_0$ of the ongoing computation of $\mathcal{N}$ over $\mathbf{u}$, are uniquely determined by the binary states $\mathbf{y}_\beta^{(t_0+1)} = \tilde{\mathbf{y}}^{(t_0)} = \left(y_1^{(t_0)}, \ldots, y_{s-1}^{(t_0)}\right) \in \{0,1\}^{s-1}$ of $\mathcal{N}$ and $\mathbf{y}_\alpha^{(t_0+1)} \in \{0,1\}^{p+1}$ due to $\mathbf{y}_\alpha^{(t_0+1)}$ is unique for $I_r \ni y_s^{(t_0)}$. In particular, the binary state $y_{\text{out}}^{(\tau_{n+|\mathbf{u}|})} \in \{0,1\}$ of the output neuron out $\in V$ in $\mathcal{N}$ after the suffix $\mathbf{u}$ has been read, where $t_0 < \tau_{n+|\mathbf{u}|} \leq t_0 + T$, is uniquely determined by the binary states $\mathbf{y}_\alpha^{(t_0+1)}$ and $\mathbf{y}_\beta^{(t_0+1)}$, according to the output protocol (9).

In other words, there is a Boolean function $f_\mathbf{u} : \{0,1\}^{p+s} \to \{0,1\}$ such that $f_\mathbf{u}\left(\mathbf{y}_\alpha^{(t_0+1)}, \mathbf{y}_\beta^{(t_0+1)}\right) = 1$ iff $\mathbf{x} \cdot \mathbf{u} \in \mathcal{L}(\mathcal{N})$ iff $\mathbf{x} \in L/\mathbf{u}$. We define the Boolean function $f : \{0,1\}^{p+s} \to \{0,1\}$ as the conjunction $f = \neg f_{\mathbf{u}_1} \wedge f_{\mathbf{u}_2 \cdot \mathbf{u}_1}$ where $\neg$ denotes the negation, or $f = f_{\mathbf{u}_1} \wedge \neg f_{\mathbf{u}_2 \cdot \mathbf{u}_1}$ which satisfies $f\left(\mathbf{y}_\alpha^{(t_0+1)}, \mathbf{y}_\beta^{(t_0+1)}\right) = 1$ iff $\mathbf{x} \in L_2 \setminus L_1$ or $\mathbf{x} \in L_1 \setminus L_2$, respectively. The Boolean function f can be computed by a binary-state two-layered neural network $\mathcal{N}_f$ that implements e.g. the disjunctive normal form of f. As depicted in Figure 2, the network $\mathcal{N}_f$ is integrated into $\mathcal{N}'$ so that the neurons $\alpha \cup \beta \subset \tilde{V}'$ create the input layer to $\mathcal{N}_f$, while the output of $\mathcal{N}_f$ represents the output neuron out$' \in \tilde{V}'$ of $\mathcal{N}'$ which thus produces $y_{\text{out}}^{(t_0+3)} = f\left(\mathbf{y}_\alpha^{(t_0+1)}, \mathbf{y}_\beta^{(t_0+1)}\right)$. Hence, $\mathcal{N}'$ recognizes $\mathcal{L}(\mathcal{N}') = L_2 \setminus L_1$ respectively $\mathcal{L}(\mathcal{N}') = L_1 \setminus L_2$ with the delay of 3 computational steps, which completes the proof of Lemma 2. $\square$

4. Separation of 1ANNs by DCFL′s

In this section, we will show the main result that any non-regular deterministic context-free language (DCFL′) cannot be recognized online by a binary-state 1ANN with one extra analog unit, which gives the stronger separation DCFL′s = (DCFLs $\setminus$ REG) $\subset$ (2ANNs $\setminus$ 1ANNs) in the ANH, implying 1ANNs $\cap$ DCFLs = 0ANNs = REG. The class DCFL′s is thus contained in 2ANNs with rational weights and has the empty intersection with 1ANNs, as depicted in Figure 1. For the proof, we will exploit the following fact that at least one DCFL′ cannot be recognized by any 1ANN, which has been shown in our previous work:

Theorem 1 *[1, Theorem 1] The non-regular deterministic context-free language $L_\# = \{0^n 1^n \,|\, n \geq 1\} \subset \{0,1\}^*$ over the binary alphabet cannot be recognized by any 1ANN with one extra analog unit having real weights.*

In order to generalize Theorem 1 to all DCFL′s, we have shown that $L_\#$ is in some sense the simplest DCFL′ which is contained in every DCFL′, as is formalized in the following Theorem 2.

Theorem 2 *[21, Theorem 1] Let $L \subseteq \Sigma^*$ be a non-regular deterministic context-free language over an alphabet Σ. Then there exist nonempty words $\mathbf{v}_1, \mathbf{v}_2, \mathbf{v}_3, \mathbf{v}_4, \mathbf{v}_5 \in \Sigma^+$ and a language $L' \in \{L, \overline{L}\}$ which is either L or its complement $\overline{L} = \Sigma^* \setminus L$, such that for all $m \geq 0$ and $n > 0$,*

$$\left(\mathbf{v}_1\mathbf{v}_2^m\mathbf{v}_3\mathbf{v}_4^{n-1}\mathbf{v}_5 \notin L' \textit{ and } \mathbf{v}_1\mathbf{v}_2^m\mathbf{v}_3\mathbf{v}_4^{n}\mathbf{v}_5 \in L'\right) \quad \textit{iff} \quad m = n\,. \tag{32}$$

This theorem is the basis for the novel concept of so-called DCFL′-simple problems, which has been inspired by this study of ANH and represents an interesting contribution to the formal language theory. Namely, the DCFL′-simple problem $L_\#$ can be reduced to every DCFL′ by the truth-table (Turing) reduction using oracle Mealy machines [21]. We will prove in the following Theorem 3 that this reduction can be implemented by 1ANNs, which generalizes Theorem 1 to any DCFL′s providing the stronger separation of 1ANNs in the ANH.

We outline the main idea of the proof. Suppose for a contradiction we have a 1ANN acceptor $\mathcal{N}$ for a DCFL′ $L = \mathcal{L}(\mathcal{N}) \subset \Sigma^*$. Theorem 2 provides $\mathbf{v}_1, \mathbf{v}_2, \mathbf{v}_3, \mathbf{v}_4, \mathbf{v}_5 \in \Sigma^+$ and $L' \in \{L, \overline{L}\}$ for this L. By Lemma 2, there is a 1ANN $\mathcal{N}'$ that accepts $\mathcal{L}(\mathcal{N}') = L_2 \setminus L_1$ if $L' = L$, or $\mathcal{L}(\mathcal{N}') = L_1 \setminus L_2$ if $L' = \overline{L}$, where $L_1 = L/\mathbf{v}_5$ and $L_2 = L/(\mathbf{v}_4 \cdot \mathbf{v}_5)$. A Mealy machine transforming an input 0^m1^n to $\mathbf{v}_1\mathbf{v}_2^m\mathbf{v}_3\mathbf{v}_4^{n-1}$ can be implemented by a binary-state NN. This NN is integrated in a bigger 1ANN $\mathcal{N}_\#$ containing $\mathcal{N}'$ as its subnetwork which implements an oracle for deciding the condition ($\mathbf{v}_1\mathbf{v}_2^m\mathbf{v}_3\mathbf{v}_4^{n-1}\mathbf{v}_5 \notin L'$ and $\mathbf{v}_1\mathbf{v}_2^m\mathbf{v}_3\mathbf{v}_4^{n}\mathbf{v}_5 \in L'$). According to (32), this means that the 1ANN $\mathcal{N}_\#$ recognizes $L_\#$, which is a contradiction to Theorem 1.

Theorem 3 *Any non-regular deterministic context-free language $L \subset \Sigma^*$ over an alphabet Σ cannot be recognized online by any 1ANN with one extra analog unit having real weights.*

Proof. Let $L \subset \Sigma^*$ be a non-regular deterministic context-free language over an alphabet Σ including $q > 0$ symbols. On the contrary assume that there is a 1ANN $\mathcal{N}$ that accepts $L = \mathcal{L}(\mathcal{N})$. Let $\mathbf{v}_1, \mathbf{v}_2, \mathbf{v}_3, \mathbf{v}_4, \mathbf{v}_5 \in \Sigma^+$ be the nonempty words and $L' \in \{L, \overline{L}\}$ be the language guaranteed by Theorem 2 for L, which satisfy condition (32). For any integer constant $c > 0$, we can assume without loss of generality that the strings $\mathbf{v}_i$ have the length at least c, that is, $|\mathbf{v}_i| \geq c$ for every $i = 1, \ldots, 5$, since otherwise we can replace $\mathbf{v}_1, \mathbf{v}_2, \mathbf{v}_3, \mathbf{v}_4, \mathbf{v}_5$ by $\mathbf{v}_1\mathbf{v}_2^c, \mathbf{v}_2^c, \mathbf{v}_2^c\mathbf{v}_3\mathbf{v}_4^c, \mathbf{v}_4^c, \mathbf{v}_4^c\mathbf{v}_5$, respectively. According to Lemma 2 for $L_1 = L/\mathbf{v}_5$ and $L_2 = L/(\mathbf{v}_4 \cdot \mathbf{v}_5)$, there is a 1ANN $\mathcal{N}'$ that accepts $\mathcal{L}(\mathcal{N}') = L_2 \setminus L_1$ if $L' = L$, or $\mathcal{L}(\mathcal{N}') = L_1 \setminus L_2$ if $L' = \overline{L}$, respectively, with the delay of 3 computational steps. It follows from (32) that for every $m \geq 0$ and $n > 0$,

$$\mathbf{v}_1\mathbf{v}_2^m\mathbf{v}_3\mathbf{v}_4^{n-1} \in \mathcal{L}(\mathcal{N}') \quad \text{iff} \quad m = n\,, \tag{33}$$

which will be used in the construction of a bigger 1ANN $\mathcal{N}_\#$ including $\mathcal{N}'$ as its subnetwork, that recognizes the language $L_\# = \{0^n1^n \,|\, n \geq 1\}$ over the binary alphabet $\{0, 1\}$. The architecture of $\mathcal{N}_\#$ is schematically depicted in Figure 3. We denote $\tilde{V}' \subset \tilde{V}_\#$ to be the corresponding sets of binary neurons in $\mathcal{N}'$ and $\mathcal{N}_\#$, respectively, while $\mathcal{N}_\#$ shares the only analog unit with $\mathcal{N}'$.

Namely, an input $\mathbf{x} = x_1 \ldots x_r \in \{0, 1\}^*$ to $\mathcal{N}_\#$ of the valid form 0^m1^n is translated to the string $\mathbf{v}_1\mathbf{v}_2^m\mathbf{v}_3\mathbf{v}_4^{n-1} \in \Sigma^*$ and presented to its subnetwork $\mathcal{N}'$ which decides online whether $m = n$ according (33). The result is used by $\mathcal{N}_\#$ for deciding whether $\mathbf{x} \in L_\#$. For this purpose, $\mathcal{N}_\#$ contains a finite

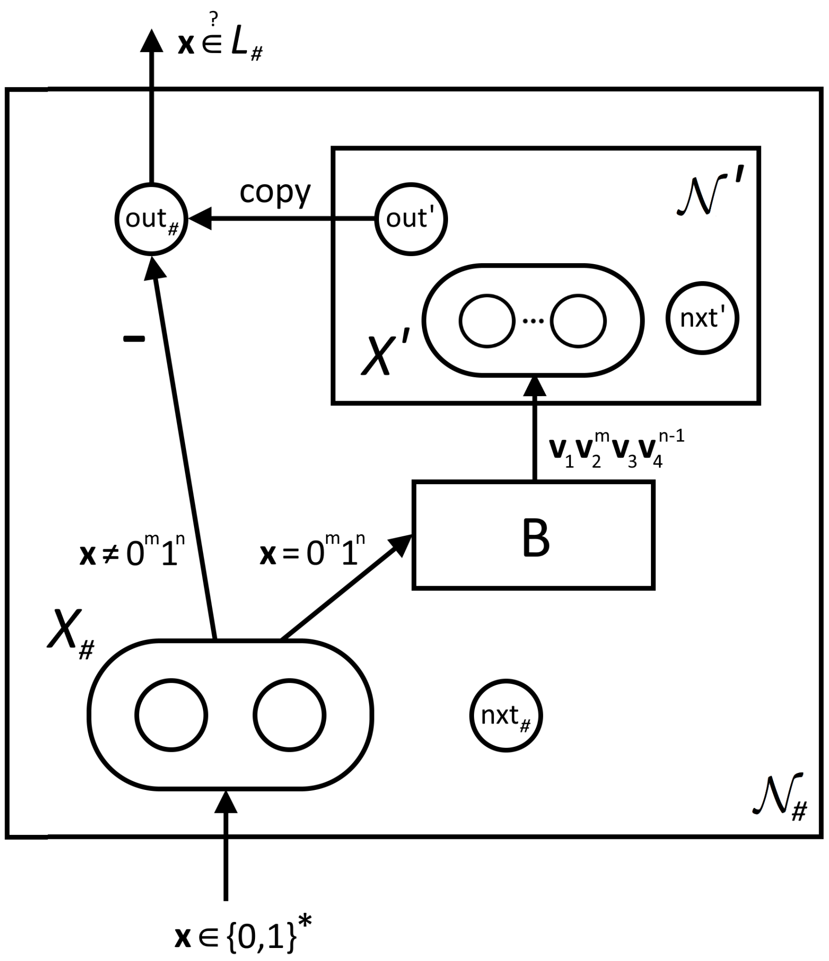

Figure 3: The reduction of $L_{\#}$ to a non-regular DCFL L.

buffer memory B organized as the queue of current input symbols from Σ^*, which are presented online, one by one, to $\mathcal{N}'$ through its q input neurons $X' \subset \tilde{V}'$ by using the one-hot encoding of Σ, when queried by $\mathrm{nxt}' \in \tilde{V}'$ according to the input protocol (7) and (8) for $\mathcal{N}'$.

At the beginning, B is initialized with the nonempty string $\mathbf{v}_1 \in \Sigma^+$ and $\mathcal{N}_{\#}$ queries on the first input bit $x_1 \in \{0,1\}$, that is, $y^{(0)}_{\mathrm{nxt}_{\#}} = 1$ where $\mathrm{nxt}_{\#} \in \tilde{V}_{\#}$, according to the input protocol (7) and (8) for $\mathcal{N}_{\#}$. Thus, at the time instant $\tau_1 = 1$, $\mathcal{N}_{\#}$ reads the first input bit x_1 through its two input neurons $X_{\#} \subset \tilde{V}_{\#}$ by using the one-hot encoding of $\{0,1\}$. If $x_1 = 1$, then $\mathbf{x} = 1\mathbf{x}' \notin L_{\#}$ is further rejected for any suffix $\mathbf{x}' \in \{0,1\}^*$ by clamping the state $y_{\mathrm{out}_\#}^{(t)} = 0$ of the output neuron $\mathrm{out}_\# \in \tilde{V}_\#$ in $\mathcal{N}_\#$ whereas $y_{\mathrm{nxt}_\#}^{(t)} = 1$, for every $t > 1$. If $x_1 = 0$, then $\mathcal{N}_\#$ writes the string $\mathbf{v}_2 \in \Sigma^+$ to B. At the same time, the computation of $\mathcal{N}'$ proceeds while reading its input from the buffer B when needed which is indicated by the neuron $\mathrm{nxt}' \in \tilde{V}'$ one computational step beforehand. Every time before B becomes empty, $\mathcal{N}_\#$ reads the next input bit $x_k \in \{0,1\}$ for $k > 1$ and writes the string $\mathbf{v}_2 \in \Sigma^+$ to B if $x_k = 0$, so that $\mathcal{N}'$ can smoothly continue in its computation. This is repeated until $\mathcal{N}_\#$ reads the input bit $x_{m+1} = 1$ for $m \geq 1$, which completes the first phase of the computation by $\mathcal{N}_\#$. In the course of this first phase, each prefix $0^k \notin L_\#$ of the input word $\mathbf{x}$, which is being read online by $\mathcal{N}_\#$, is rejected by putting the state $y_{\mathrm{out}_\#}^{(\tau_{k+1})} = 0$ of its output neuron $\mathrm{out}_\#$ for every $k = 1, \ldots, m$, according to the output protocol (9) for $\mathcal{N}_\#$.

At the beginning of the subsequent second phase when the input bit $x_{m+1} = 1$ has been read, $\mathcal{N}_\#$ writes the string $\mathbf{v}_3\mathbf{v}_4 \in \Sigma^+$ to B and continues uninterruptedly in the computation of $\mathcal{N}'$ over the input being read from the buffer B when required. Every time before B becomes empty which will precisely be specified below, $\mathcal{N}_\#$ reads the next input bit $x_{m+n} \in \{0,1\}$ for $n > 1$ and writes the string $\mathbf{v}_4 \in \Sigma^+$ to B if $x_{m+n} = 1$, so that $\mathcal{N}'$ can smoothly carry out its computation. If $x_{m+n} = 0$, then $\mathbf{x} = 0^m 1^{n-1} 0\, \mathbf{x}' \notin L_\#$ is further rejected for any suffix $\mathbf{x}' \in \{0,1\}^*$ by clamping the states $y_{\mathrm{out}_\#}^{(t)} = 0$ and $y_{\mathrm{nxt}_\#}^{(t)} = 1$ since that.

It follows that in the second phase, $\mathcal{N}'$ decides online for each $n > 0$ whether the input word $\mathbf{v}_1\mathbf{v}_2^m\mathbf{v}_3\mathbf{v}_4^{n-1} \in \Sigma^+$ of length $\ell = |\mathbf{v}_1\mathbf{v}_3| + m \cdot |\mathbf{v}_2| + (n-1)\cdot|\mathbf{v}_4|$ belongs to $\mathcal{L}(\mathcal{N}')$, where the result is indicated through its output neuron $\mathrm{out}' \in \tilde{V}'$ at the time instant $\tau'_{\ell+1}+3$ with the delay of 3 computational steps after the next symbol subsequent to $\mathbf{v}_1\mathbf{v}_2^m\mathbf{v}_3\mathbf{v}_4^{n-1}$ is read, according to the delayed output protocol (9) for $\mathcal{N}'$. For sufficiently large length $|\mathbf{v}_4| > 3$, the output neuron out' thus signals whether $\mathbf{v}_1\mathbf{v}_2^m\mathbf{v}_3\mathbf{v}_4^{n-1} \in \mathcal{L}(\mathcal{N}')$, while still reading the next string $\mathbf{v}_4$ corresponding to the last input bit $x_{m+n} = 1$ of the current input 0^m1^n to $\mathcal{N}_\#$. At the next time instant $\tau_{m+n+1} = \tau'_{\ell+1} + 4$, when the subsequent input bit $x_{m+n+1} \in \{0,1\}$ is presented to $\mathcal{N}_\#$, which is queried by $\mathcal{N}_\#$ via the state $y_{\mathrm{nxt}_\#}^{(\tau_{m+n+1}-1)} = 1$ of the neuron $\mathrm{nxt}_\#$ one step beforehand, the output neuron $\mathrm{out}_\#$ of $\mathcal{N}_\#$ copies the state of out', providing the result of the computation by $\mathcal{N}_\#$ over the input word $\mathbf{x} \in \{0,1\}^*$ according to the output protocol (9) for $\mathcal{N}_\#$. Namely, $y_{\mathrm{out}_\#}^{(\tau_{m+n+1})} = 1$ iff $\mathbf{v}_1\mathbf{v}_2^m\mathbf{v}_3\mathbf{v}_4^{n-1} \in \mathcal{L}(\mathcal{N}')$ iff $m = n$ iff $0^m1^n \in L_\#$ according to (33), which ensures $\mathcal{L}(\mathcal{N}_\#) = L_\#$.

The preceding online reduction of any input 0^m1^n for $\mathcal{N}_\#$ to the input

$\mathbf{v}_1\mathbf{v}_2^m\mathbf{v}_3\mathbf{v}_4^{n-1}$ for $\mathcal{N}'$ can clearly be realized by a finite automaton, including the implementation of the finite buffer memory B. This finite automaton can further be implemented by a binary-state neural network by using the standard constructions [3, 4, 5, 7], which is wired to the 1ANN $\mathcal{N}'$ in order to create the 1ANN $\mathcal{N}_\#$ recognizing the language $\mathcal{L}(\mathcal{N}_\#) = L_\#$ online, as described above. In particular, the synchronization of these two networks is controlled by their input/output protocols, while the operation of $\mathcal{N}'$ can suitably be slowed down for sufficiently large length of strings $\mathbf{v}_i$. However, we know by Theorem 1 that there is no 1ANN that accepts $L_\#$, which is a contradiction completing the proof of Theorem 3. $\square$

5. Conclusion

In this paper, we have refined the analysis of the computational power of discrete-time binary-state recurrent neural networks αANNs extended with α analog-state neurons by proving a stronger separation 1ANNs $\subsetneqq$ 2ANNs in the ANH depicted in Figure 1. Namely, we have shown that the class DCFL′s is contained in 2ANNs $\setminus$ 1ANNs, which implies 1ANNs $\cap$ DCFLs = 0ANNs = REG. For this purpose, we have reduced the DCFL′ $L_\# = \{0^n1^n \mid n \geq 1\}$, which is known to be not in 1ANNs [1], to any DCFL′.

It follows that $L_\#$ is in some sense the simplest language in the class DCFL′s. This is by itself an interesting contribution to computability theory, which has inspired the novel concept of a DCFL′-simple problem that can be reduced to any DCFL′ by the truth-table (Turing) reduction using oracle Mealy machines [21]. The present proof of the stronger separation 1ANNs $\subsetneqq$ 2ANNs thus represents the first non-trivial application of this concept. We believe that this approach can open a new direction of research aiming towards the existence of the simplest problems in traditional complexity classes as a methodological counterpart to the hardest problems in a class (such as NP-complete problems in NP) to which all the problems in this class are reduced. This could provide new proof techniques of extending a lower-bound result known for one problem to the whole class of problems. We conjecture that our separation result can further be strengthen to *nondeterministic* context-free languages (CFLs) by showing that $L_\#$ is even CFL′-simple, which would give 1ANNs $\cap$ CFL′s = 0ANNs where CFL′s = CFLs $\setminus$ REG. We note the interesting fact that $L_\#$ cannot be CSL′-simple since we know 1ANNs accept some non-regular CSLs (CSL′s) outside CFLs such as (1).

Moreover, it is an open question whether there is a non-context-sensitive language that can be accepted *offline* by a 1ANN, which does not apply to an online input/output protocol since we know online 1ANNs $\subset$ CSLs. Another important challenge for future research is the separation 2ANNs $\subsetneqq$ 3ANNs of the second level in the ANH and the relation between 2ANNs and CFLs, e.g. the issue of whether 2ANNs $\cap$ CFLs $\overset{?}{=}$ DCFLs.

Acknowledgments

The presentation of this paper benefited from valuable suggestions of anonymous reviewers. The research was done with institutional support RVO: 67985807 and partially supported by the grant of the Czech Science Foundation No. GA19-05704S.

References

[1] J. Šíma, Analog neuron hierarchy, Neural Networks 128 (2020) 199–218.

[2] N. Alon, A. K. Dewdney, T. J. Ott, Efficient simulation of finite automata by neural nets, Journal of the ACM 38 (2) (1991) 495–514.

[3] B. G. Horne, D. R. Hush, Bounds on the complexity of recurrent neural network implementations of finite state machines, Neural Networks 9 (2) (1996) 243–252.

[4] P. Indyk, Optimal simulation of automata by neural nets, in: Proceedings of the STACS 1995 Twelfth Annual Symposium on Theoretical Aspects of Computer Science, Vol. 900 of LNCS, LNCS, Springer, 1995, pp. 337–348.

[5] M. Minsky, Computations: Finite and Infinite Machines, Prentice-Hall, Englewood Cliffs, 1967.

[6] J. Šíma, Energy complexity of recurrent neural networks, Neural Computation 26 (5) (2014) 953–973.

[7] J. Šíma, J. Wiedermann, Theory of neuromata, Journal of the ACM 45 (1) (1998) 155–178.

[8] H. T. Siegelmann, E. D. Sontag, On the computational power of neural nets, Journal of Computer System Science 50 (1) (1995) 132–150.

[9] J. Šíma, P. Orponen, General-purpose computation with neural networks: A survey of complexity theoretic results, Neural Computation 15 (12) (2003) 2727–2778.

[10] J. Schmidhuber, Deep learning in neural networks: An overview, Neural Networks 61 (2015) 85–117.

[11] S. A. Korsky, R. C. Berwick, On the computational power of RNNs, arXiv:1906.06349 (2019).

[12] W. Merrill, Sequential neural networks as automata, arXiv:1906.01615 (2019).

[13] G. Weiss, Y. Goldberg, E. Yahav, On the practical computational power of finite precision RNNs for language recognition, in: Proceedings of the ACL 2018 Fifty-Sixth Annual Meeting of the Association for Computational Linguistics, Vol. 2, Association for Computational Linguistics, 2018, pp. 740–745.

[14] A. A. Mali, A. G. Ororbia, C. L. Giles, A neural state pushdown automata, IEEE Transactions on Artificial Intelligence 1 (3) (2020) 193–205.

[15] J. Stogin, A. A. Mali, C. L. Giles, Provably stable interpretable encodings of context free grammars in RNNs with a differentiable stack, arXiv:2006.03651 (2020).

[16] M. Suzgun, S. Gehrmann, Y. Belinkov, S. M. Shieber, Memory-augmented recurrent neural networks can learn generalized Dyck languages, arXiv:1911.03329 (2019).

[17] X. Yu, N. T. Vu, J. Kuhn, Learning the Dyck language with attention-based seq2seq models, in: Proceedings of the BlackboxNLP@ACL 2019 Second ACL 2019 Workshop on Analyzing and Interpreting Neural Networks for NLP, Association for Computational Linguistics, 2019, pp. 138–146.

[18] W. Merrill, Formal language theory meets modern NLP, arXiv:2102.10094 (2021).

[19] J. Šíma, Subrecursive neural networks, Neural Networks 116 (2019) 208–223.

[20] J. Šíma, P. Savický, Quasi-periodic β-expansions and cut languages, Theoretical Computer Science 720 (2018) 1–23.

[21] P. Jančar, J. Šíma, The simplest non-regular deterministic context-free language, in: Proceedings of the MFCS 2021 Forty-Sixth International Symposium on Mathematical Foundations of Computer Science, Vol. 202 of LIPIcs, Schloss Dagstuhl - Leibniz-Zentrum für Informatik, 2021, pp. 63:1–63:18.

[22] J. Šíma, M. Plátek, One analog neuron cannot recognize deterministic context-free languages., in: Proceedings of the ICONIP 2019 Twenty-Sixth International Conference on Neural Information Processing of the Asia-Pacific Neural Network Society, Part III, Vol. 11955 of LNCS, 2019, pp. 77–89.

[23] P. Jančar, F. Mráz, M. Plátek, J. Vogel, On monotonic automata with a restart operation, Journal of Automata, Languages and Combinatorics 4 (4) (1999) 287–311.

[24] P. Orponen, Computing with truly asynchronous threshold logic networks, Theoretical Computer Science 174 (1-2) (1997) 123–136.