Complex Systems **4** (1990) 509–518

Global Dynamics in Neural Networks II

Max Garzon
Stan Franklin
Institute for Intelligent Systems and Department of Mathematical Sciences, Memphis State University, Memphis, TN 38152 USA

Abstract. Determining just what tasks are computable by neural networks is of fundamental importance in neural computing. The configuration space of several models of parallel computation is essentially the Cantor middle-third set of real numbers. The Hedlund–Richardson theorem states that a transformation from the Cantor set to itself can be realized as the global dynamics of a cellular automaton if and only if it takes the quiescent configuration to itself, commutes with shifts, and is continuous in the product topology. An analogous theorem characterizing the realizability of self-mappings of the Cantor set as net-input global dynamics of neural networks has recently been established. Here we give a characterization of such realizability as the more natural activation global dynamics of neural networks. We also present such a characterization for realizability via global dynamics of more general automata networks. This dynamical systems approach to neural computing allows precise formulations of significant problems about the computational power of neural networks.

1. Introduction

Artificial neural networks (or connectionist models) [9], herein simply called neural networks, are often thought of as models of massively parallel computation. These neural networks can be naturally considered as dynamical systems [2]. In this paper, we are concerned with which dynamical systems can be realized by such neural networks. The motivation for this concern comes from similar work in cellular automata [5, 8].

Deterministic cellular automata [10–12] can be viewed as discrete dynamical systems with local dynamics defined by a single finite state machine. This local dynamics induces a global dynamics, i.e., a self-mapping of configuration (i.e., global state) space $\mathcal{C}$. Since not all self-maps of $\mathcal{C}$ arise this way, it is natural to ask what self-mappings of the configuration space

arise (or can be realized) as the global dynamics associated with some cellular automaton. This question is answered satisfactorily by Richardson's theorem [8]: it is necessary and sufficient for the mapping to preserve the *quiescent configuration* O, to commute with all the shift operators on the underlying cellular space, and, most importantly, to be continuous with respect to the product topology on configuration space. A one-dimensional version of this result follows from earlier work by Hedlund [5]. Since the set of local states of a cellular automaton is finite and the number of cells countably infinite, the configuration space in its product topology is homeomorphic to the Cantor middle-third set of real numbers [6, p. 97]. Thus, Richardson's Theorem can be regarded as a characterization of the realizability of self-maps of the Cantor set in terms of the global dynamics of a cellular automaton.

Synchronous, discrete neural networks can also be thought of as discrete dynamical systems with their local dynamics defined by local, nonuniform activation functions together with a local nonuniform input function consisting of the weighted sum of input values. However, unlike cellular automata, their local dynamics give rise to global dynamics both on the net-input space and on the activation space of the network. It is natural to ask again which self-maps of the net-input space or the activation space arise from such local nonuniform dynamics. As before, it would be desirable to characterize these self-maps in terms of a minimal set of global properties. Since neural networks also have finitely many local activation values and countably many cells, their net-input spaces and activation spaces are also homeomorphic to Cantor sets [6, p. 97]. Thus one may ask which self-maps of Cantor sets are realizable as the global dynamics of some neural network. This question has been previously answered for net-input global dynamics [2] (see theorem 3.1 below). Here we provide an answer for the activation global dynamics case.

Cellular automata and neural networks are each examples of a more general kind of model, *automata networks* [3, 7]. The analogous question of which self-maps of Cantor sets are realizable as the global dynamics of some automata network is also answered.

Viewing the global dynamics of neural networks as self-maps of the Cantor set leads to precise formulations of important problems about neural networks which are posed in the last section.

2. Definitions

Intuitively, a neural network consists of *cells* (also called units, nodes, etc.) or processors capable of some arithmetic, connected by links bearing *weights* . Cells sum their weighted inputs and apply an *activation function* to calculate their new activation state.

Formally, this means that a neural network is built on a *directed graph*, i.e., a structure consisting of a set of *vertices* V (representing the cells of the network) and *arcs* (directed edges representing the links of the network).

Each arc from vertex j into vertex i is labeled with a weight w_{ij}. In order for the network to be physically realizable, each vertex must have only finitely many incoming and outgoing arcs, i.e., the graph must be *locally finite*. In order to have full computational ability (i.e., that of Turing machines [1]) neural networks must allow for an arbitrarily large number of cells. We will simply assume a countably infinite number. To enable the necessary arithmetic, we assume that all activation values and weights come from a ring with unity. Thus, a neural network over a finite ring with unity R is built on a countably infinite, locally finite, arc-directed graph D.

In addition to the ring R and the graph D, a neural network needs for each cell i an *activation function* $f_i : R \to R$. We assume that each f_i satisfies $f_i(0) = 0$ (in order to avoid spontaneous generation of activation) and that $f_i(1) \neq 0$ (to avoid trivial networks). In theorem 3.1 below we further assume that each $f_i(1)$ is a unit in R.

The weighted sum of the inputs to each cell from its neighbors is given by a function called net_i, where i is the index of the given cell. Thinking of a neural network as a discrete dynamical system, at any time $t+1$ (t a nonnegative integer), the net-input at i is given by

$$net_i(t+1) = \sum_j w_{ij} a_j(t) \tag{2.1}$$

where

$$a_j(t) = f_j(net_j(t)) \tag{2.2}$$

is the activation of the cell j at time t, and the sum is taken over all cells j supporting arcs into i. These equations define the *local dynamics* of the network.

There are two possible neural network analogs of the notion of a configuration of a cellular automaton. One might look at the vector of all net-inputs to individual cells during a particular time step, or one might choose the vector of all activations of individual cells during that time step. At a fixed time t, the network has at each cell i its net input $net_i(t)$ and its activation $a_i(t)$. The vector of net-inputs x has in its ith component the net-input $x_i = net_(t)$ of the ith cell of D, a value from the ring R, and is thus a member of R^V. (Recall that R^V is the set of all functions from V to R, or the cartesian product of R with itself the cardinality of V times.) Similarly, the vector of activations $(a_i(t))$ is also a member of R^V. In the following we will use $\mathcal{C}$ to refer to activation space and call it the configuration (activation) space of the network, and we will use R^V to refer to the net-input space.

Each of these possibilities gives rise to its own global dynamics. At each tick of the time clock, the current net-input vector changes, as does the activation vector. These changes reflect the two distinct global dynamics of the network.

More formally, the family of functions f_i mapping R into R gives rise to a product function $F : R^V \to \mathcal{C}$ so that the following diagram commutes under composition of functions:

$$\begin{array}{ccc} & F & \\ R^V & \longrightarrow & \mathcal{C} \\ \pi_i \downarrow & & \downarrow \pi_i \\ R & \longrightarrow & R \\ & f_i & \end{array}$$

where π_i is the projection from R^V onto R. This means that for each cell i and each net-input vector $x \in R^V$, $F(x)_i = f_i(x_i)$.

We are now able to define the *net-input global dynamics* $T_{net} : R^V \to R^V$ of the network. For any net-input vector $x \in R^V$ put

$$T_{net}(x)_i = net_i(F(x)) \tag{2.3}$$

This gives rise to the following commutative diagram:

$$\begin{array}{ccc} & T_{net} & \\ R^V & \longrightarrow & R^V \\ F \downarrow & & \downarrow \pi_i \\ \mathcal{C} & \longrightarrow & R \\ & net_i & \end{array}$$

This global dynamics T_{net} describes the evolution of the entire network via the net-input vectors.

In a manner similar to the definition of F above, for each cell i, the net-input function net_i maps R into R. This family of functions gives rise to a product function net: $\mathcal{C} \to R^V$ so that the following diagram commutes under composition of functions:

$$\begin{array}{ccc} & net & \\ \mathcal{C} & \longrightarrow & R^V \\ id \downarrow & & \downarrow \pi_i \\ \mathcal{C} & \longrightarrow & R \\ & net_i & \end{array}$$

where π_i is the projection from R^V onto R and id is the identity map. This means that for each cell i and each activation vector $x \in \mathcal{C}$, $net_i(x) = \pi_i(net(x))$.

In an analogous fashion we now define the *activation global dynamics* $T_{act} : \mathcal{C} \to \mathcal{C}$ of the network. For any activation vector $x \in \mathcal{C}$ put

$$T_{act}(x)_i = f_i(net_i(x)) \tag{2.4}$$

This gives rise to the following commutative diagram:

$$\begin{array}{ccc} \mathcal{C} & \xrightarrow{T_{act}} & \mathcal{C} \\ net_i \downarrow & & \downarrow \pi_i \\ R & \xrightarrow[f_i]{} & R \end{array}$$

With these two notions of global dynamics defined for neural networks in place, we turn our attention to automata networks.

Like cellular automata and neural networks, each *automata network* is built on a *digraph* containing vertices representing the cells of the network and directed edges representing the links of the network. As in neural networks, this digraph can be quite irregular in structure. The processor at each vertex is a finite-state machine that takes its inputs from its neighboring cells at adjacent nodes in the network. Again, the network is assumed to be locally finite and to have a countably infinite number of cells. Also for physical realizability, we assume that the cardinality of the state sets is uniformly bounded. Hence one may assume that each Q_i is contained in a single finite state Q. Also assume that Q contains a common quiescent state denoted by 0.

Associated to every automata network, there is a *configuration space* $\prod_i Q_i$ consisting of all *configurations* (or *total states*, or *state vectors*) of the space. It is again homeomorphic to the Cantor set [6, p. 97] and will be denoted by $\mathcal{C}$.

An automata network operates as follows. Synchronously, the finite-state machine M_i occupying a vertex i of D looks up its input in the states $x_{i_1}, \ldots, x_{i_d}$ its neighbor cells and its own state x_i, then changes its state according to a prespecified local dynamics δ_i. Also, each δ_i preserves quiescent states, i.e., $\delta_i(0, 0, \ldots, 0) = 0$. The automata network performs its calculation by repeating this atomic move any (possibly very large) number of times.

These local functions induce a global dynamics

$$T : \mathcal{C} \to \mathcal{C}$$

$$T(x)_i = \delta_i(x_i, x_{i_1}, \ldots, x_{i_d}) \tag{2.5}$$

where $i_1, \ldots, i_d$ are the cells adjacent to i, and x_j is the current state of cell j.

3. Results

The realizability of a self-map of $\mathcal{C}$ as net-input global dynamics is characterized by the following theorem, which was proven in [2] for R a field. The same proof applies to the slightly more general case stated next.

Theorem 3.1. *A self-map $T : \mathcal{C} \to \mathcal{C}$ is realizable as the net-input global dynamics of a neural network with activation functions $\{f_i : R \to R\}$ (where R is a finite ring with unity and each $f_i(1)$ is a unit) if and only if*

1. *$T(O) = O$;*
2. *T is continuous;*
3. *$T(e^k)$ has finite support for each pixel configuration e^k; and*
4. *T and $\{f_i\}$ are related, for all x_j in R, by*

$$T\left(\sum_j x_j e^j\right) = \sum_j \frac{f_j(x_j)}{f_j(1)} T(e^j)_i$$

Here e^k denotes the pixel configuration with activation 1 at the kth cell and quiescent elsewhere.

While the net-input space of a neural network and its associated global dynamics are technically easier to handle, the activation space of the network and its associated dynamics are more natural objects of study. One normally thinks of the "current configuration" of a neural network as its vector of activation values at the given time. The following theorem characterizes the realizability of self-maps of the Cantor set as activation global dynamics of a neural network. Again R is a finite ring with unity.

Theorem 3.2. *A self-map $T : \mathcal{C} \to \mathcal{C}$ is realizable over a finite ring R with unity as an activation global dynamics of a neural network if and only if*

1. *$T(O) = O$;*
2. *T is continuous;*
3. *$T(e^k)$ has finite support for each pixel configuration e^k; and*
4. *$T = F \circ L$, where L is a linear self-map of $\mathcal{C}$ and F is strictly local.*

Here, F is strictly local if $F(x)_i = F(x_i e^i)_i$ for all x and i.

In this case, we say that such a self-map T is *implemented* by or *realized on* the corresponding network. Note that condition 4 holds with linear activation map F if and only if T is linear.

In each of these two theorems, condition 1 disallows spontaneous generation within the network. Condition 2 allows the recovery of the underlying network structure. Condition 3 reflects the local finiteness of the network. Condition 4 mirrors the local dynamics of the network. Thus, as one would

expect, the first two conditions are two of those from Richardson's theorem, while the other two result from the less regular type of architecture of the network and from the characteristic form of its local dynamics. (It follows from conditions 1 and 2 that the sum in the right-hand side of condition 4 in theorem 3.1 is finite.)

Theorem 3.2 makes it easy to find common self-maps of the Cantor set that are not neurally computable. The mapping defined by $T(x) = 1 - x$ reflects the Cantor set about $x = 1/2$. It fails to satisfy condition 1. The mapping defined by $T(x) = 3x \bmod 1$ simply shifts left the digits in the ternary representation of x (except for $x = 1/2, 2/3, 1$), then dropping the left-most.[1] Since T is not continuous at $x = 1/3$, it fails to satisfy condition 2. Finally, the self-map that converts from ternary to binary representation of points of the Cantor set fails to satisfy condition 3 since $1/3$ is a ternary pixel $1000\ldots$ with an infinite binary representation $010101\ldots$.

A characterization of global dynamics of automata networks is given next. Here we let $\mathcal{C} = \prod_{i \in V} Q_i$ be an encoding of the Cantor set. Each Q_i is a finite set of states contained in a common finite set Q and V is a countably infinite index set.

Theorem 3.3. *A self-map $T : \mathcal{C} \to \mathcal{C}$ is realizable as the global dynamics of an automata network if and only if*

1. *$T(O) = O$;*
2. *T is continuous;*
3. *each $j \in V$ T-influences only finitely many $i \in V$.*

Here j T-influences i (for $i, j \in V$) if for some $x \in \mathcal{C}$ there is some $y \in \mathcal{C}$ so that $y_k = x_k$ for all k different from j and $T(y)_i \neq T(x)_i$.

4. Proofs

Proof of theorem 3.3. It is easy to see that each of the three conditions is necessary. The first is immediate since each δ_i sends quiescent input to 0. The continuity of T follows from that of each $\pi_i \circ T = \delta_i$ since T is the product of the δ_i. (δ_i can be considered to be a function on the larger domain $\mathcal{C}$ by computing its value at an arbitrary $x \in \mathcal{C}$ by extending x to be quiescent elsewhere.) For the third condition note that unless j is a vertex indexing the domain of δ_i it cannot T-influence i. Since j has finite out-degree, it can contribute to the definition of only finitely many such domains.

For the sufficiency suppose that $T : \mathcal{C} \to \mathcal{C}$ satisfies the three conditions under the encoding $\mathcal{C} = \prod_{i \in V} Q_i$ as above. Let V be the vertex set and Q_i the state set of the ith finite-state machine.

[1] Recall that the points in the Cantor set are precisely those real numbers is the unit interval that admit a ternary representation containing no 2s.

The continuity of T will allow the recovery of the links between the vertices in V, completing the underlying digraph. For each vertex $i \in V$ and for each state $q \in Q_i$, let $X^{i,q} = \{x \in \mathcal{C} | x_i = q\}$. The set $X^{i,q}$ is both open and closed in the product topology. Since T is continuous $T^{-1}(X^{i,q})$ is open and hence the union of basic open sets of $\mathcal{C}$, i.e., sets that restrict only finitely many components. Since $\mathcal{C}$ is compact and $T^{-1}(X^{i,q})$ is closed, it is the union of finitely many such basic open sets. Let $V^{q,i}$ be the set of vertices indexing the restricted components. Assume also that each restricted coordinate is necessary in the sense that making it unrestricted would drive its basic neighborhood outside of $T^{-1}(X^{i,q})$. Let $V^i = \cup\{V^{q,i} | q \in Q_i\}$. For each $i \in V$, V^i is a finite set. For each $j \in V^i$ include a link form j to i in the underlying digraph. Thus each $i \in V$ has finite in-degree.

If j T-influences i, there is an $x \in \mathcal{C}$ satisfying the defining condition above. If $T(x)_i = q$, then $x \in T^{-1}(X^{i,q})$, and thus x belongs to some one of the finitely many basic neighborhoods covering $T^{-1}(X^{i,q})$. If none of these basic neighborhoods restrict j, the condition would be contradicted. Thus there is a link from j to i. Likewise, if there is such a link, $j \in V^{q,i}$ and thus is restricted in some basic neighborhood $B \subseteq T^{-1}(X^{i,q})$. Any $x \in B$ will serve to show that j T-influences i. Thus j T-influences i just in case there is a link from j to i. The third condition of the theorem now implies that each vertex has finite out-degree, and the underlying digraph is locally finite.

Take V^i to define the domain of δ_i. Given any input states to i extend them with zeros to a configuration x and define the value of δ_i to be $T(x)_i$. It is easy to see that the choice of x does not affect the value. This defines, for each i, a finite-state machine in such a way as to yield T as global dynamics of the network.

Proof of theorem 3.2. We first show the necessity of the four conditions. The first three follow immediately from theorem 3.3 since every neural network is an automata network. For the fourth condition, let $L = net$. L is clearly linear and $T = F \circ L$, where F is the product of the activation functions as defined above. One readily checks that F is strictly local.

For the sufficiency, suppose that the four conditions are satisfied. Choose a countably infinite set V to serve as the vertices of the underlying digraph. The continuity of T along with $T(O) = O$ provides the links in the digraph as in [2]. Links constructed in this way yield a graph of finite in-degree. Condition 3 implies finite out-degree, and thus the digraph is locally finite. To completely specify the neural network, it remains to determine the weights on the links and the activation functions. F will supply the activation functions and L the weights. Define the weight on the link from vertex j to vertex i by $w_{ij} := L(e^j)_i$. Finally, define the activation function for each vertex i by $f_i(r) := F(re^i)_i$. Clearly $f_i(0) = 0$ by conditions 1 and 4, so that each f_i is an activation function. It remains only to be shown that T is the global dynamics of this neural network. The following calculation uses successively

the definitions of w_{ij} and f_i, the linearity of $\pi_i \circ L$, and the strict locality of F.

$$\begin{aligned} f_i\left(\sum_j w_{ij}x_j\right) &= f_i\left[\sum_j x_j(\pi_i \circ L(e^j))\right] \\ &= f_i \circ \pi_i \circ L\left[\sum_j x_j e^j\right] \\ &= \pi_i \circ F\left[\left(L\left(\sum_j x_j e^j\right)\right)_i e^i\right] \\ &= \left[F \circ L\left(\sum_j x_j e^j\right)\right]_i \\ &= T\left(\sum_j x_j e^j\right)_i \qquad \blacksquare \end{aligned}$$

5. Open problems

Clearly few neural networks are cellular automata, although both are particular cases of automata networks. Viewing the dynamics of any of these three types of networks as a self-map of the Cantor set makes possible the comparison of their computational power. It is relatively easy to show that neural networks are at least as powerful as cellular automata [4]. The most interesting problem thus becomes *whether or not these three computational models are equivalent* in the sense that they realize, or compute, the same self-maps of the Cantor set.

A related problem that seems to be of fundamental importance in both computer science and cognitive science is the *encoding problem*. Intuitively, the images of a pattern of activation under various bijective (or just injective) neurally computable maps are just encodings of the given pattern. A first step in characterizing just what tasks are computable by neural networks would be to determine equivalent reformulations (encodings) of the given task. In the view taken here, this problem precisely asks for *a characterization of the bijective neurally computable self-maps of the Cantor set.*

A more general problem concerns a *characterization of injective global dynamics and their relation to the class of surjective dynamics.* It is well known that for a cellular automaton, injectivity of its global dynamics automatically implies surjectivity [8].

The characterizations given above of the neurally computable self-maps of the Cantor set are strongly dependent on the encoding of $\mathcal{C}$. It would be of interest to *find an encoding-independent characterization of self-maps of the Cantor set.*

Finally, the examples following theorem 3.2 suggest the following problem. Consider self-maps f of the full real interval $[0,1]$. Represent points in $[0,1]$ by binary expansions, and regard them as ternary representations of elements

of the Cantor set. Thus f can be considered as a self-map of the Cantor set. Under this encoding, *what exactly are the neurally computable functions on the unit interval?*

Acknowledgments

This work was partially supported by NSF Grant DCR 8602319.

References

[1] S.P. Franklin and M. Garzon, "Neural computability," in *Progress in Neural Networks*, Vol. 1, Ch. 6, O.M. Omidvar, ed. (Ablex, Norwood, NJ, 1990).

[2] S.P. Franklin and M. Garzon, "Global dynamics in neural networks," *Complex Systems*, **3** (1989) 29–36.

[3] F. Fogelman-Soulie, Y. Robert, and M. Tchuente, *Automata Networks in Computer Science* (Manchester University Press, 1987).

[4] M. Garzon and S.P. Franklin, "Neural computability II," *Proc. Int. Joint Conf. on Neural Networks*, Washington, D.C., Vol. I (1989) 631–637.

[5] G.A. Hedlund, "Endomorphism and automorphism of the shift dynamical system," *Math. Syst. Theory*, **3** (1969) 320–375.

[6] J.G. Hocking and G.S. Young, *Topology* (Addison-Wesley, Boston, 1969).

[7] C. Kemke, "Modelling neural networks by means of networks of finite automata," *Proc. IEEE First Intl. Conf. Neural Networks*, **III** (1987) 23–29.

[8] D. Richardson, "Tessellation with local transformations," *J. Comput. Syst. Sci.*, **6** (1972) 373–388.

[9] D.E. Rumelhart, J.L. McClelland, et al. (eds.), *Parallel Distributed Processing*, Vol. 1 (MIT Press, 1986).

[10] T. Toffoli and N. Margolus, *Cellular Automata Machines* (MIT Press, 1987).

[11] J. von Neumann, *Theory of Self-Reproducing Automata* (University of Illinois Press, Chicago, 1966).

[12] S. Wolfram, *Theory and Applications of Cellular Automata* (World Scientific, Singapore, 1986).