Time Out of Joint: Attractors in Asynchronous Random Boolean Networks

Inman Harvey[1] **and Terry Bossomaier**[2]
[1]Centre for Computational Neuroscience and Robotics
School of Cognitive and Computing Sciences
and School of Biological Sciences
University of Sussex, Brighton UK
[2]School of Information Technology
Charles Sturt University, Bathurst NSW, Australia
inmanh@cogs.susx.ac.uk tbossomaier@csu.edu.au

Abstract

Random Boolean networks (RBNs) are complex systems composed of many simple components which have been much analysed and shown to have many robust generic properties. Some synchronous versions have been influential as highly abstract models of specific biological systems, but for many biological phenomena asynchronous versions are more plausible models. Though asynchronous RBNs are indeterministic they can be shown to have generic properties that are simpler than, and very different from, the synchronous versions. These properties are demonstrated for the first time here through computer simulation and through analysis.

1 Introduction

Models of complex physical and biological phenomena inevitably ignore much of the detail of the real phenomena and simplify into systems with a small number of concepts. Complex behaviour can be generated from conceptually simple primitive elements if they interconnect and interact in large numbers. Cellular automata (CAs) and Random Boolean Networks (RBNs) are two such classes of models.

With such models any one out of an enormous range of possible initial states for the system can be specified, and the pathway through state space observed as the system is updated. An important result from the theory of RBNs is that under certain conditions the multiplicity of initial states converge onto a small number of attractors [8]. When using an RBN as a model of a genetic regulatory network this result has suggested that with (for instance) 100,000 binary switches, each assumed to be modelling a switching gene regulated by two others, despite the $2^{100,000}$ possible initial states of the system there will be only of the order of $\sqrt{100,000} \simeq 300$ attractors, or classes of end-states. This is seen as a reasonable correspondence with the number of genes and number of cell-types in the human body, and it is suggested that the behaviour of such RBNs underlies the relationship between these numbers.

In this paper we draw attention to the fact that these RBN models rely on synchronous updating. Such updating will give a deterministic pathway from any given initial state of the system, and this fact is used in the analysis of RBNs with such tools as DDLAB [16]. We here give a first analysis of some aspects of *asynchronous* RBNs and show that their behaviour is radically different from the synchronous versions. There cannot be cyclic attractors though point attractors are still possible. If such point attractors are present then they are usually in very small numbers, typically single figures, regardless of the size of the system.

This casts doubt on the relevance of the (synchronous) RBN results to such biological phenomena as genetic regulatory networks, unless new arguments are presented as to why such networks should update synchronously. For many physical and biological phenomena the assumption of asynchrony seems more plausible.

2 Cellular Automata and Random Boolean Networks

CAs have a regular spatial array of nodes or cells, each of which can take one of a finite set of values; and an update rule whereby the subsequent value of each cell is a function of the present values of itself and its local neighbours. From an initial state of the CA where the value of each cell is specified, the same update rule is applied at each cell simultaneously to generate the subsequent state of the CA. A sequence of such iterations gives rise to a deterministic path through the CA state

space, leading to a point attractor or a cyclic attractor in this space (for a CA with a finite number of nodes any path through the finite state space must eventually cycle).

A Boolean Network (BN) is a variation on a CA in which the property of neighbourhood between cells laid out in a spatial array is replaced by directed links between nodes regardless of spatial location. Each of N nodes can take the value 0 or 1, and when updated its new value depends on the current values of those other nodes (possibly including itself) which have links directed onto it. Typically each node has the same number K of links arriving at it, and the update rule is some Boolean function of these K inputs which can be specified with a lookup table of 2^K binary entries — there are a possible 2^{2^K} such lookup tables.

For a Random BN the links between nodes are specified at random, as are the Boolean functions applicable at each node; once specified these links and Boolean functions do not alter. We shall here allow at most one influencing link from one node to another. For each node there are $N!/(N-K)!$ possible ordered choices of K different links, so for the system as a whole there are $(N!/(N-K)!)^N$ possible arrangements of connectivity. It is necessary here to take account of the ordering because most Boolean functions will be affected differently by successive incoming links. Choosing for each of N nodes a possible lookup table gives a total of $(2^{2^K})^N$ possible combinations of N Boolean functions. Thus there are a very large number, say a *gazillion* G RBNs where

$$G = (\frac{2^{2^K} N!}{(N-K)!})^N$$

Many of these RBNs will be identical to each other subject only to relabelling of the nodes or links. The calculations above assume that the nodes/links can be labelled and distinguished from each other. If the different decision had been made to count two RBNs as one when renumbering nodes/links can make them identical, then the number of combinations would be smaller.

Despite this enormous range of possible RBNs for any given values of N and K a number of conclusions can be drawn about their generic behaviour. One interesting result from which many conclusions have been drawn is that as K decreases from N to 1, the generic behaviour of such networks undergoes a phase transition at $K = 2$ 'from chaos to order'. In [8] Kauffman draws the following conclusions for *synchronous* RBNs:

- $K = N$: there are about N/e cyclic attractors with associated basins of attraction — a relatively small number. The median expected length of state cycles is $0.5 \times 2^{N/2}$ — an exponentially large number. The system is unstable to perturbations, in that typically a single flip of a node perturbed from outside moves the whole system into a different basin.

- $K \geq 5$: there are still a modest number of chaotic attractors (in the sense of having very long state cycles susceptible to perturbations) for such values of K, and even perhaps for K as low as 3.

- $K = 2$: such systems exhibit very high order. The expected median state cycle length is about $\sqrt{N}$ — a surprisingly small number. The number of state cycle attractors is also about $\sqrt{N}$. These cycles are generally robust to perturbations, which usually fail to shift the system into a different attractor.

- $K = 1$: the network falls apart into separate loops with tails. This gives separate isolated subsystems, whose product gives the overall behaviour. Median lengths of state cycles are of order $\sqrt{\pi N/2}$, and the number of such attractors is exponential in N.

The particular properties of $K = 2$ has resulted in the suggestion [8] that these can be seen as models of genetic regulatory networks, as discussed in Section 1.

3 The Boolean idealisation

In [8] (pp. 182–183) RBNs are proposed by Kauffman as a means of constructing a statistical mechanics over ensembles of complex dynamical systems which contain a multitude of coupled variables. Such systems include genetic regulatory networks, the immune system and idiotype networks, autocatalytic polymer systems and neural networks.

In fitting such complex systems into a framework such as RBNs, simplifications and abstractions must be made; two of these are the Boolean idealisation and the Synchrony idealisation. The Boolean idealisation assumes that some important features of such systems can be captured if the coupled variables are allowed only the values 0 and 1, instead of a continuous range.

This can be justified in many circumstances. Variables representing physical quantities at a node of a system typically have a floor value of zero (or of saturation in a negative sense), and a maximum ceiling value of positive saturation. This gives some version of a sigmoid response curve of output as a function of inputs. When a number of such nodes with sigmoidal functions are coupled together, then feedback interactions result in values at each node being *quickly* pushed to a maximum or minimum [8, 14, 15].

Hence the Boolean idealisation has some justification and some plausibility in these circumstances; particularly where the time constants implied by *"quickly"* mean that periods spent in transient values are negligible compared to periods when values are at a maximum or minimum.

4 The Synchrony idealisation

The same cannot be said for the idealisation of such complex systems as synchronous. In [8] the only mention of synchrony made by Kauffman is (p. 189):

> The simplest class of Boolean networks is *synchronous*, which means that all elements update their activities at the same moment.

Later in [8] (p. 483) it is pointed out that in the synchronous RBNs modelled, attractors are stable to 'most but not all perturbations' in which the activities of any single 'gene' are altered (read: a single node has its value changed). This is used to suggest that:

> This length [of cycle] does not change dramatically for asynchronous models, as just noted.

However in this quotation the sense of 'asynchronous' appears to be that of occasional perturbations in an otherwise synchronous system. As will be shown below, when there is no synchrony the behaviour of RBNs is very different.

In [3] Boolean networks are used as models of genetic regulatory networks, with a discussion of the issues involved in making simplifying assumptions. Although their aim is to achieve "a biologically defensible model", Dellaert and Beer use synchronous updating, with this justification:

> Discrete time, synchronously updating networks are certainly not biologically defensible: in development the interactions between regulatory elements do not occur in a lock-step fashion. The alternative is to update all nodes asynchronously, each node having a given probability at any time to recompute its value from its inputs at that time. This introduces an element of nondeterminism however, that might render any genetic search in a space of such networks very difficult. In addition, they are less readily analyzed than their synchronous counterparts, for which there are excellent analysis tools available [the authors quote here (Wuensche 1994) [16]].

It appears that reasons for rejecting asynchrony have been a desire for simplicity, and worries about indeterminism. When in an asynchronous RBN (ARBN) nodes are updated in a random order this does indeed introduce indeterminism. We can thus no longer rely on the argument from deterministic pathways through state spaces with a finite number of points to deduce that an attractor (point or cyclic) must eventually be reached from any initial point. It is tempting to assume from this that in ARBNs attractors are non-existent or at any rate are. This assumption we shall show to be false.

With CAs there has been some discussion in recent years of asynchronous updating [1, 13], and a recognition that the behaviour of synchronous and asynchronous versions can vary widely. In [7] it was pointed out by Huberman and Glance that CA simulations by Nowak and May of Prisoner's Dilemma interactions based on synchronous updating [10] lead to conclusions which cannot be justified when more realistic assumptions of asynchrony are tested out. This led to a response at the Third European Conference on Artificial Life by Nowak and May [9]:

> There are, however, some situations where it may be more appropriate to work in continuous time, choosing individual sites at random, evaluating all the scores, and updating immediately. Huberman and Glance [7], indeed suggest that "if a computer simulation is to mimic a real world system ...it should contain procedures that ensure that the updating of the interacting entities is continuous and asynchronous". We strongly disagree with this extreme view, believing that discrete time is appropriate for many biological situations, and continuous time for others.

Whilst concurring with this last sentence, the view we take is that when modelling physical or biological systems of many interacting parts, asynchronous models should be the default version unless there are good reasons advanced for considering discrete time versions. Hopfield makes a similar point [6] in the context of Hopfield networks as models of brain processes, in contrast to Perceptron models:

> ...Perceptron modelling required synchronous neurons like a conventional digital computer. There is no evidence for such global synchrony and, given the delays of nerve signal propagation, there would be no way to use global synchrony effectively. Chiefly computational properties which can exist in spite of asynchrony have interesting implications in biology.

Adrian Thompson points out (personal communication) that the Synchrony idealisation is perhaps made most dangerous exactly when it is associated with the Boolean idealisation. It is a common technique when producing a computational simulation of a continuous analogue system to approximate the differential equations which govern the interdependent changes of variables by fine time-slicing. With this technique, all the variables are updated synchronously, and *in the absence of discontinuities* arbitrary precision can be achieved by making the time-slices fine enough. However the Boolean idealisation introduces just the kind of discontinuities that can make the Synchrony idealisation misleading — except in those cases where it can genuinely be shown

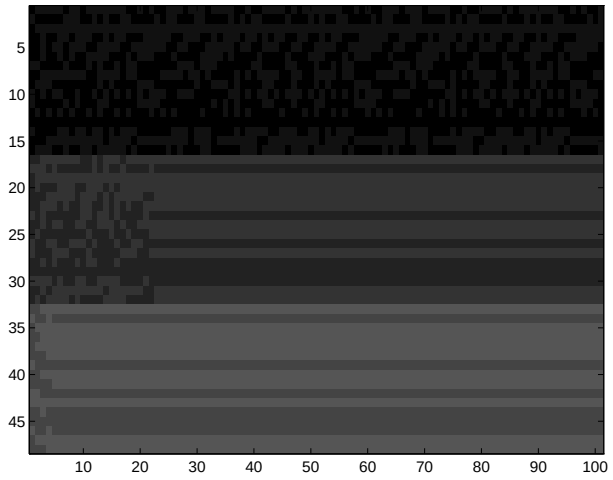

Figure 1: *Plots of the three methods of iterating an RBN ($N = K = 16$) with the same connectivity, Boolean functions and initial state. Rows 1–16, method 1; rows 17–32, method 2; rows 33–48, method 3. The same initial position is shown on the left, and the state is plotted after N updates in each case; in the first case this is N simultaneous updates. It can be seen that the synchronous version appears to have reached a cyclic attractor of period 38 after $10N$ updates, the second has converged to a point attractor by $22N$ updates, the last version reached a point attractor after just $4N$ updates. In the latter two asynchronous cases the same attractor was reached.*

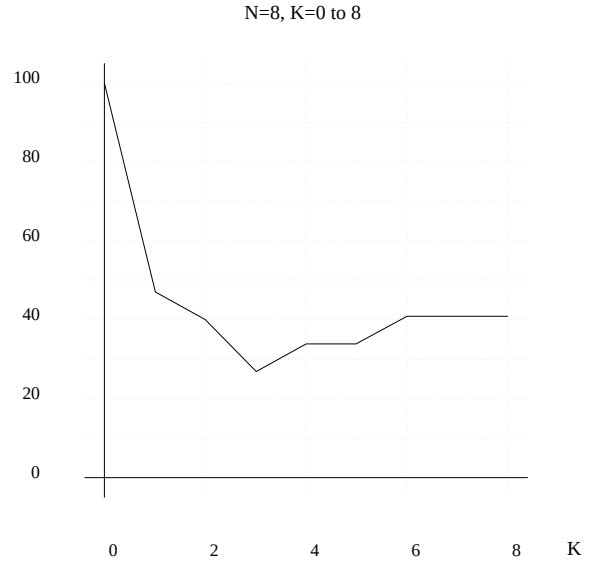

Figure 2: *Here is shown the number of runs out of 100 where a point attractor is reached within $10000N$ updates; for $N = 8$ and different values of K.*

that numerous discontinuous changes should indeed be modelled as simultaneous.

Though synchronous RBNs have been used as biological models, there seems to be a surprising lack of discussion, either of an experimental or analytic nature, of ARBNs in the literature. The remainder of this paper is a first attempt at repairing this omission.

5 Experiments with ARBNs

A number of RBNs were simulated with different values for N and K. In initial experiments the same RBN was iterated from the same arbitrary initial position using three alternative methods for updating.

1. Conventional synchronous updating of all N nodes simultaneously.

2. The N nodes were updated in a randomly ordered sequence; before each sequence a new random ordering of the N nodes was made. This implies that after each sequence of N, every node had been updated exactly once.

3. For each update a node was selected at random. in practice, a sequence of N updates was drawn with replacement from the numbers 1 to N.

By projecting the evolving patterns on the monitor as the simulation ran it was seen that frequently the second and third methods arrived fairly quickly at an unchanging pattern, a point attractor in state space; an example is given in Figure 1. This was contrary to the tempting assumption that such attractors in ARBNs were rare, and made it feasible to automate a search for them. The test for reaching such a point attractor was to check every possible update in turn, and verify that none of them changed the value of any node. It was observed that in general when the third method arrived at an attractor it did so earlier than the second method, so in further experiments only the third method was used.

In this search, for all update methods a maximum of $10000N$ node updates was made from a random initial state, equivalent to 10000 synchronous updates by the first method. For reasons of speed, checks were made at intervals to see whether any of the methods had already reached a point attractor, in which case that part of the run was discontinued. A record was made for each N, K of how often out of 100 trials a point attractor was reached within this maximum timespan, with random choices of connections, Boolean functions, and initial states.

The results shown in Figures 2 and 3 were that point attractors were frequently found for all K, but with *least* probability around $K = 3$. In order to get a picture of typical sizes of basins of attraction for a point attractor, a number of ARBNs which reached a PA were iterated

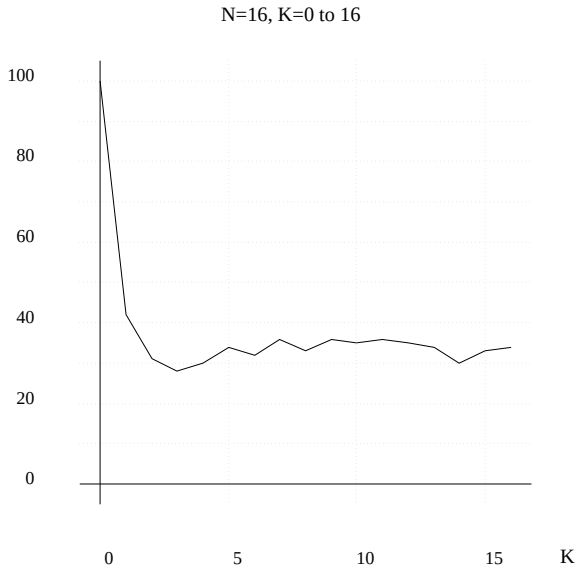

Figure 3: *As the previous figure, but for* $N = 16$ *and different values of* K.

K:	1	3	6	16
Reached PA	63	38	53	43
1 PA	42	18	27	1
2 PAs	17	11	18	10
3 PAs	1	2	5	10
4 PAs	2	2	-	1
5 PAs	-	1	-	-
6 PAs	1	-	-	-
7 PAs	-	1	-	-
1 PA or noncv	-	3	3	11
2 PAs or noncv	-	-	-	7
3 PAs or noncv	-	-	-	3

Table 1: *For* $N = 16$ *and each of the various values of* K *listed, 100 new runs were done on different ARBNs. For those ARBNS that reached a PA on the first run, a further 9 runs (making 10 in all) were done from different randomly chosen initial states. The number of different PAs reached were noted, together with those that sometimes did not reach a PA on later runs before* $10000N$ *iterations — nonconverged or 'noncv'.*

for a maximum of $10000N$ updates from a number of randomly chosen different initial states. Results are shown in Table 1 and indicate that for these values of N, when point attractors are found their basins of attraction generally include most of state space. It should be noticed that for $K \leq 6$ it is more common to have a single PA than 2 or more, but for the maximum value of K 2 or 3 PAs (or 1 PA plus failure to converge) are more common than having a single PA that is reliably reached.

This suggests that although for all K there are skewed distributions of PAs (with, we shall show later, a mean of one per ARBN), the nature of the skewed distribution varies with K. For low K there will be a small number of ARBNs with a large number of PAs, and a large number with either zero or one. For large K, when there are PAs there tend to be, in these examples, 2 or 3 of them.

6 Loose Attractors and Point Attractors in ARBNs

The indeterminacy of ARBNs means that attractors and basins of attraction can be somewhat different from their counterparts in deterministic systems. Point attractors, where the network settles into a state from which no possible update can shift it, are the only clear case where the counterparts can be considered identical.

When the network passes indefinitely through a subset of its possible states, this is generally neither strictly a cyclic attractor nor a strange attractor, in the deterministic sense of these words. We shall call $\mathcal{A}$ a *loose attractor* if for all possible states S of the network such that $S \in \mathcal{A}$ and for all possible successor states (through one or more updates) S' of S, $S' \in \mathcal{A}$. The *definite basin* $B_d(\mathcal{A})$ of such a loose attractor $\mathcal{A}$ is the set of states from which *all possible paths* lead (in the long term) into states of $\mathcal{A}$. The *possible basin* $B_p(\mathcal{A})$ is the set of states from which at least *one possible path* leads into $B_d(\mathcal{A})$. Those states in $B_p(\mathcal{A})$ which do not feature in $B_d(\mathcal{A})$ can be considered as $R(\mathcal{A})$, the 'rim of potential access' to the definite basin $B_d(\mathcal{A})$. There is no comparable concept in the deterministic world of synchronous RBNs, but with ARBNs a state can lie in the possible basins of more than one attractor (point or loose attractor). This is demonstrated with a simple example in Figure 4.

In the language of digraphs, or directed graphs, a set of points such that there is a directed pathway between any two is termed a *strongly directed component* (SDC). Our definition of loose attractor corresponds to such a SDC; the trajectory through state space is ergodic wandering though all the states within it. There are well known results concerning similar components in *undirected* random graphs [4, 2]. As connectivity between points in a random graph increases, there is a threshold after which a 'giant component' emerges, which contains a majority of the points. The threshold is when there are $N/2$ edges in an undirected graph of N nodes. At a further threshold of $0.5Nlog(N)$ edges the graph becomes fully connected, and Hamiltonian cycles start to appear.

There seems to be a shortage of comparable results for digraphs. In [12, 11] experimental results for simulated random digraphs suggest that the threshold for the emergence of a giant SDC basin of attraction is when there are about $N\pi/2$ edges.

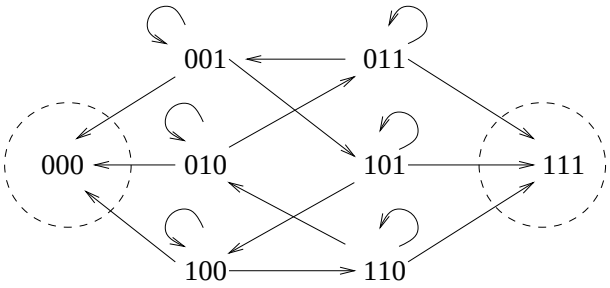

Figure 4: *The possible paths through state space of a specific ARBN with $N = 3$, $K = 1$. Each node when updated has a Boolean function which makes that node copy the value of the node to the left (with wrap-round). Each state has 3 possible paths leading from it, depending on which node is updated; the states 000 and 111 are point attractors from which all paths return to the same state. All states except these two lie in the possible basins, and rims of potential access, to both PAs.*

7 An Energy Measure for ARBNs

There are interesting analogies between ARBNs and Hopfield networks [6]. In such networks there are a number of nodes or neurons which can take values 0 or 1, which are updated asynchronously; each node has a threshold and weighted links from the other ones, and when updated changes its value according to whether the weighted sum of inputs is above or below the threshold. The universal connectivity of Hopfield networks is comparable to RBNs where $K = N$, but the threshold functions of Hopfield networks are equivalent to a subset of the Boolean functions of RBNs.

The particular properties of Hopfield networks with symmetric weights allowed the definition of an 'energy function' which successive updates tend to minimise. Hence the network converges to local minima of this function, the attractors of state space. There is no obvious comparable energy function for ARBNs.

One function for ARBNs which updates do not in general tend to minimise is 'volatility'. For any particular state of the N nodes this can be defined as the number of nodes which, if chosen for update, will change value. When the volatility is zero we have reached a point attractor, but the volatility fitness landscape is in general complex with many local non-zero minima.

Our initial technique for attempting to find point attractors was to use simulated annealing over the volatility landscape in a manner comparable to Boltzmann machines [5]. This method proved unsuccessful, until the simple strategy of just letting the ARBNs run freely was adopted, with the results reported above.

8 Analysis of ARBNs

For some fixed values of N and K, let us consider the ensemble of all possible different ARBNs of N nodes each with K links. As before, we are assuming that nodes can be distinguished, and hence two ARBNs which are identical bar different ordering of the nodes do indeed count as two different members of this ensemble. The number of members G was calculated above (Section 2) in terms of N and K.

From a randomly chosen state S of an ARBN randomly selected from this ensemble, there are N possibilities for the choice of which node is next to be updated — depending on the random order of updating. Any such choice of node i will result in all other nodes remaining the same, while node i either flips or remains the same depending on its associated Boolean function. As the Boolean functions are randomly set up, we can assume in the absence of other information a probability of 0.5 that there is no change and the system remains in state S when the i^{th} node is updated. Given N possibilities for i, the probability that S remains unchanged whichever node is updated is $0.5^N = 1/2^N$ — the probability that S is a point attractor. We can here multiply together these N probabilities of $1/2$; provided we are considering the ensemble as a whole, these probabilities are independent.

This probability $1/2^N$ of being a point attractor holds true for all 2^N possible states of each of the G members of the ensemble. Hence if we consider all $G \times 2^N$ possible states of members of the ensemble, precisely G of them will be point attractors. Hence the expected number of point attractors in any randomly drawn member of the ensemble — any randomly chosen ARBN — is precisely one. It should be noted that this is *independent* of the values of N and K.

Of course this does not mean that each ARBN has exactly one point attractor (PA); the average of one per ARBN is distributed otherwise. If it was the case that for *each* ARBN from the ensemble, the probabilities of each state being a point attractor were independently $1/2^N$, then we would have a Poisson distribution. Using the approximation $(1-1/2^N)^{2^N} \simeq 1/e$ for large N, we would calculate probabilities of 0, 1, 2 ...n ... point attractors as respectively $1/e$, $1/e$, $1/e^2$ ... $1/e^n$ ... $(1/e \simeq 0.3679)$. This Poisson distribution gives the mean number of PAs as one, and the probability that an arbitrary ARBN has at least one PA as $1 - 1/e \simeq 0.6321$.

However the distribution does not follow this pattern, because for any one ARBN the probability of one state being a PA is *not independent* of the probability of another state also being so. Only within the ensemble as a whole, where we can vary Boolean functions and connecting links, can we treat the probabilities as independent for each state. Between ARBNs the distribution is in fact highly skewed. We can identify specific ARBNs

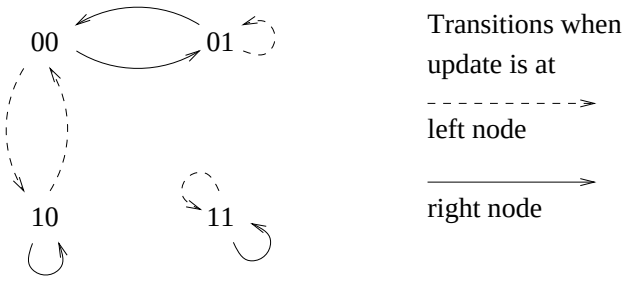

Figure 5: *State space of a particular ARBN with $N = K = 2$ with Boolean functions as shown in the following table. There are 4 possible states for the pair of nodes to have, and at any time there are 2 equally likely possibilities for the next node which happens to be updated.*

Node being updated	self	other	new value
First	0	0	1
	0	1	0
	1	0	0
	1	1	1
Second	0	0	1
	0	1	0
	1	0	0
	1	1	1

Table 2: *Lookup table showing the Boolean functions for the particular $N = 2$, $K = 2$ ARBN of the previous figure.*

from the ensemble which have every one of their possible 2^N states a PA; consider the case where the Boolean functions at every node specify that the node retains its value on update, regardless of the values of any linked nodes. Due to the skewed distribution, we can expect the probability that an arbitrary ARBN has at least one PA will be less than that calculated for the Poisson distribution; the experimental results confirm this.

ARBNs have loose attractors rather than the cyclic attractors of RBNs, because of the indeterminism in order of update. The loose and cyclic attractors can only be one and the same for the special case where $N = 1$ and the Boolean is such that the system cycles between values 0 and 1 at this single node. For $N = 2$ we can construct an example with a single point attractor plus a loose attractor, as seen in Figure 5. Here $N = 2$ and $K = 2$, and the update rules in Table 2 can be summarised as: for whichever node is chosen to be updated, if the *other* node has value 0 the update node should be flipped, otherwise it should stay the same.

This raises the possibility that there may in a large system be one or more point attractors, but their basins of attraction may be insignificant. So we should analyse the probability of a point, or a subset of points, being disconnected from the rest of state space.

Consider first the probability of a point in state space being a 'Garden of Eden' (GoE) state which has no possible ancestors (other than itself[1]). There are N possible neighbours in state space which each *could* (with 50% probability) have been an ancestor, if their Boolean functions had given appropriate values; we are temporarily ignoring the additional possibility that it could be its own ancestor. As with the calculations for the expected number of point attractors, we consider the ensemble of all G possible ARBNs with specific values of N and K. The probability of any one point being a GoE point for any one member of this ensemble is $1/2^N$, the reciprocal of the number of points in state space, giving a total of G GoE points to be shared amongst the G ARBNs; the mean number of GoE states per ARBN is one, again independent of N and K.

For point attractors we argued above that the distribution is skewed, by demonstrating how individual ARBNs with a very large number of PAs can be found. Those very same ARBNs specified above also have every point in state space a GoE state with no possible (different) ancestor. So once again it appears that the distribution is highly skewed. However, outside these abnormal special cases, it would appear that the probability that a point attractor should also happen to be a Garden of Eden point is vanishingly small for large N; so the example shown for $N = 2$ should be considered an aberration.

9 Special cases: K=0, 1

For very small values of K we can analyse the evolution of an ARBN. Setting $K = 0$ implies that the Boolean function at each node depends on no other values, and gives a fixed result (either 0 or 1). Hence from any initial values for the nodes, after each has been updated once they will all have come to rest at their final values, a point attractor in state space; there is always exactly one such attractor.

When $K = 1$ the update rule at each node depends on the value of one other node, and on the Boolean function used. There are 4 possible Boolean functions of one input I, which can be classified as output = 0, 1, I (copy) and $\tilde{\text{I}}$ (differ). In the first two cases the value of I is irrelevant. So we can expect some (on average 50%) of the nodes to be 'volatile' or subject to influence from exactly one other node; and the other nodes will have settled on their final fixed value by the time they have been updated once.

If a volatile node is influenced by another node which itself is (or becomes) frozen in value, this in turn freezes this volatile node. However if there is a Hamiltonian directed cycle of volatile nodes each influencing the next,

[1]There is a further category of Garden of Eden states which do not have themself as a possible ancestor; this category excludes point attractors.

then such a cycle has the potential to stay volatile forever. To check whether such a cycle keeps altering nodes indefinitely one should classify each Boolean function in the chain into 'copy' or 'differ'. If the number of 'differs' in the chain is odd the cycle can never terminate, and the system will not reach a point attractor; indeed there will not be a point attractor for it to reach.

If the number of 'differs' is even then that particular cycle will settle down into one of two possible fixed point attractors, depending on the initial state and the order of updating. For example, as shown in Figure 4 a cycle of 3 nodes each influencing the next with a 'copy' function will remain for ever unchanged if all 3 nodes start off with value 0, or all 3 with value 1; from any other initial values the conclusion will depend on the (random) order of updating. If all such cycles are of this 'even' form (or there are no such cycles) then the system will settle down into a point attractor *from any initial state*; otherwise it will never settle down.

Thus for $K = 1$ we expect that for those ARBNs which have point attractors, their basins of attraction include every possible initial point This is consistent with the experimental results shown in Table 1.

10 Increasing K: 2, 3 ...

If we increase K to 2 or 3, then it is still true that ARBNS have an expected one point attractor each (with a skewed distribution), but in such cases there is now a real chance that many initial positions fall outside their basin of attraction. The example shown in Figure 5 shows one such case for $K = 2$ and $N = 2$; though there is a point attractor, from 75% of possible initial states it is never reached.

The experimental evidence (Table 1) indicates that $K = 3$ gives the smallest chance of arriving at a point attractor. The number of cases where the system sometimes failed to converge to a PA though one was known to exist was 3 out of 38. This still leaves open the possibility that some of the 62 runs which were aborted after failing to reach a PA on the first run did in fact contain PAs that were not found. As K gets larger it becomes increasingly unlikely that a point attractor (and its basin of attraction) should remain isolated from the rest of state space. Although analytical calculations have not been made, this should be expected from consideration of the increase in connectivity of a digraph as the number of edges per node increases.

11 Summarising ARBNs

We are now in a position to draw up a list of properties of ARBNs which can be compared with that given for synchronous RBNs in Section 2.

- $K = N$: the expected number of point attractors for any ARBN is one, though with a skewed distribution. The limited experimental evidence suggests that when a PA exists, there are frequently 2 or 3 of them. When there are such attractors then their basin of attraction generally covers most of state space. When there is no such attractor the system will continue changing ergodically, in a loose attractor.
- $K \geq 5$: the expected number of attractors is the same as when $K = N$; it seems to be more common to have just one or two PAs.
- $K = 3, 2$: the number of attractors still obeys the same law. The skewed distribution includes a small number of ARBNs which have a large number of PAs.
- $K = 1$: the numbers of attractors obeys the same law, but now any such attractors have basins of attraction covering all of state space — there is no chance of 'missing' an attractor if one exists.
- $K = 0$: in this limiting case there is exactly one attractor which is reached from all initial states.

The two lists have such widely different characteristics that there is no sense in which the behaviour of synchronous RBNs gives any clue towards the behaviour of ARBNs.

12 Conclusions

Synchronous RBNs have been much analysed, and their properties have been used as models of biological phenomena such as genetic regulatory networks. They are models at a high level of abstraction, where significant simplifying idealisations have been made. The assumption has been made that despite such idealisations the generic properties of these complex systems is largely robust to those properties that have been ignored.

The Boolean idealisation is one such simplification, and arguments have been presented to justify this as plausible for many classes of biological systems. The Synchrony idealisation is a common practical one to make when approximating continuous systems with fine time-slicing — but this in general falls down when dealing with discontinuous systems such as those produced through the Boolean idealisation. It follows that results from synchronous RBNs may only be applicable to biological systems where there are special reasons to believe that some synchronising mechanism exists.

One might assume that ARBNs have similar behaviour to the synchronous version — the main thrust of this paper is that on the present evidence this would be very mistaken. The non-deterministic nature of ARBNs has prevented the application of methods used to analyse the synchronous version, and it seems some have been deterred from using ARBNs because of their assumed intractability. However it turned out to be remarkably easy

to obtain these very preliminary results for the generic behaviour of ARBNs, both through computer simulations and through analysis.

The indeterminism means that there are no cyclic attractors, only point or loose attractors; the expected number of point attractors in an ARBN is one, independent of the value of K. When such point attractors exist, their basins generally drain most of state space — all of state space in the case of $K = 1$. These characteristics of point attractors differs enormously from those of synchronous RBNs, and there is as yet no evidence of any relationship between loose attractors in ARBNs and cyclic attractors in synchronous RBNs.

In the absence of justifications for the assumption of synchrony, these results cast doubt on the value of synchronous RBNs as models for many biological phenomena, including genetic regulatory networks.

Acknowledgments

This work has been supported with funding from the authors' Universities, and in particular IH acknowledges funding from CSU for a Visiting Research Fellowship to Bathurst where this work was done.

References

[1] A. I. Adamatsky. Complexity of identifying asynchronous nonstationary cellular automata. *Journal of Computer and Systems Science International*, 31(3):127–130, 1993.

[2] B. Bollobas. *Random Graphs*. Academic Press, London, 1985.

[3] F. Dellaert and R. Beer. Toward an evolvable model of development for autonomous agent synthesis. In P. Maes and R. Brooks, editors, *Proceedings of Alife 4*, pages 246–257, Cambridge MA, 1994. MIT Press.

[4] P. Erdos and A. Renyi. *On the Evolution of Random Graphs*. Institute of Mathematics, Hungarian Academy of Sciences, 1960.

[5] G. E. Hinton and T. J. Sejnowski. Learning and relearning in Boltzmann machines. In D. E. Rumelhart and J.L. McClelland, editors, *Parallel Distributed Processing, Explorations in the Microistructure of Cognition, Vol. 1*, pages 282–317. MIT Press, 1986.

[6] J. J. Hopfield. Neural networks and physical systems with emergent collective computational abilities. *Proceedings of the National Academy of Sciences*, 79:2554–2558, 1982.

[7] B. A. Huberman and N. S. Glance. Evolutionary games and computer simulations. *Proc. of the National Academy of Sciences*, 90(16):7716–7718, 1993.

[8] S. A. Kauffman. *The Origins of Order*. Oxford University Press, 1993.

[9] R. M. May, S. Bonhoeffer, and M. A. Nowak. Spatial games and evolution of cooperation. In F. Morán, A. Moreno, J. J. Merelo, and P. Chacón, editors, *Advances in Artificial Life*, pages 749–759, 1995.

[10] M. A. Nowak and R. M. May. Evolutionary games and spatial chaos. *Nature*, 359:826–829, 1992.

[11] D. Seeley. Network evolution and the emergence of structure. In T. Bossomaier and D. Green, editors, *Complex Systems*. Cambridge University Press, 1997.

[12] D. Seeley and S. Ronald. The emergence of connectivity and fractal time in the evolution of random digraphs. In T. Bossomaier and D. Green, editors, *Complex Systems: from Biology to Computation*. IOS Press, Amsterdam, 1992.

[13] M. Sipper, M. Tomassini, and M.S. Capcarrere. Evolving asynchronous and scalable non-uniform cellular automata. In *Proc. of Intl. Conf. on Artificial Neural Networks and Genetic Algorithms (ICANNGA97)*. Springer-Verlag, 1997.

[14] D. Thieffry and R. Thomas. Logical synthesis of regulatory models. In *Self-Organization and Life: From Simple Rules to Global Complexity, European Conference on Artificial Life (ECAL-93)*. Brussels, Belgium, 1993.

[15] R. Thomas. Boolean formalisation of genetic control circuits. *Journal of Theoretical Biology*, 42:563–585, 1973.

[16] A. Wuensche. The ghost in the machine: Basins of attraction of random boolean networks. In C. G. Langton, editor, *Artificial Life III*, pages 465–501, Reading, MA, 1994. Addison-Wesley.