INTERNATIONAL JOURNAL OF CIRCUIT THEORY AND APPLICATIONS
Int. J. Circ. Theor. Appl., **26**, 365–373 (1998)

A RIGOROUS DESIGN METHOD FOR BINARY CELLULAR NEURAL NETWORKS[†]

IZTOK FAJFAR*, FRANC BRATKOVIČ, TADEJ TUMA AND JANEZ PUHAN

Faculty of Electrical Engineering, University of Ljubljana, Tržaška 25, 1000 Ljubljana, Slovenia

SUMMARY

In order to be able to take full advantage of the great application potential that lies in cellular neural networks (CNNs) we need to have successful design and learning techniques as well. In almost any analogic CNN algorithm that performs an image processing task, binary CNNs play an important role. We observed that all binary CNNs reported in the literature, except for a connected component detector, exhibit monotonic dynamics. In the paper we show that the local stability of a monotonic binary CNN represents sufficient condition for its functionality, i.e. convergence of all initial states to the prescribed global stable equilibria. Based on this finding, we propose a rigorous design method, which results in a set of design constraints in the form of linear inequalities. These are obtained from simple local rules similar to that in elementary cellular automata without having to worry about continuous dynamics of a CNN. In the end we utilize our method to design a new CNN template for detecting holes in a 2D object. © 1998 John Wiley & Sons, Ltd.

KEY WORDS: monotonic binary CNN; linear inequalities; design method

1. INTRODUCTION

Cellular neural networks (CNNs) are non-linear, mainly locally connected, arrays of analog processors placed on a multidimensional grid. Since their advent in the autumn of 1988,[1, 2] various generalizations of the CNN paradigm have been introduced, the most significant of which is a CNN Universal Machine,[3] the first algorithmically programmable analog array computer on a single chip. With such a chip in hand, we hold a great application potential but, to take full advantage of its capabilities, we need to have successful design and learning strategies as well (see Reference 4 for an exhaustive overview of the existing design methods and learning algorithms).

Design by analogy to well-known image processing algorithms was the first approach to produce some useful networks. Amongst systematic approaches the simplest ones just try to set up a certain pattern of stable and unstable equilibria for a particular application. Apart from some optional hints on shaping their basins of attraction, those techniques do not guarantee that the network will be functional, i.e. that all its initial states will converge to the prescribed global stable equilibria (see, for example, References 5 and 6). Gradient-based methods[7] fix this problem by taking into account initial states as well, or even learning prescribed trajectories. One problem with those, however, is that they are prone to get stuck in local minima. For global learning, simulated annealing has turned out to be the most robust tool[4] for learning templates for quite a wide spectrum of CNNs including non-linear and delay-type templates, and good solutions have been found even in difficult cases. Simulated annealing algorithms, however, are too expensive, in terms of computational requirements, for some simple applications that lend themselves to an analytic approach.[8] But even the approach in Reference 8

* Correspondence to: Iztok Fajfar, Faculty of Electrical Engineering, University of Ljubljana, Tržaška 25, 1000 Ljubljana, Slovenia. Email: iztok.fajfar@fe.uni-lj.si

[†] Part of this work has been reported in the Proceedings of the 1996 International Workshop on Cellular Neural Networks and their Applications held in Seville[16]

CCC 0098–9886/98/040365–09$17.50
© 1998 John Wiley & Sons, Ltd.

Received 20 May 1996
Revised 22 December1997

is quite complicated since it involves estimating boundaries of continuous trajectories, and therefore requires some knowledge about continuous time evolution of the network.

In image processing tasks, binary (see Definition 2 in the next section) CNNs play an important role as instructions in analogic CNN algorithms.[9–12] The crucial observation that the dynamics of almost all binary CNNs reported in the literature (connected component detector is the only known exception) is monotonic allowed us to construct a remarkably simple, and exact, design strategy for this class of CNNs. The method is much similar to that of setting up local rules in an elementary cellular automaton consisting of a 2D grid of sites with values 0 or 1.

Throughout the paper we focus on a single cell located in ith row and jth column of a two-dimensional space-invariant CNN, whose dynamics is governed by the following equations:

$$\dot{x}_{ij} = -x_{ij} + \sum_{C_{kl}\in\mathcal{N}_{ij}^{\rho}} A_{k-i,l-j}\, y_{kl} + \sum_{C_{kl}\in\mathcal{N}_{ij}^{\rho}} B_{k-i,l-j}\, u_{kl} + I \tag{1}$$

$$y_{ij} = f(x_{ij}) = 1/2(|x_{ij}+1| - |x_{ij}-1|) \tag{2}$$

where x_{ij}, u_{ij}, and y_{ij} denote the state, input, and output of cell C_{ij}, respectively, I is a constant current offset, and ρ, the size of the neighbourhood, is some non-negative integer. $A_{k-i,l-j}$ and $B_{k-i,l-j}$ are constant feedback and feedforward template elements connecting cell C_{ij} to C_{kl}. Furthermore, we assume u_{ij} to be constant and allow $B_{k-i,l-j}$ to be non-linear.

The *neighbourhood* of the cell C_{ij} is the set of its nearest cells and is defined as

$$\mathcal{N}_{ij}^{\rho} = \{C_{kl}\colon \max(|k-i|, |l-j| \leqslant \rho)\}. \tag{3}$$

In the next section we first introduce some definitions and notations that will be useful in later sections. In Section 3 some important results on monotonic CNNs are given, and in Section 4 we propose a formal design strategy based on these results. Finally, in Section 5, some design examples show how the method works in practice.

2. PRELIMINARY DEFINITIONS AND NOTATIONS

Before we move to the main results let us first introduce some definitions and notations.

For the purpose of this paper it will be convenient to pack all the outputs of the cells in the neighbourhood $\mathcal{N}_{ij}^{\rho}$ in a $(2\rho+1)^2$-dimensional (local) vector $\mathbf{y}_{ij}$, and all the constant terms in a single constant $K_{ij} = \sum_{C_{kl}\in\mathcal{N}_{ij}^{\rho}} B_{k-i,l-j}u_{kl} + I$. Then (1) assumes the form

$$\dot{x}_{ij} = -x_{ij} + \mathbf{a}^{\mathrm{T}}\mathbf{y}_{ij} + K_{ij} \tag{4}$$

where the vector $\mathbf{a}$ represents a feedback template in a form of a vector instead of the more usual matrix form.

Sometimes it will be handy to pack all the external inputs to a cell, and all the constant terms, in a single time-dependent function

$$g_{ij}(t) = \sum_{C_{kl}\in\mathcal{N}_{ij}^{\rho}\setminus C_{ij}} A_{k-i,l-j}\, y_{kl} + K_{ij} \tag{5}$$

The expression $C_{kl} \in \mathcal{N}_{ij}^{\rho} \setminus C_{ij}$ stands for all the cells in the neighbourhood $\mathcal{N}_{ij}^{\rho}$ of the cell C_{ij}, except the cell C_{ij} itself. This time (1) becomes

$$\dot{x}_{ij} = -x_{ij} + A_{0,0}\, y_{ij} + g_{ij}(t) \tag{6}$$

Definition 1. According to the three-segment piecewise-linear output function (2) we will distinguish three regions of operation of a cell; *linear region*, $\mathscr{L}$, where $|x_{ij}| < 1$ and $\dot{y}_{ij} = \dot{x}_{ij}$, *upper saturation region*, $\mathscr{S}^{\mathrm{u}}$, where $x_{ij} \geqslant 1$ and $\dot{y}_{ij} = 0$, and *lower saturation region*, $\mathscr{S}^{\ell}$, where $x_{ij} \leqslant -1$ and $\dot{y}_{ij} = 0$.

Definition 2. A CNN is called a *binary CNN* if all $x_{ij}(0), y_{ij}(t \to \infty) \in \{-1, 1\}$.

Remark. It was shown in Reference 1 that $A_{0,0} > 1$ represents a sufficient condition for all the stable equilibria to lie in a saturation region, which implies binary outputs at $t \to \infty$.

Definition 3. We say that cell C_{ij} of a binary CNN is *locally stable* at time τ iff[13]

$$\operatorname{sgn}((A_{0,0} - 1)y_{ij}(\tau) + g_{ij}(\tau)) = y_{ij}(\tau) \tag{7}$$

If the cell is not locally stable, then it is *locally unstable*.

Remark. The above definition requires that, if a cell is in one of its two saturation regions, then the sign of the time derivative of the state variable on the boundary of that region, $\dot{x}_{ij}(\tau)|_{x_{ij}=y_{ij}(\tau)}$, should be equal to the sign of $y_{ij}(\tau)$. This implies that if g_{ij} does not change, then the cell stays in the same saturation region forever.

Definition 4. Let us delete from the unit matrix $\mathbf{U}$ all the rows that lie at the positions of zeros in $\mathbf{a}$ and denote the obtained matrix by $\mathbf{U}^*$, then compute a vector $\boldsymbol{\eta}_{ij} = \mathbf{U}^*\mathbf{y}_{ij}$. A vector $\boldsymbol{\eta}_{ij}$ will be called a *local pattern* if $\boldsymbol{\eta}_{ij} \in \{-1, 1\}^n$, where n is the number of non-zero elements in $\mathbf{a}$.

A local pattern $\boldsymbol{\eta}_{ij}$ will be called stable if the cell C_{ij} is locally stable. Otherwise it will be called unstable. The stable and unstable local patterns are the elements of the *stable local pattern set* $\mathscr{P}^{\mathrm{s}} \subseteq \{-1, 1\}^n$ and the *unstable local pattern set* $\mathscr{P}^{\mathrm{u}} \subseteq \{-1, 1\}^n$, respectively, which are two disjoint sets of local patterns: $\mathscr{P}^{\mathrm{s}} \cap \mathscr{P}^{\mathrm{u}} = \{\ \}$.

Definition 5. Let $\mathscr{M}$ be a set of initial local patterns $\boldsymbol{\eta}_{ij}(0)$ of a binary CNN for which $\dot{y}_{i,j} \leqslant 0$ (resp. $\geqslant 0$) for all i and j, and $t \geqslant 0$. If $\mathscr{M} \neq \{\}$, then the network is said to be non-increasing (resp. non-decreasing) in $\mathscr{M}$, or simply monotonic.

Remark. For practical reasons $\mathscr{M}$ will consist only of those local patterns that can appear as a part of any possible initial state in a particular application.

Note that Definition 5 does not require dynamics to be monotonic everywhere in a state space but only in linear region. In general, a monotonic CNN can exhibit an arbitrary dynamics in saturation regions (e.g. shadow detector or hole filler).

3. MONOTONIC CNNs

A class of monotonic CNNs defined in the previous section has a property that all cells can change their output values only in a single direction. They can change their value either only from black to white (i.e. from 1 to -1) or only from white to black. The purpose of this section is to state a simple condition under which one can easily find out whether network dynamics will be monotonic or not, and to prove that for such networks stability of local patterns is a sufficient condition for the network to be functional. By *functional* we mean that the network is guaranteed to lead any possible initial state to the prescribed stable equilibrium.

Remark. In this and the next section we will limit ourselves to non-increasing CNNs, and everything we will say for them applies for non-decreasing networks as well, which can be seen by simply changing some signs and inequalities to their opposites.

Theorem 1. If all the off-centre feedback elements of a binary CNN are non-negative, $A_{0,0} > 1$, *and* $\mathscr{M}$ *is such that for all* $\boldsymbol{\eta}_{ij}(0) \in \mathscr{M}$ *the corresponding derivatives* $\dot{y}_{ij}(0^+) \leqslant 0$,† *then the CNN is non-increasing in* $\mathscr{M}$ *(or monotonic).*

Proof. Consider a non-saturated cell C_{ij} and a derivative of its output $\dot{y}_{ij} = (\mathrm{d}y_{ij}/\mathrm{d}x_{ij})\dot{x}_{ij} = (A_{0,0} - 1)x_{ij}(t) + g_{ij}(t)$. Suppose that at some time t_m all the derivatives $\dot{y}_{kl \neq ij} \leqslant 0$, and $\dot{x}_{ij} < 0$. Because $A_{0,0} > 1$, and since all the off-centre feedback coefficients are non-negative, the signs of the derivatives remains the same as long as no cell changes its operation from saturated to linear region, or *vice versa*.

Suppose that at time t_{m+1} one or more cells change its operation mode, and we only have to prove that no cell can switch from $\mathscr{S}^\ell$ to $\mathscr{L}$, in which case the cell would produce a positive output derivative. Since for $t_m \leqslant t < t_{m+1} g_{ij}(t)$ is a non-increasing continuous function of time (cf. equation (5)) we have $g_{ij}(t_{m+1}) \leqslant g_{ij}(t_m)$. Hence, the derivative at the lower boundary of $\mathscr{L}$ is $\dot{y}_{ij}(t_{m+1})|_{x_{ij}=-1^+} = \dot{x}_{ij}(t_{m+1})|_{x_{ij}=-1^+} = (-1^+)(A_{0,0} - 1) + g_{ij}(t_{m+1}) \leqslant (-1^+)(A_{0,0} - 1) + g_{ij}(t_m) = \dot{x}_{ij}(t_m)|_{x_{ij}=-1^+} = \dot{y}_{ij}(t_m)|_{x_{ij}=-1^+} \leqslant 0$.‡

We just showed that at the time t_{m+1} no output derivative $\dot{y}_{ij} > 0$ can emerge, hence, by applying the proof recursively with $t_0 = 0^+$ and $m = 0, \ldots, k$, $y_{ij}(t_{k+1}) = y_{ij}(t \to \infty)$ for all i and j, we prove Theorem 1. □

Corollary 1. If the stability of initial local patterns $\boldsymbol{\eta}_{ij}(0) \in \mathscr{M}$ *of monotonic binary CNN is preserved as one varies the network parameters, and the off-centre feedback elements remain non-negative, then the CNN remains monotonic.*

Proof. If the stability of initial local patterns is preserved, then no new unstable initial local pattern can emerge. That implies that the signs of all $\dot{y}_{ij}$ at time $t_0 = 0$ for every possible $\boldsymbol{\eta}_{ij}(0) \in \mathscr{M}$ remain non-positive. □

Before we can develop a formal design strategy for monotonic binary CNNs we need to prove one more theorem.

Theorem 2. If one varies the parameters of a monotonic binary CNN in a way that all the off-centre feedback elements remain non-negative, and the local pattern stability does not change, except possibly for the unreachable local patterns, then the functionality of the network remains unchanged.

Remark. Unreachable local patterns are those that, even if they are stable, do not attract any of the initial local patterns. Their stability is therefore of no practical importance.

Proof. Consider first all the locally unstable cells at time $t_0 = 0$. By Corollary 1 they will all sooner or later enter their lower saturation regions and stay there forever, even though the parameter values are changed. During the time of their transition they can cause certain other neighbouring cells to become locally unstable, and these cells will also start descending towards $\mathscr{S}^\ell$, possibly leading to instability of some further cells.

To prove that this evolution always ends in the same global stable equilibrium let us assume that an arbitrary cell, say cell C_{ij}, fails to change its operation from $\mathscr{S}^{\mathrm{u}}$ to $\mathscr{S}^\ell$ because of the alternation of network parameters. This means that at least one of the cell's neighbours has also failed to do so, because if it had not, then C_{ij} could not have stayed in $\mathscr{S}^{\mathrm{u}}$ unless we have created a new stable local pattern. But then also for this second cell, for the same reason, there must exist another cell which has also failed to leave $\mathscr{S}^{\mathrm{u}}$.

† At time $t = 0$ all the derivatives $\dot{y}_{ij} = 0$ (cf. Definitions 1 and 2)

‡ Notation -1^+ stands for a value infinitesimally more than -1

Because the size of the network is finite, applying this reasoning recursively, we must come to a cell which should have left $\mathscr{S}^{\mathrm{u}}$ at time t_0 but failed to do so, which again contradicts our initial assumption that no new stable local pattern was created.

In a similar way we can prove that no cell can fail to stay in $\mathscr{S}^{\mathrm{u}}$ provided that no new unstable local pattern was created. □

4. LOCAL RULES AND CNN DESIGN

We are now able to develop a formal design strategy based on the so-called *local rules*. Theorem 2 allows us to eliminate analogue dynamics and, provided the stability of reachable local patterns is preserved, the following discrete time evolution of a cell leads to the same CNN functionality as equation (6):

$$y_{ij}(t+1) = \mathrm{sgn}((A_{0,0}-1)y_{ij}(t) + g_{ij}(t)) = \phi(\boldsymbol{\eta}_{ij}(t)) \tag{8}$$

Now the output of the cell C_{ij} is only the function of the local pattern in the previous time step, and the only constraint imposed on $\phi(\cdot)$ is that $y_{ij}(t+1) \leqslant y_{ij}(t)$ for every reachable local pattern.

In order to design a particular application one only needs to define a function $\phi(\cdot)$ as a set of local rules, i.e. for each of the reachable local patterns one must decide upon the value of y_{ij} in the next time step.

When once $\phi(\cdot)$ has been set up the next step is to determine the stability of the local patterns, and the following are the local rules translated into stability conditions:

1. If $\phi(\boldsymbol{\eta}_{ij}(t)) = y_{ij}(t)$, then $\boldsymbol{\eta}_{ij}(t)$ should be a stable local pattern.
2. If $\phi(\boldsymbol{\eta}_{ij}(t)) < y_{ij}(t)$, then $\boldsymbol{\eta}_{ij}(t)$ should be an unstable local pattern.

Having established the stability of local patterns we set up the set of linear inequalities for stable patterns

$$\boldsymbol{\eta}_{ij} \in \mathscr{P}^{\mathrm{s}}, \quad y_{ij}((\mathbf{U}^{*}\mathbf{a})\boldsymbol{\eta}_{ij} + K_{ij}) > 1 \tag{9}$$

and for unstable patterns

$$\boldsymbol{\eta}_{ij} \in \mathscr{P}^{\mathrm{u}}, \quad y_{ij}((\mathbf{U}^{*}\mathbf{a})\boldsymbol{\eta}_{ij} + K_{ij}) < 1 \tag{10}$$

Inequalities (9) and (10) are derived from (7), and can be solved using one of the methods in References 14 or 6.

It should be noted that if one wants the network obtained by the above procedure operate properly, then it is important that appropriate boundary conditions are introduced in the form of dummy border cells. These are usually set to the background colour of the processed image. It can be shown that such boundary conditions often inflate the area bounded by inequalities (9) and (10), and can therefore increase the robustness of the network. This is due to the fact that zero boundary conditions—which are tantamount to absence of border cells—often impose some additional stability conditions (9) and (10).

5. SOME EXAMPLES

5.1. *Logical or*

As the first example we show how a multiple input logical or can be designed using the proposed method. The rules that cell C_{ij} must obey are very simple:

1. If a cell is white, then, if all its neighbours are also white, it must stay white, otherwise it should become black.
2. If a cell is black, then it should remain black.

From these two rules it is obvious that the network behaviour will be monotonic, and that, in order that the network will work properly, all the boundary cells must be fixed to the value -1.

A CNN defined by the following template could perform the task:

$$\mathbf{A} = \begin{bmatrix} 0 & A & 0 \\ A & A_{0,0} & A \\ 0 & A & 0 \end{bmatrix}, \quad I. \tag{11}$$

From the above two rules and template (11) we arrive to the following stable and unstable local pattern sets:

$$\mathscr{P}^{\mathrm{s}} = \{(-1,-1,-1,-1,-1)^{\mathrm{T}}, (x,x,1,x,x)^{\mathrm{T}}\} \tag{12}$$

$$\mathscr{P}^{\mathrm{u}} = \{-1,1\}^5 \setminus \mathscr{P}^{\mathrm{s}} \tag{13}$$

where the symbol x stands for a don't-care value, i.e. either 1 or -1. The middle element in a pattern vector corresponds to the cell C_{ij}.

From the stability of the local patterns (12) and (13), and from inequalities (9) and (10), we obtain the following design constraints:

$$-(-A_{0,0} - 4A + I) > 1 \tag{14}$$

$$A_{0,0} - 4A + I > 1 \tag{15}$$

$$-(-A_{0,0} - 2A + I) < 1 \tag{16}$$

Note that constraints derived from most local patterns are surplus and have been therefore omitted. Using the method in Reference 6, while limiting absolute values of all the parameters to 10, the solution is $A_{0,0} = 4$, $A = 2.5$, and $I = 10$.

5.2. *Hole-Filler*

Next we try to design a hole-filling CNN, which can be defined by a template in the form

$$\mathbf{A} = \begin{bmatrix} 0 & A & 0 \\ A & A_{0,0} & A \\ 0 & A & 0 \end{bmatrix}, \quad B \tag{17}$$

The picture to be processed is fed to the input while the initial state values are set to 1.

The rules that cell C_{ij} must follow are:

1. If $u_{ij} = 1$ then the cell must stay black forever.
2. If $u_{ij} = -1$ then the cell stays black only if all the four of its neighbours also stay black. Otherwise it should switch to white.

We must construct stable and unstable local pattern sets for each of the two possible input values u_{ij} separately. For $u_{ij} = 1$ we infer from the first rule that the stable pattern set is

$$\mathscr{P}^{\mathrm{s}} = \{(x,x,1,x,x)^{\mathrm{T}}\} \tag{18}$$

Because all the cells are initially black all the other local patterns are clearly unreachable and we do not care about their stability.

From the second rule we have that for $u_{ij} = -1$,

$$\mathscr{P}^{\mathrm{s}} = \{(1,1,1,1,1)^{\mathrm{T}}, (x,x,-1,x,x)^{\mathrm{T}}\} \setminus \{(1,1,-1,1,1)^{\mathrm{T}}\} \tag{19}$$

$$\mathscr{P}^{\mathrm{u}} = \{(x,x,1,x,x)^{\mathrm{T}}\} \setminus \{(1,1,1,1,1)^{\mathrm{T}}\} \tag{20}$$

In this case the pattern $(1,1,-1,1,1)^{\mathrm{T}}$ is unreachable.

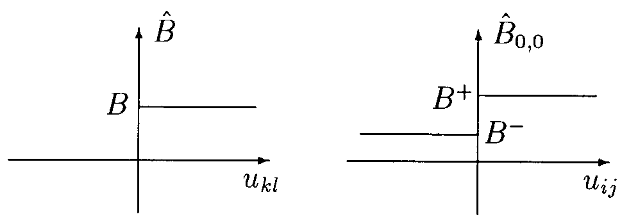

Figure 1. Non-linear feedforward template elements of hole-detecting CNN

From (18) we get a design constraint

$$A_{0,0} - 4A + B > 1 \tag{21}$$

while (19) and (20) yield

$$A_{0,0} + 4A - B > 1 \tag{22}$$

$$A_{0,0} + 2A - B < 1 \tag{23}$$

from which we get a solution $A_{0,0} = 4$, $A = 2{\cdot}5$, and $B = 10$.

5.3. *Hole-detecting CNN*

A step in recognizing certain objects is to detect and count the number of holes they contain. One way of detecting holes is first to fill and then extract them by means of logical operation between original image and the one with filled holes. In this section we show that it is possible to design a CNN that performs a hole-detecting operation in a single transient.

Again, all the initial states are set to 1, and cell C_{ij} of a hole-detecting CNN must obey the following rules:

1. If $u_{ij} = 1$ then the cell must change to white.
2. If $u_{ij} = -1$ then the cell changes to white if and only if the number of black outputs of the neighbouring cells, y_{kl}, plus the number of black inputs to the neighbouring cells, u_{kl}, is smaller than four.

It is evident that in order to fulfill the second rule we have to introduce some off-centre feedforward elements, so that the template for this kind of task will have the following form:

$$\mathbf{A} = \begin{bmatrix} 0 & A & 0 \\ A & A_{0,0} & A \\ 0 & A & 0 \end{bmatrix}, \qquad \mathbf{B} = \begin{bmatrix} 0 & \hat{B} & 0 \\ \hat{B} & \hat{B}_{0,0} & \hat{B} \\ 0 & \hat{B} & 0 \end{bmatrix} \tag{24}$$

where the feedforward template elements $\hat{B}_{0,0}$ and $\hat{B}$ are non-linear, and are shown in Figure 1.

The purpose of the off-centre elements $\hat{B}$ is to compensate for negative outputs $y_{kl} = -1$ at the locations where $u_{kl} = 1$. Therefore, they should be zero for $u_{kl} = -1$ and positive for $u_{kl} = 1$. The element $\hat{B}_{0,0}$ must be non-linear, or the design constraints for this problem cannot be satisfied (cf. (27) and (36)).

From rule 1 we have the local pattern sets

$$\mathscr{P}^{\mathrm{u}} = \{(x, x, 1, x, x)^{\mathrm{T}}\} \tag{25}$$

$$\mathscr{P}^{\mathrm{s}} = \{(x, x, -1, x, x)^{\mathrm{T}}\} \tag{26}$$

from which we obtain

$$A_{0,0} + 4A + B^+ + 4B < 1 \tag{27}$$

In order to satisfy rule 2 we would have to construct a local pattern sets for every possible number of $u_{kl} = 1$ which is a little bit impractical, so we try to set up appropriate inequalities directly. First, let us consider a case where no neighbouring $u_{kl} = 1$. In that case a black cell is locally stable iff all the four of the neighbouring cells are black

$$A_{0,0} + 4A + B^{-} > 1 \tag{28}$$

and unstable otherwise:

$$A_{0,0} + 2A + B^{-} < 1 \tag{29}$$

Again we have one unreachable pattern $(1, 1, -1, 1, 1)^{\mathrm{T}}$, while the other cases with C_{ij} white should be locally stable but the stability condition is already satisfied by (29).

In the same way we get the inequalities for the case where exactly one of the neighbours has black input, $u_{kl} = 1$. Now all the black cells with three or more black neighbours should be locally stable, and other black cells locally unstable:

$$A_{0,0} + 2A + B^{-} + B > 1 \tag{30}$$

$$A_{0,0} + B^{-} + B < 1 \tag{31}$$

And, equivalently, for the last three cases we have

$$A_{0,0} + B^{-} + 2B > 1 \tag{32}$$

$$A_{0,0} - 2A + B^{-} + 2B < 1 \tag{33}$$

$$A_{0,0} - 2A + B^{-} + 3B > 1 \tag{34}$$

$$A_{0,0} - 4A + B^{-} + 3B < 1 \tag{35}$$

$$A_{0,0} - 4A + B^{-} + 4B > 1 \tag{36}$$

A solution to (27)–(36) is $A_{0,0} = 1{\cdot}1, A = 0{\cdot}7, B = 1{\cdot}5, B^{-} = -2{\cdot}5$, and $B^{+} = -10$.

6. CONCLUSIONS

Although the design method proposed in the paper is limited to a class of monotonic binary CNNs, we believe that it is an important contribution to the pool of existing CNN design and learning algorithms. Binary CNNs constitute a significant part of many analogic CNN algorithms, and all the so far reported binary CNNs except for a connected component detector, whose dynamics has been studied extensively in Reference 15, are monotonic.

We have shown that for this class of CNNs the stability of reachable local patterns is a sufficient condition for functionality of the network. That reduces a problem of designing such networks to a simple fixed point design, which consists of merely finding necessary *local* stability conditions for a problem in hand, and then solving a resulting set of linear inequalities. This is significant simplification over the method in Reference 8, the only other known analytical design method for CNNs, and it makes the method well suited for automated design. Apart from that, our method is also directly applicable to discrete-time CNNs.

Another advantage of the results reported in the paper is the simple condition summarized in Theorem 1, which tells us in which case CNN dynamics will be monotonic. Experiments indicate that incorporating some simple non-linear elements in such networks can speed up the overall CNN transient up to two orders of magnitude,[16] which is mainly due to the increased propagation speed obtained by those non-linearities.

ACKNOWLEDGEMENT

Authors are grateful to Dr. Péter Szolgay for his helpful comments and suggestions.

REFERENCES

1. L. O. Chua and L. Yang, 'Cellular neural networks: theory', *IEEE Trans. Circuits and Systems*, **CAS-35**, 1257–1272 (1988).
2. L. O. Chua and L. Yang, 'Cellular neural networks: applications', *IEEE Trans. Circuits Systems*, **CAS-35**, 1273–1290 (1988).
3. T. Roska and L. O. Chua, 'The CNN Universal Machine: an analogic array computer', *IEEE Trans. Circuits Systems–II*, **CAS-II-40**, 163–173 (1993).
4. J. A. Nossek, 'Design and learning with cellular neural networks', *Int. J. Circuit Theory Appl.*, **24**, 15–24 (1996).
5. F. Zou, S. Schwarz, and J. A. Nossek, 'Cellular neural network design using a learning algorithm', *Proc. 1st IEEE Int. Workshop on Cellular Neural Networks and their Applications*, Budapest, December 1990, pp. 73–81.
6. G. Seiler, A. J. Schuler and J. A. Nossek, 'Design of robust cellular neural networks', *IEEE Trans. Circuits Systems–I*, **CAS-I-40**, 358–364 (1993).
7. A. Schuler, P. Nachbar and J. A. Nossek, 'State-based backpropagation-through-time for CNNs', *Proc. 11th Eur. Conf. on Circuit Theory and Design*, Davos, August 1993, pp. 33–38.
8. L. O. Chua and P. Thiran, 'An analytic method for designing simple cellular neural networks', *IEEE Trans. Circuits Systems*, **CAS-38**, 1332–1341 (1991).
9. M. Csapodi and T. Roska, 'Dynamic analogic CNN algorithms for a complex recognition task–a first step towards a bionic eyeglass', *Int. J. Circuits Theory Appl.*, **24**, 127–144 (1996).
10. L. O. Chua, T. Roska, P. L. Venetianer, and Á. Zarándy, 'Some novel capabilities of CNN: game of life and examples of multipath algorithms', *Proc. 2nd IEEE Int. Workshop on Cellular Neural Networks and their Applications*, Munich, October 1992, pp. 276–281.
11. Á. Zarándy, F. Werblin, T. Roska, and L. O. Chua, 'Novel types of analogic CNN algorithms for recognizing bank-notes', *Proc. 3rd IEEE Int. Workshop on Cellular Neural Networks and their Applications*, Rome, December 1994, pp. 273–278.
12. T. Roska and L. Kék (eds), 'Analogic CNN program library', *Tech. Rep. DNS-5-1994*, Analogical and Neural Computing Laboratory, Computer and Automation Institute, Hungarian Academy of Sciences, Budapest, 1994.
13. I. Fajfar and F. Bratkovič, 'Finding all stable equilibria of cellular neural networks', *Int. J. Circuit Theory Appl.*, **22**, 145–155 (1994).
14. I. Fajfar and F. Bratkovič, 'Statistical design using variable parameter variances and application to cellular neural networks', *Proc. 3rd IEEE Int. Workshop on Cellular Neural Networks and their Applications*, Rome, December 1994, pp. 147-152.
15. P. P. Civalleri and M. Gilli, 'Global dynamic behaviour of a three-cell connected component detector CNN', *Int. J. Circuit Theory Appl.*, **23**, 117–135 (1995).
16. I. Fajfar and F. Bratkovič, 'Design of monotonic binary-valued cellular neural networks', *Proc. 4th IEEE Int. Workshop on Cellular Neural Networks and their Applications*, Sevilla, June 1996, pp. 321–326.