DOI: http://dx.doi.org/10.7551/978-0-262-31050-5-ch035

Finding Optimal Random Boolean Networks for Reservoir Computing

David Snyder[1], Alireza Goudarzi[2], and Christof Teuscher[3]

[1]Department of Computer Science
[2]Systems Science Graduate Program
[3]Department of Electrical and Computer Engineering
Portland State University, Portland, OR 97201
[1]dsnyder@cecs.pdx.edu [2]alirezag@cecs.pdx.edu [3]teuscher@pdx.edu

Abstract

Reservoir Computing (RC) is a computational model in which a trained readout layer interprets the dynamics of a component called a reservoir that is excited by external input stimuli. The reservoir is often constructed using homogeneous neural networks in which a neuron's in-degree distributions as well as its functions are uniform. RC lends itself to computing with physical and biological systems. However, most such systems are not homogeneous. In this paper, we use Random Boolean Networks (RBN) to build the reservoir. We explore the computational capabilities of such a RC device using the temporal parity task and the temporal density classification. We study the sufficient dynamics of RBNs using *kernel quality* and *generalization rank* measures. We verify findings by Lizier et al. (2008) that the critical connectivity of RBNs optimizes the balance between the high memory capacity of RBNs with $\langle K \rangle < 2$ and the higher information processing of RBNs with $\langle K \rangle > 2$. We show that in a RBN-based RC system, the optimal connectivity for the parity task, a processing intensive task, and the density classification task, a memory intensive task, agree with Lizier et al.'s theoretical results. Our findings may contribute to the development of optimal self-assembled nanoelectronic computer architectures and biologically-inspired computing paradigms.

Introduction

In this paper, we propose discrete Boolean networks with heterogeneous in-degrees and transfer functions for Reservoir Computing (RC). Reservoir computing is an emerging paradigm in Recurrent Neural Networks (RNN) (Haykin, 2009) that promotes computing using intrinsic dynamics of an excited system called the reservoir (Lukosevicius and Jaeger, 2009). The reservoir acts as a temporal kernel function, projecting the input stream into a higher dimensional space, thereby creating features for the readout layer. To produce the desired output, the readout layer performs a dimensionality reduction on the traces of the input signal in the reservoir. A system with sufficiently rich dynamics can remember perturbations by an external input over time. Two advantages of using RC are: computationally inexpensive training and flexibility in reservoir implementation. This makes RC suitable for emerging unconventional computing paradigms, such as computing with physical phenomena (Fernando and Sojakka, 2003) and self-assembled electronic architectures (Teuscher et al., 2009). Maass et al. (2002) initially proposed a version of RC called Liquid State Machine (LSM) as a model of cortical microcircuits. Independently, Jaeger (2001) introduced a variation of RC called Echo State Machine (ESM) as an alternative RNN approach for control tasks. Variations of both LSM and ESM have been proposed for many different machine learning and system control tasks (Lukosevicius and Jaeger (2009)). Büsing et al. (2010) conducted a comprehensive study of reservoir performance using different metrics as a function of connectivity K, the logarithm of the number of states per node m, and the variance of the weights in the reservoir. They concluded that for binary networks the performance of an RC system is max-

imized for sparse reservoir networks, but as the number of states per node increases the performance becomes insensitive to sparsity. Insofar, most of the RC research is focused on reservoirs with homogeneous in-degrees and transfer functions. However, due to high design variation most self-assembled systems are heterogeneous in their connectivity and transfer functions. Wang et al. (2006) introduced a hybrid RC device that uses both sigmoidal and wavelet nodes and showed that it improved the reservoir performance. Here, we use a well-known simple heterogeneous Boolean network model for the reservoir. Kauffman (1969) first introduced this model to study gene regulatory networks. Kauffman showed these Boolean networks to be in a complex dynamical phase at "the edge of chaos" when the average connectivity (in-degree) of the network is $\langle K \rangle = 2$ (critical connectivity). Rohlf et al. (2007) showed that near critical connectivity information propagation in Boolean networks becomes independent of system size. Goudarzi et al. (2012) studied adaptive computation and task solving in Boolean networks and found that learning drives the network to the critical connectivity $\langle K \rangle = 2$. Here, we show that heterogeneous discrete Boolean networks in the super-critical regime ($\langle K \rangle > 2$) can be used as the reservoir to perform non-trivial computation. To the best of our knowledge this is the first time that discrete, heterogeneous dynamical networks have been used in reservoir computing.

Experimental Setup

Model

Structurally, RC is made up of three parts: input layer, reservoir or kernel, and readout layer. The input layer excites the reservoir by passing the input signal to it and the readout layer interprets the traces of the input signal in reservoir dynamics to compute the desired output. In our model, the reservoir is a Random Boolean Network (RBN). The fundamental subunit in a RBN is a node with K input connections. At any instant in time, the node can assume either of the two binary states "0" or "1." The node updates its state at time t according to a K-to-1 Boolean mapping of its K inputs. Therefore, the state of a single node at time $t+1$ is only determined by its K inputs at time t and by one of the 2^{2^K} Boolean functions used by the node. Formally, a RBN is a collection of N such binary nodes. For each node i out of N nodes, the node receives K_i inputs, each of which is connected to one of the N nodes in the network. In this model, self-connections are allowed.

The network is random in two different ways: 1) the source nodes for an input are chosen from the N nodes in the network with uniform probability and 2) the Boolean function of node i is chosen from the $2^{2^{K_i}}$ possibilities with uniform probability. Each node sends the same value on all of its output connections to the destination nodes. The average connectivity will be $\langle K \rangle = \frac{1}{N}\sum_{i=1}^{N} K_i$. We study the properties of RBNs characterized by N nodes and average connectivity $\langle K \rangle$. This refers to all the instantiations of such RBNs. Once the network is instantiated, the collective time evolution at time t can be described as using $x_i^{t+1} = f_i(x_1^t, x_2^t, \ldots, x_{K_i}^t)$, where x_i^t is the state of the node i at time t and f_i is the Boolean function that governs the state update of the node i. The nodes are updated synchronously, i.e., all the nodes update their state according to a single global clock signal.

From a graph theoretical perspective, a RBN is a directed graph with N vertices and $L = \lfloor \langle K \rangle N \rfloor$ directed edges. We construct the graph according to the random graph model (Erdös and Rényi, 1959). We call this model a heterogeneous RBN because each node has a different in-degree. In the classical RBN model, all the nodes have identical in-degrees and therefore are homogeneous. The original model of Kauffman (1969) assumes a static environment and therefore does not include exogenous inputs to the network. To use RBNs as the reservoir, we introduced I additional input nodes that distribute the input signals to the nodes in the network. The source node of K_i links for each node i is randomly picked from $N + I$ nodes with uniform probability. The input nodes are not counted in calculating $\langle K \rangle$. For online computation, the reservoir is extended by a separate readout layer with O nodes. Each node in the readout

layer is connected to each node in the reservoir. The output of the node o in the readout layer at time t is denoted by y_o^t and is computed according to $y_o^t = sign\left(\sum_{j=1}^{N} \alpha_j x_j^t + b\right)$. Parameters α_j are the weights on the inputs from node j in the reservoir to node o in the readout layer and b is the common bias for all the readout nodes. Parameters α_j and b can be trained using any regression algorithm to compute a target output (Jaeger, 2001).

Measures

Kernel Measurements We characterize the quality of the reservoir by measuring *Kernel Quality* (KQ) and *Generalization Rank* (GR). KQ measures how well the reservoir separates different input streams and GR measures how well the reservoir classifies similar input streams in the same class. This is called the *separability* property (Maass et al., 2002). Büsing et al. (2010) introduced kernel quality KQ as a practical measure that can directly quantify this requirement in any given reservoir. Kernel quality is formally defined as the rank ρ of the matrix $\mathcal{M}$ of which columns are reservoir states after being driven by random input signals. To measure this quantity for a reservoir with N nodes, we first create S random input signals of length $\mathcal{T}$, where $S = N$ so that a square matrix is formed. We drive the reservoir by each input signal for $\mathcal{T}$ time steps and store the state of each node in the reservoir in a column of $\mathcal{M}$ and calculate $\rho(\mathcal{M})$. GR is calculated the same way as KQ, but using input streams in which the last τ bits are identical. Thus reservoirs with optimal separability will have a high KQ and a low GR. We identify the optimal separability by finding the class of reservoirs in which the difference $\Delta = |KQ - GR|$ is maximal.

Tasks We used the temporal parity and density classification tasks to test the experimental performance of the reservoir systems. According to the task, the RC system is expected to continuously evaluate n bits which were injected into the reservoir beginning at $\tau + n$ time steps in the past. Each task necessitates knowledge of the entire window in addition to its unique requirements. The parity task demands more processing, while in the relatively simple density task, memory of the input is more significant.

Temporal Parity The task determines if n bits $\tau + n$ to τ time steps in the past have an odd number of "1" values. Given an input stream u, where $|u| = \mathcal{T}$, a delay τ, and a window $n \geq 1$,

$$PAR_n(t) = \begin{cases} u(t-\tau) & : n = 1 \\ \oplus_{i=0}^{n-1} u(t-\tau-i) & : \text{otherwise} \end{cases}$$

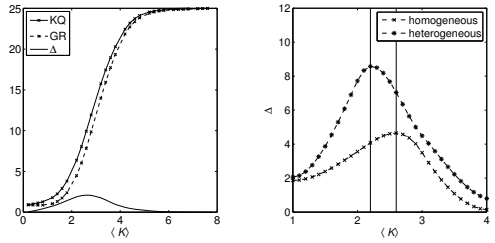
where $\tau + n \leq t \leq \mathcal{T} - \tau - n$.

Temporal Density The task determines whether or not an odd number of bits $\tau + n$ to τ time steps in the past have more "1" values than "0." Given an input stream u, where $|u| = \mathcal{T}$, a delay τ, and a window $n = 2k + 1$ where $k \geq 1$,

$$DNS_n(t) = \begin{cases} 1 & : 2\sum_{i=0}^{n-1} u(t-\tau-i) > n \\ 0 & : \text{otherwise} \end{cases}$$

where $\tau + n \leq t \leq \mathcal{T} - \tau - n$.

Training and Evaluation For every system, we generate 150 random input streams of $\mathcal{T} = 10$ to comprise a training set, and likewise, 150 randomly generated input streams for the testing set. We train the output node with a form of gradient descent in which the weights of the incoming connections are adjusted after every time step in each training example. Given our system and tasks, this form of gradient descent appears to yield better training and testing accuracies than the conventional forms. We use a learning rate $\eta = 0.01$, and train the weights for up to 20,000 epochs. The accuracy of a single training or testing stream is determined by the number of times that the output of the network at time t matches the expected output as specified by the task divided by the total number of values in the output stream. The accuracy on each input set is summed together, and divided



(a) KQ, GR, and Δ for heterogeneous RBNs. (b) Δ for homogeneous and heterogeneous RBNs.

Figure 1: (a) shows the kernel quality KQ, the generalization rank GR, and their difference $\Delta = |KQ - GR|$ of RBN reservoirs with $N = 25$ and $0.10 \leq \langle K \rangle \leq 7.90$. We find the optimal connectivity $\langle K \rangle = 2.25$ at which Δ is maximum. (b) shows the difference Δ for RBNs with homogeneous ($- \times -$) and heterogeneous ($- * -$) in-degree distribution. The peak of the difference Δ predicts greater computational power for heterogeneous RBNs at all connectivities, and is maximum at a lower average connectivity $\langle K \rangle = 2.25$.

by the total number of input streams in the set to produce either the training accuracy T or testing accuracy (generalization capability) G. We are interested in finding the optimal average connectivity $\langle K \rangle$ that maximizes G.

Results

Reservoir Quality

We conduct a comprehensive study of kernel quality and generalization rank in RBN reservoirs for different system sizes N and average connectivity $\langle K \rangle$. Despite the fundamental difference between the reservoirs used in our study and the ones used in (Büsing et al., 2010) (see Models), we find similar behavior in kernel quality KQ and generalization rank GR of RBN. Figure 1 shows KQ and GR for RBNs with size $N = 25$ as a function of the average connectivity $\langle K \rangle$. We used a Spline fit through the data points to create an approximate model for KQ, GR, and Δ.

While both KQ and GR follow a transition-like curve, where near $\langle K \rangle \approx 2.0$ they undergo a sudden jump, Δ only shows a maxima at $\langle K \rangle = 2.25$ and decreases for $\langle K \rangle \geq 2.25$. This peak in Δ value represents an optimal connectivity for RBN reservoirs that optimizes both memory and information processing (Lizier et al., 2008).

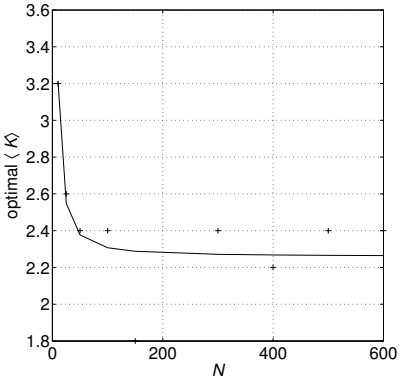
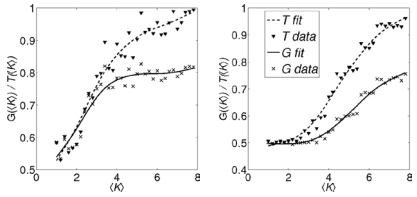


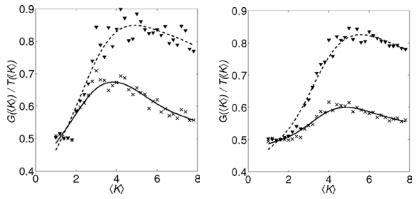
Figure 2: The optimal connectivity for finite system sizes $10 \leq N \leq 600$. The best fit through the data point shows a power-law decay of optimal connectivity with increasing system size (N).

The Δ on Figure 1(a) corresponds to RBNs with binomial in-degree distribution (heterogeneous). To understand the effect of in-degree distribution we compare the Δ calculation for RBNs with uniform in-degree distribution (homogeneous). Figure 1(b) shows the curve of Δ for different $\langle K \rangle$. We see that for heterogeneous RBNs, Δ peaks at $\langle K \rangle = 2.25$, whereas for homogeneous RBNs, Δ peaks at $\langle K \rangle = 2.65$. Also, the value of the Δ for heterogeneous RBNs is twice as high as the Δ for homogeneous systems, which signifies a greater separation ability of the RBN reservoir (Büsing et al., 2010).

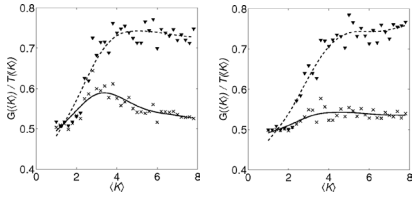
The significance of the optimal connectivity that maximizes the difference Δ in RC is twofold: increased memory capacity and high classification power. Suitable reservoirs need to retain the perturbations from the past input signals to be able to make computations in time. In addition to memory capacity, the optimal connectivity increases the ability of the reservoir to differentiate the input signals that are very different to each other while



(a) $DNS_3, \tau = 1$ (b) $PAR_3, \tau = 1$



(c) $DNS_3, \tau = 7$ (d) $PAR_3, \tau = 7$



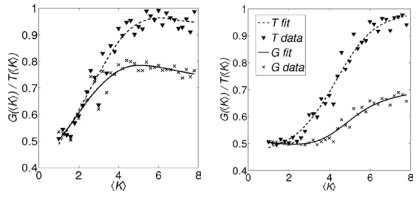
(e) $DNS_3, \tau = 13$ (f) $PAR_3, \tau = 13$

Figure 3: Accuracies for a window size of $n = 3$ and a system size of $N = 500$ as a function of τ and $\langle K \rangle$. Dashed lines illustrate the training accuracy whereas solid lines correspond to the generalization accuracy. The best generalization decreases as τ increases, but favors increasingly small $\langle K \rangle$.

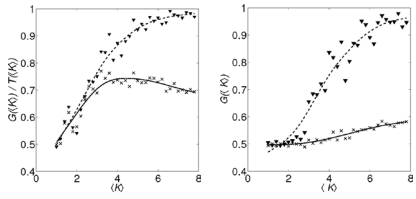
classifying similar (but non-identical) inputs to the same class.

Task Solving

To establish the usefulness of the reservoir quality measurements in RBN reservoirs and to determine the computational power of these RC systems, we use the temporal parity PAR_n and density classification DNS_n tasks. We create reservoirs of various size N with connectivity $\langle K \rangle$ to solve the temporal tasks with different time delays, τ. The temporal parity and the temporal density tasks are both complex tasks that require a knowledge of the entire window of bits over which the



(a) $DNS_5, \tau = 1$ (b) $PAR_5, \tau = 1$



(c) $DNS_7, \tau = 1$ (d) $PAR_7, \tau = 1$

Figure 4: Generalization accuracy for DNS_n and PAR_n tasks for $n = 5$ and $n = 7$. For DNS_n with high window size n, the optimal $\langle K \rangle$ decreases as n increases. This behavior is analogous to what we observe for tasks requiring smaller n and larger τ. For PAR_n the difficulty increases for high n and thus the optimal connectivity increases. Dashed lines illustrate the training accuracy T whereas solid lines correspond to the generalization accuracy G.

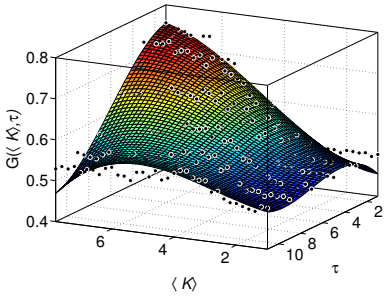
task is being computed. However, solving the parity task requires the reservoir to remember all the incoming bits perfectly, otherwise the performance cannot be better than chance. On the other hand, the density task is more decomposable and a classifier with imperfect knowledge of the input bits may still predict a correct answer if it learns the underlying structure of the task. This decomposability of the tasks can be easily characterized using information theoretical reconstructability analysis (Zwick, 2004) of the truth table of the tasks (Goudarzi et al., 2012). Figure 3 shows how the interplay between task complexity, memory capacity and classification capability of the reservoir affects the training and generalization performance in RBN-based RC. We found that larger system sizes accentuate the generalization accuracy trends and so here we use a reservoir of size $N = 500$.

For both tasks we see that the optimal training

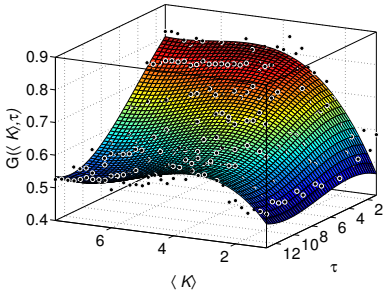
and generalizations for small time delay τ occur at high average connectivity $\langle K \rangle = 8$. As the time delay τ increases, the optimal generalization and training accuracies are at lower values of $\langle K \rangle$. We observe that the shift in the optimal connectivity towards lower $\langle K \rangle$ is faster for the density task than the parity task as τ increases from 1 to 13. We also observe that higher connectivity $\langle K \rangle$ causes the training process to overfit the reservoir dynamics. This becomes obvious by reduced generalization accuracy while the training accuracy stays constant or increases. We combined the generalization trends seen in Figure 3 as well as those of other values of τ to construct generalization surfaces for density and parity as functions of $\langle K \rangle$ and τ, using polynomial interpolation of the data points. We observe that the generalization accuracy of the parity task is very sensitive to both average connectivity $\langle K \rangle$ and time delay τ while the generalization accuracy of the density task is more robust to the decrease in $\langle K \rangle$.

The effect of increasing the window size n is similar to increasing τ. In Figure 4(a) and Figure 4(c) we see that the generalization ability of the networks computing DNS_5 and DNS_7 is greatest when $\langle K \rangle$ is lower than in the optimal connectivity shown for DNS_3 in Figure 3(a). We find that the optimal properties of the reservoir on the task DNS_3 when subject to $\tau = 5$ are similar to those of DNS_7 with $\tau = 1$. Increasing τ and n both put more demand on the memory of the reservoir. This drives the optimal connectivity towards low $\langle K \rangle$, where persistent memory is higher. On the other hand, for PAR_n the difficulty of the computation increases exponentially and pushes the optimal connectivity to higher $\langle K \rangle$.

We can explain the observable trends of generalization accuracy as a function of $\langle K \rangle$ and τ using the task complexity and kernel quality measurements. In RC, the reservoir is a collection of coupled filters. Successful computation in RC depends on the reservoir's ability to transform the fluctuations in the input signal into correlated fluctuations in the reservoir. If enough nodes in the reservoir can be perturbed by the input signal, the readout layer will be able to find a suitable mapping between the reservoir state and the target output. If the desired task is highly nonlinear, the readout layer will require more correlated variations in the reservoir to be able to find the right mapping. In addition to task complexity, the low time delay requirement favors reservoirs with high connectivity. Solving a task with low time delay implies that the fluctuations in the input signal should be propagated to enough nodes in the reservoir within the required time delay τ, which is true in networks with higher connectivity. On the other hand, tasks with a long time delay need to extract and manifest the fluctuations in the input signal after longer time intervals and therefore require less connectivity. For long-time-delay tasks,

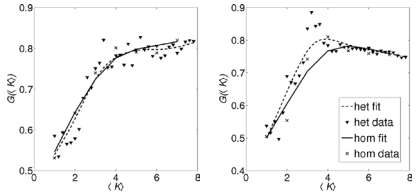


(a) $PAR_3(\langle K \rangle, \tau)$



(b) $DNS_3(\langle K \rangle, \tau)$

Figure 5: Memory of prior input fades over time, so the generalization accuracy of the RC system diminishes as τ increases. Tasks which require a reservoir to remember old inputs perform best at a $\langle K \rangle$ which is smaller than the optimal $\langle K \rangle$ for those reservoirs that have less time to process input.



(a) $DNS_3, \tau = 1$ (b) $DNS_3, \tau = 3$

(c) $DNS_3, \tau = 5$ (d) $DNS_3, \tau = 7$

Figure 6: Generalization accuracies $G(\langle K \rangle)$ for DNS_3 of homogeneous in-degree RBNs superimposed over the corresponding measurement for heterogeneous RBNs.

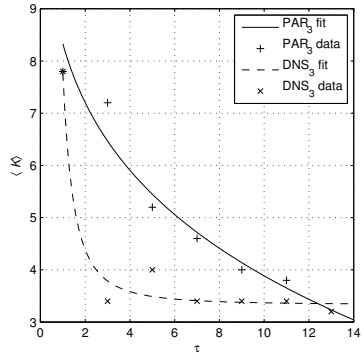


Figure 7: Optimal connectivity of RBN reservoirs for online computation as a function of time delay τ. Solid and dashed lines correspond to the optimal connectivity for the temporal parity and the temporal density tasks respectively.

higher connectivity in networks results in fast percolation of the input signal followed by rapid distortion of the correlations between reservoir dynamics and the past signal due to the new input. In other words, memory of the older inputs will be quickly wiped by the new inputs. Therefore, high connectivity networks achieve higher information processing while having lower memory capacity and low connectivity networks have lower information processing power and higher memory capacity (Lizier et al., 2008). This trade-off between information processing and memory explains the generalization accuracy trends.

For the more complex task of parity with a low time delay, only the reservoirs with high connectivity may extract the required features of the inputs instantaneously (i.e., $\tau = 1$). As we increase τ, the reservoir requires a lower connectivity to solve the task. We observe the same trend in the density task as well.

We test the generalization accuracy $G(\langle K \rangle)$ of the homogeneous RBN reservoir for the DNS_3 task and we find that the generalization performance of the RBN reservoir is invariant against the in-degree heterogeneity (see Figure 6).

Discussion

Our results show complex interactions in real-time computation in RC between task complexity, time delay τ, and average connectivity $\langle K \rangle$. We see that optimal connectivity for online computation is a function of the task complexity and the time delay required by the task. We study the change in optimal connectivity as a function of τ by plotting the connectivity of the highest G for different τ (Figure 7). The solid and dashed lines of Figure 7 correspond to the parity and the density tasks respectively. Both curves show that lower connectivity reservoirs are more suitable for online computation as the time delay increases. This lower $\langle K \rangle$ which produces the optimal G in memory intensive tasks may approach that of the $\langle K \rangle$ found to achieve the optimal separability (see Reservoir Quality). The optimal connectivity for the density task follows a sharper decrease for higher τ, due to its relative simplicity in comparison to the parity task (see Task Solving). The parity task is highly nonlinear. Even for high τ, which demands greater memory, the reservoir requires higher connectivity to be able to extract features in a way that are classifiable by the linear readout layer. As in the density task, the parity task requires lower connec-

tivity when τ is high. However, for $1 < \tau < 13$, the optimal $\langle K \rangle$ of PAR_n is higher than that of DNS_n, for all $n > 1$ that we explored.

Conclusion

In this study, we investigated RC using RBNs and observed that for online computations, RBNs show a trade-off between memory capacity and information processing for different average connectivity $\langle K \rangle$. Because of the flexibility of reservoir implementations, RC lends itself to computing with unconventional devices. RC has been proposed for computing with preexisting systems, and while many of these systems have high variation in their individual components, investigation has been primarily towards reservoirs of homogenous networks, both in functions as well as connectivity. By investigating the optimal connectivities of discrete, heterogenous networks, we have begun to bridge the gap between those reservoirs commonly investigated, and those that could model natural phenomena. In a future study, we will investigate the relation between optimal connectivity and task complexity. The information-theoretical framework developed in (Prokopenko et al., 2011) may help us explore the connection between optimal computation and the nature of the dynamical phase transition in the reservoir.

Acknowledgements

The work was supported by PSU's MCECS Undergraduate Research & Mentoring Program as well as NSF grants #1028120 and #1028378.

References

Büsing, L., Schrauwen, B., and Legenstein, R. (2010). Connectivity, dynamics, and memory in reservoir computing with binary and analog neurons. *Neural Computation*, 22(5):1272 – 1311.

Erdös, P. and Rényi, A. (1959). On random graphs. *Publ. Math. Debrecen*, 6:290–297.

Fernando, C. and Sojakka, S. (2003). Pattern recognition in a bucket. In *Proceedings of the 7th European Conference on Advances in Artificial Life (ECAL 2003)*, volume 2801 of LNCS, pages 588–597. Springer.

Goudarzi, A., Teuscher, C., Gulbahce, N., and Rohlf, T. (2012). Emergent criticality through adaptive information processing in boolean networks. *Phys. Rev. Lett.*, 108:128702.

Haykin, S. (2009). *Neural Networks and Learning Machines (3rd Edition)*. Pearson Education, Inc., New York, NY.

Jaeger, H. (2001). The "echo state" approach to analysing and training recurrent neural networks. Technical Report GMD Rep. 148, St. Augustin: German National Research Center for Information Technology.

Kauffman, S. A. (1969). Metabolic stability and epigenesis in randomly constructed genetic nets. *Journal of theoretical biology*, 22(3):437–467.

Lizier, J., Prokopenko, M., and Zomaya, A. (2008). The information dynamics of phase transitions in random Boolean networks. In *Eleventh International Conference on the Simulation and Synthesis of Living Systems (ALife XI)*, pages 374–381, Cambridge, MA, USA. MIT Press.

Lukosevicius, M. and Jaeger, H. (2009). Reservoir computing approaches to recurrent neural network training. *Computer Science Review*, 3(3):127–149.

Maass, W., Natschläger, T., and Markram, H. (2002). Real-time computing without stable states: a new framework for neural computation based on perturbations. *Neural computation*, 14(11):2531–60.

Prokopenko, M., Lizier, J. T., Obst, O., and Wang, X. R. (2011). Relating fisher information to order parameters. *Phys. Rev. E*, 84:041116.

Rohlf, T., Gulbahce, N., and Teuscher, C. (2007). Damage spreading and criticality in finite random dynamical networks. *Phys. Rev. Lett.*, 99(24):248701.

Teuscher, C., Gulbahce, N., and Rohlf, T. (2009). An assessment of random dynamical network automata. *International Journal of Nanotechnology and Molecular Computation*, 1(4):58–73.

Wang, S., Yang, X.-J., and Wei, C.-J. (2006). Harnessing non-linearity by sigmoid-wavelet hybrid echo state networks (SWHESN). In *The 6th World Congress on Intelligent Control and Automation (WCICA 2006)*, volume 1, pages 3014– 3018.

Zwick, M. (2004). An overview of reconstructability analysis. *Kybernetes*, 33(5/6):877–905.