Quantum Recurrent Networks for Simulating Stochastic Processes

Michail Zak[1] & Colin P. Williams[2]

Ultracomputing Group, Mail Stop 126-347, Jet Propulsion Laboratory,
California Institute of Technology, Pasadena, CA 91109-8099
colin@solstice.jpl.nasa.gov, zak@solstice.jpl.nasa.gov

Abstract. We introduce the concept of quantum recurrent networks by incorporating classical feedback loops into conventional quantum networks. We show that the dynamical evolution of such networks, which interleave quantum evolution with measurement and reset operations, exhibit novel dynamical properties finding application in pattern recognition, optimization and simulation. Moreover, decoherence in quantum recurrent networks is less problematic than in conventional quantum network architectures due to the modest phase coherence times needed for network operation.

Introduction

Large scale classical simulations of stochastic processes require vast quantities of random numbers. However, since the pioneering work of Church, Turing, Post and Gödel, it has been known that classical computers can only compute *functions*. In other words, the class of tasks that can be accomplished with a classical computer is exactly equivalent to the class of computable functions. However, as there is no *function* for computing a true random number, classical computers can only feign randomness. The purported calls to the "random number generator" often seen in modern programming languages are, in reality, calls to a *pseudo-random* number generator. A pseudo-random number generator is a deterministic function whose successive outputs pass many of the statistical tests of randomness. Unfortunately, the sequence of outputs can also harbor subtle correlations that are not immediately apparent from the common statistical measures of randomness.

[1] Supported by the JPL Information and Computing Technologies Research Section.
[2] Supported by the NASA/JPL Center for Integrated Space Microsystems and by the JPL Information and Computing Technologies Research Section.

To illustrate this point vividly, consider the RANDU "linear congruential generator", a notoriously bad pseudo-random number generator that was common on IBM mainframes of the 1960s. A linear congruential generator is defined by:

$$N_{k+1} = (\ell N_k + m) \bmod n$$

where ℓ, m, n are fixed integers and $k = 1, 2, 3, \ldots$ The resulting sequence of numbers, appears, superficially, to generate a set of random samples from a uniform distribution that lie in the range 0 to $n-1$ inclusive. We say "superficially" in the sense that, the sequence of numbers $N_1, N_2, \ldots$ passes many statistical tests of randomness. However, there is a subtle correlation lurking amongst these numbers that becomes apparent if you use them to choose a set of (supposedly) random points in a high dimensional space. In particular, if, as in RANDU, $\ell = 65539$, $n = 2^{31}$, $m = 0$ and $N_0 = 1$ then successive triples produced by the generator, N_k, N_{k+1}, N_{k+2}, can be taken to define the x-, y- and z-coordinates of a point in a 3-dimensional space. These points are plotted in fig.1 below from different viewing angles. From most viewing angles the points appear to be randomly distributed. But from a particular viewing angle you can see that they are not at all randomly distributed. In fact they lie in a set of parallel planes. Thus the sequence of supposedly "random" numbers output from the linear congruential generator are not random at all and could give misleading results if used in a numerical simulation of a stochastic process.

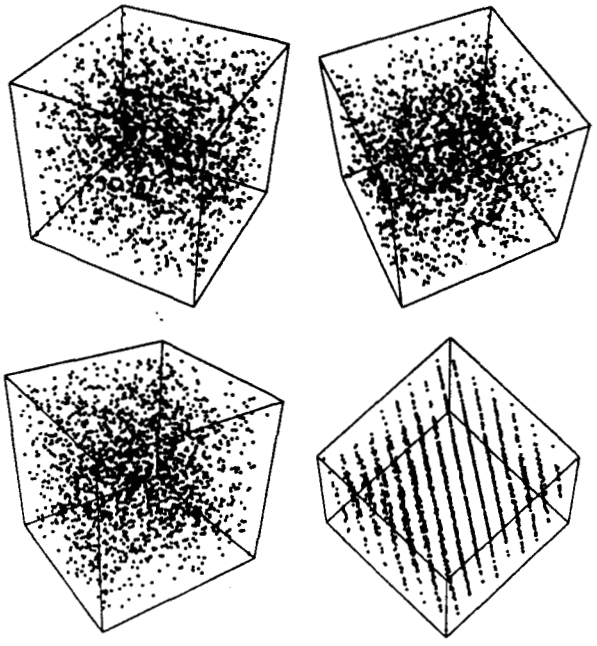

Fig.1 The same cube of points seen from four different viewpoints

Pseudo-random number generators are better today than they were in 1960. But one should not be complacent. As recently as 1992 bugs in a supposedly "good" pseudo-random number generator were discovered when a numerical simulation of an Ising spin system was performed as a test of the simulator against a known benchmark for which analytical results were known [Ferrenberg et al. 92].

One way to correct the problems inherent with pseudo-random number generators is to build a generator that exploits a truly random process itself. Strictly speaking, there is no such thing as randomness in classical physics. Nevertheless, certain classical dynamical systems exhibit a high degree of instability that would seem to make them reasonable candidates for effectively random number generators. Unfortunately, yet again one must be careful. Recently a study of the use of the logistic map, one of the simplest generators of deterministic chaos, has revealed that the chaotic sequence of numbers output does not possess exactly the same statistical properties as a truly random sequence[Phatak & Rao 95]. The deviation might be quite slight, but in situations where billions upon billions of random numbers are needed it could lead a Monte Carlo simulation astray.

The moral is that you cannot use classical physics to generate truly random numbers. However, in quantum physics, the non-deterministic outcome of a measurement made on a system in a superposed quantum state is, *as a matter of principle*, random.

Unfortunately, quantum non-determinism is generally regarded as being of lesser importance than other quantum phenomena such as quantum interference and entanglement. This is partly because many people believe, mistakenly, that pseudo-random number generators are "good enough" and partly because the impressive speedups exhibited by quantum algorithms for factoring composite integers and for finding an item in an unsorted database, are due to interference and entanglement effects rather than non-determinism. However, such a dismissal of quantum non-determinism is premature. No matter how good new pseudo-random number generators are purported to be, their adequacy can only be assessed empirically within the context of a specific application. Moreover, the key quantum effect on which quantum cryptography depends is quantum non-determinism. We argue that as quantum non-determinism is, intrinsically, a random process, it provides a much better basis for the design of a random number generator.

It is easy to define a quantum procedure for selecting random integers in the range 0 to $2^n - 1$ by preparing n qubits in the state $|0\rangle|0\rangle|0\rangle\cdots|0\rangle$, applying the Walsh-Hadamard transform to each qubit separately, creating an equal superposition of all possible states of the register, and then reading the memory register. Once you have a mechanism for generating uniformly distributed random numbers you can create a generator for any other distribution using a function transformation[Tuckwell 95].

However true random number generation is not the same as true stochastic process generation. For example, in a Markov process the probability of obtaining a particular outcome for the next state depends upon the identity of the last state visited. By contrast the sequence of outcomes from a quantum random number generator are independent, identically distributed random variables. It is therefore interesting to ask whether there is a more *direct* way of using quantum mechanics to simulate stochastic processes?

Quantum Recurrent Networks

We can begin be asking what general features must such a simulator possess? First, we need to be able to generate a sequence of classically observable samples. This suggests that we are going to have to imagine a quantum device that allows repeated measurements. Second, we need to be able bias the probability that a given state will appear as the next output given knowledge of some or all of the previous outputs. This is because, by definition, stochastic processes possess such a memory effect. The simplest way to accommodate such a memory effect is to imagine that the device is reset in a new state that somehow takes account of the states visited so far. These considerations lead naturally to our notion of a "quantum recurrent network".

A quantum recurrent network consists of a conventional quantum network augmented with a classical measurement and quantum reset operation. The design of a one dimensional quantum recurrent network is shown in Fig.2.

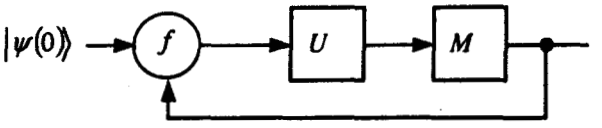

Fig. 2. A One dimensional quantum recurrent network

An initial state, $|\psi\rangle$, is fed into the network, transformed under the action of a unitary operator, U, subjected to a measurement, indicated by the measurement operator $M\{\ \}$, and the result of the measurement is used to control the new state fed back into the network at the next iteration. One is free to record, duplicate or even monitor the sequence of measurement outcomes, as they are all merely bits and hence constitute classical information. Moreover, one is free to choose the function used during the reset phase, including the possibility of adding no offset state whatsoever. Such flexibility makes the QRN architecture remarkably versatile. To simulate a Markov process, it is sufficient to return just the last output state to the next input at each iteration.

Quantum Reset Operation

The reset operation can be accomplished using conditional quantum logic. The basic strategy is to condition the operation performed on the offset state $|\psi\rangle$ upon the last measured outcome, $|i\rangle$. For example, $|\psi\rangle$ describes the state of 3 qubits, then the reset circuit will have the form shown in Fig. 3.

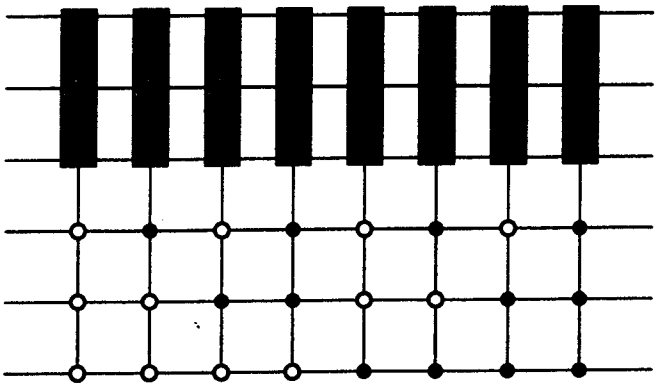

Fig. 3. Most general quantum reset circuit.

Here, a white dot represents the binary value 0 and a black dot the binary value 1. Mathematically, such a reset circuit is described by a 64×64 block diagonal unitary matrix in which each block, S_i, is an 8×8 unitary matrix.

$$\text{Reset Circuit} \equiv \begin{pmatrix} S_0 & & & & & & & \\ & S_1 & & & & & & \\ & & S_2 & & & & & \\ & & & S_3 & & & & \\ & & & & S_4 & & & \\ & & & & & S_5 & & \\ & & & & & & S_6 & \\ & & & & & & & S_7 \end{pmatrix}$$

Thus if the last measured output were, say, $|i\rangle = |011\rangle$ then the state re-entering the circuit at the next iteration will be $S_3|\psi\rangle$. An unfortunate drawback of the reset scheme is that the circuit contains an exponential number of gates. Later, we shall see that it is possible to invent alternative reset strategies that do not require exponentially sized circuits.

Network Dynamics

By design, we imbue the quantum recurrent network with a discrete time evolution according to the equation:

$$|\psi(t+\Delta t)\rangle = f\left(M\{U|\psi(t)\rangle\}, |\psi(0)\rangle\right)$$

where $|\psi(t)\rangle$ is the input to the network at time t, U is a unitary operator defined by $U = \exp(iH\Delta t/\hbar)$, and M is a measurement operator (in the computational basis) that has the effect of projecting the evolved state $U|\psi(t)\rangle$ into one of the eigenvectors of M. The curly brackets are intended to emphasize that M is to be taken as a measurement operation not a matrix product. In the simplest case, the combination function f, can be merely addition followed by renormalization, giving the specialized update rule:

$$|\psi(t+\Delta t)\rangle = \left\| M\{U|\psi(t)\rangle\} + |\psi(0)\rangle \right\|$$

where the notation $\|\cdot\|$ represents renormalization. As each iteration involves a measurement and reset operation, decoherence, the phenomenon that bedevils most hypothetical quantum computations, can be largely ignored, as the quantum recurrent networks need only operate coherently in between successive measurement operations; an interval of duration Δt.

In general, if one were to record the sequence measurement outcomes, it would not settle down to a predictable pattern. Instead, the sequence would hop about erratically between a finite, but possibly exponentially large, number of states, executing a true stochastic process. If the initial, i.e., offset, state vector is

$$|\psi(0)\rangle = \begin{pmatrix} a_0(0) \\ a_1(0) \\ \vdots \\ a_N(0) \end{pmatrix}$$

and M is a measurement operator in the computational basis, then $|\psi(t+\Delta t)\rangle$, the recurrent state re-entering the circuit, must consist of the sum of the offset state, $|\psi(0)\rangle$ plus quantum state (constructed afresh), $|i\rangle$, that corresponds to the last measured outcome, i. Hence, the recurrent state takes one of the forms:

$$|\phi_0\rangle = \frac{1}{\sqrt{R_0}} \begin{pmatrix} 1+a_0(0) \\ a_1(0) \\ \vdots \\ a_{N-1}(0) \end{pmatrix} = \frac{1}{\sqrt{R_0}} \begin{pmatrix} a_0^{(0)} \\ a_1^{(0)} \\ \vdots \\ a_{N-1}^{(0)} \end{pmatrix}$$

$$|\phi_1\rangle = \frac{1}{\sqrt{R_1}}\begin{pmatrix} a_0(0) \\ 1+a_1(0) \\ \vdots \\ a_{N-1}(0) \end{pmatrix} = \frac{1}{\sqrt{R_1}}\begin{pmatrix} a_0^{(1)} \\ a_1^{(1)} \\ \vdots \\ a_{N-1}^{(1)} \end{pmatrix}$$

$$\vdots$$

$$|\phi_{N-1}\rangle = \frac{1}{\sqrt{R_{N-1}}}\begin{pmatrix} a_0(0) \\ a_1(0) \\ \vdots \\ 1+a_{N-1}(0) \end{pmatrix} = \frac{1}{\sqrt{R_{N-1}}}\begin{pmatrix} a_0^{(N-1)} \\ a_1^{(N-1)} \\ \vdots \\ a_{N-1}^{(N-1)} \end{pmatrix}$$

with re-normalization factors:

$$R_0 = |1+a_0(0)|^2 + |a_1(0)|^2 + \ldots$$

$$R_1 = |a_0(0)|^2 + |1+a_1(0)|^2 + \ldots$$

$$\vdots$$

$$R_{N-1} = |a_0(0)|^2 + |a_1(0)|^2 \ldots + |1+a_{N-1}(0)|^2$$

Thus, the recurrent (quantum) states entering the circuit and the measured (classical) outcomes follow the same Markov process. The transition probability matrix, T_1, for this process is given by examining how each of the recurrent states, $|\phi_0\rangle \ldots |\phi_{N-1}\rangle$ evolve under the action of U:

$$T_1 = \begin{pmatrix} \left|\frac{b_0^{(0)}}{\sqrt{R_0}}\right|^2 & \left|\frac{b_1^{(0)}}{\sqrt{R_0}}\right|^2 & \left|\frac{b_2^{(0)}}{\sqrt{R_0}}\right|^2 & \cdots \\ \left|\frac{b_0^{(1)}}{\sqrt{R_1}}\right|^2 & \left|\frac{b_1^{(1)}}{\sqrt{R_1}}\right|^2 & \left|\frac{b_2^{(1)}}{\sqrt{R_1}}\right|^2 & \cdots \\ \vdots & \vdots & \vdots & \ddots \\ \left|\frac{b_0^{(N-1)}}{\sqrt{R_{N-1}}}\right|^2 & \left|\frac{b_1^{(N-1)}}{\sqrt{R_{N-1}}}\right|^2 & \cdots & \left|\frac{b_{N-1}^{(N-1)}}{\sqrt{R_{N-1}}}\right|^2 \end{pmatrix}$$

where

$$b_j^{(i)} = \sum_{\ell=0}^{N-1} U_{j\ell} a_\ell^{(i)} = U_{ji} + \sum_{\ell=0}^{N-1} U_{j\ell} a_\ell(0) .$$

T_1 specifies, therefore, a classical transition probability matrix between a set of quantum states (the recurrent states) or equivalently between a set of classical states (the measurement outcomes).

Stochastic Attractors

The process defined by the transition probability matrix T_1 generates a *truly* random sequence of eigenstates. Although, for a given choice of U and $|\psi(0)\rangle$ every realization of the process will, in general, yield a different sequence of states, the statistical properties of these sequences, such as the frequencies of the various states visited, will eventually converge to a fixed distribution. This fixed distribution, is called a *stochastic attractor*, and may be calculated as the fixed point of T_1 acting on any re-entrant state $|\phi_i\rangle$.

Thus the quantum recurrent network provides a mechanism for generating true stochastic attractors. Our model uses neither pseudo-random number generators nor classical white noise. The time taken to converge to this attractor is governed by the size of the largest eigenvalue of the transition matrix T_1.

To be useful for Monte Carlo simulation, one would like to be able to tailor the quantum recurrent network so that it generates stochastic attractors that have prescribed characteristics. This can be accomplished by a process called "learning". "Learning" consists of adapting model parameters until the quantum recurrent network produces the desired stochastic attractor to within an acceptable tolerance. This can be accomplished by varying the initial state fed into the network, $|\psi(0)\rangle$, the duration of the coherent evolution phase, Δt, or by selecting a different unitary matrix, U. From a practical perspective, as U will be embodied in physical hardware, it will be easier to perform learning by varying just $|\psi(0)\rangle$ and/or Δt. Varying Δt does, of course, change the unitary transformation applied during the coherent evolution phase, so it achieves a similar effect to picking a different U.

An arbitrary $N \times N$ dimensional unitary matrix has exactly N^2 free parameters. Therefore, in principle, a one dimensional QRN can generate a stochastic attractor having up to N^2 degrees of freedom. The method used to find a unitary matrix U that will generate a stochastic process with the desired properties, could be based on analytic minimization, gradient descent or genetic algorithms[Hertz, et al. 91].

$$|\psi(0)\rangle = (-.093-.368i)|000\rangle - (.389+.311i)|001\rangle - (.029-.114i)|010\rangle + (.109-.221i)|011\rangle + (.325-.347i)|100\rangle - (.290+.162i)|101\rangle - (.330-.165i)|110\rangle + .244|111\rangle$$

1000 iterations of the QRN having this $\hat{U}$ and $|\psi(0)\rangle$ induces the stochastic attractor shown in Fig. 5.

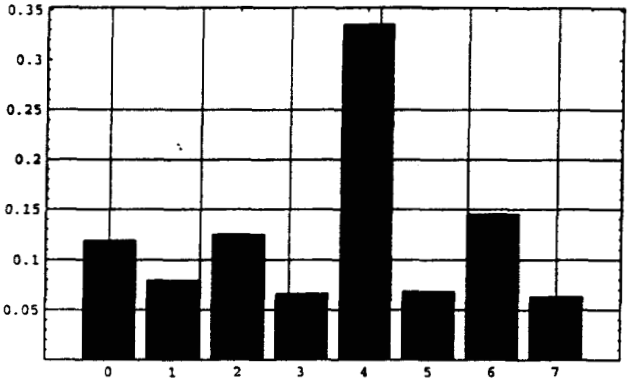

Fig. 5. The attractor generated by the "trained" QRN.

As you can see the agreement between the target attractor and the QRN's attractor is very good. Moreover, one also learns a plausible transition probability matrix for the Markov chain that induces the target attractor.

$$T_1 = \begin{pmatrix} .243 & .009 & .236 & .088 & .121 & .045 & .201 & .057 \\ .173 & .003 & .090 & .069 & .151 & .343 & .048 & .123 \\ .147 & .025 & .319 & .061 & .168 & .066 & .155 & .059 \\ .026 & .033 & .243 & .187 & .371 & .022 & .066 & .052 \\ .129 & .152 & .028 & .001 & .537 & .036 & .095 & .022 \\ .108 & .036 & .010 & .203 & .244 & .069 & .118 & .021 \\ .016 & .139 & .030 & .021 & .319 & .028 & .409 & .038 \\ .059 & .052 & .148 & .063 & .378 & .082 & .048 & .170 \end{pmatrix}$$

This transition probability matrix is not unique because, if the attractor contains N degrees of freedom, the corresponding transition probability matrix contains $N^2 - N$ degrees of freedom. Thus the transition probability matrix is not as constrained as the attractor itself.

Simulating an Eight State Markovian Process

Suppose we want to generate an 8 state Markov process that would generate the stochastic attractor shown in Fig. 4.

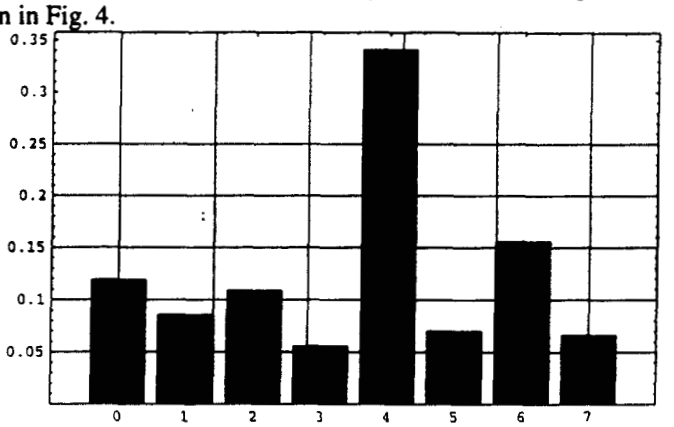

Fig. 4. Stochastic attractor defined on 8 states.

For this attractor, the probabilities of seeing states 0 through 7 are given by 0.119, 0.085, 0.108, 0.055, 0.341, 0.070, 0.156, and 0.066 respectively. As the attractor is one dimensional it can be obtained using a one dimensional QRN acting on 3 qubits. To design such a QRN, we begin with a random unitary matrix, $\hat{U}$

$$\hat{U} = \begin{pmatrix} -.322+.197i & .027+.235i & -.283-.008i & .105-.389i & -.344-.32i & -.112-.385i & .145-.059i & .369+.128i \\ .105+.285i & .072+.342i & .182+.228i & .241+.216i & -.148-.209i & -.182+.352i & .465+.325i & -.169+.151i \\ .252+.206i & -.345-.155i & .31+.326i & .02+.37i & .088-.143i & .005-.433i & .116-.341i & .245+.077i \\ .018-.508i & .061+.162i & -.213-.032i & -.405+.251i & .167-.16i & -.308+.203i & .276-.184i & .336+.161i \\ -.141-.236i & -.189-.333i & -.147-.117i & .149+.297i & -.565+.353i & -.229-.15i & .19+.042i & -.197+.192i \\ -.08-.335i & -.253+.498i & .065+.278i & -.178-.103i & -.15+.164i & .337-.187i & .239-.083i & -.274-.33i \\ .221-.338i & .205+.041i & -.101+.625i & .177-.099i & -.123+.003i & .02-.032i & -.417+.111i & .024+.383i \\ .18+.105i & .172+.342i & -.136-.222i & .019+.417i & -.065+.345i & .314-.189i & -.103+.339i & .429-.007i \end{pmatrix}$$

Given $\hat{U}$, we can "train" the QRN to generate the desired stochastic attractor simply by varying the offset state, $|\psi(0)\rangle$. The training is complete when the difference between the QRN's stochastic attractor and the target stochastic attractor is less than some threshold. For the given $\hat{U}$ and the given target attractor, a satisfactory offset state vector is given by:

The *k*-Parallel Case

Next we generalize the concept of a quantum recurrent network to the case in which there are k networks working in parallel (see Fig. 6).

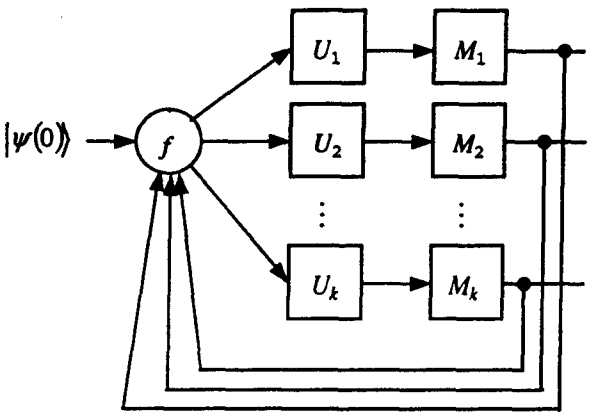

Fig. 6. The k-parallel quantum recurrent network

During the quantum evolution and measurement phases each network acts independently of the rest. However, during the reset operation the results of all the measurements are combined with the initial state to yield k identical input states. Note that the reset operation does not require an *arbitrary* quantum state to be cloned (which is impossible). Instead it only requires that k classical states, the outcomes of the k independent measurements, be copied. As this information is purely classical it can be copied freely. Moreover, the initial state $|\psi(0)\rangle$ is known and can be generated afresh as needed by each of the k networks. As a result, the feedback process we propose is guaranteed to be physically realizable. The purpose of moving to the k-parallel quantum recurrent network is to permit us to generate *multi-dimensional* stochastic attractors. The reset operation gives us the flexibility to introduce correlations between the attractors in each dimension.

For the k-parallel QRN, the elements of the transition probability matrix now define the probability of making transitions between *sets* of measurement outcomes. The state entering each of the k networks at each iteration will have a form such as:

$$\left|\phi_{(i_1 i_2 \ldots i_k)}\right\rangle = \frac{1}{\sqrt{R_{(i_1 i_2 \ldots i_k)}}} \begin{pmatrix} a_0^{(i_1 i_2 \ldots i_k)} \\ a_1^{(i_1 i_2 \ldots i_k)} \\ \vdots \\ a_{N-1}^{(i_1 i_2 \ldots i_k)} \end{pmatrix}$$

where the sequence $i_1 i_2 \ldots i_k$ specifies the last ordered set of measurement outcomes obtained from the k networks and $R_{i_1 i_2 \ldots i_k}$ is the renormalization constant given by:

$$R_{(i_1 i_2 \ldots i_k)} = \left|a_0^{(i_1 i_2 \ldots i_k)}\right|^2 + \left|a_1^{(i_1 i_2 \ldots i_k)}\right|^2 + \ldots \left|a_{N-1}^{(i_1 i_2 \ldots i_k)}\right|^2$$

The mathematical form of the amplitude $a_\ell^{(i_1 i_2 \ldots i_{N-1})}$ depends upon how many of the components in the k-parallel network produced the same measurement outcome at the last iteration through the QRN i.e. how many of the i_ℓ, in the sequence $i_1 i_2 \ldots i_k$ were the same. If outcome i_ℓ is obtained n_{i_ℓ} times we have $a_\ell^{(i_1 i_2 \ldots i_k)} = n_{i_\ell} + a_\ell(0)$.

As there are k networks and each network can produce one of N outcomes (independently), the k-parallel transition matrix defines a mapping from N^k distinct sets of input states to N^k sets of output states. If we denote the probability of transitioning from the set of inputs $i_1 i_2 \ldots i_k$ to the set of outputs $j_1 j_2 \ldots j_k$ by $p_{j_1 \ldots j_k}^{(i_1 \ldots i_k)}$ we have:

$$p_{j_1 \ldots j_k}^{(i_1 \ldots i_k)} = \left|\frac{U_1 b_{j_1}^{(i_1 i_2 \ldots i_k)}}{\sqrt{R_{(i_1 i_2 \ldots i_k)}}}\right|^2 \left|\frac{U_2 b_{j_2}^{(i_1 i_2 \ldots i_k)}}{\sqrt{R_{(i_1 i_2 \ldots i_k)}}}\right|^2 \ldots \left|\frac{U_k b_{j_k}^{(i_1 i_2 \ldots i_k)}}{\sqrt{R_{(i_1 i_2 \ldots i_k)}}}\right|^2$$

where

$${}^U b_j^{(i_1 i_2 \ldots i_k)} = \sum_{s=1}^{k} U_{j i_s} + \sum_{\ell=0}^{N-1} U_{j\ell} a_\ell(0)$$

Thus the k-parallel transition probability matrix has a tensor structure of the form $T_k = \left\{p_{j_1 j_2 \ldots j_k}^{(i_1 i_2 \ldots i_k)}\right\}_{N^k \times N^k}$ where the sequences $i_1 i_2 \ldots i_k$ and $j_1 j_2 \ldots j_k$ are defined with respect to some consistent ordering.

For the k-parallel architecture, there are $k N^2$ free parameters. Thus we ought to be able to generate k-dimensional stochastic attractors having up to $k N^2$ degrees of freedom.

An interesting corollary of the QRN dynamics concerns the dynamical behavior of two k=1 QRNs in comparison to a single k=2 QRN that combines them both. For simplicity we can set the initial (offset) state vector $\left|\psi(0)\right\rangle$ to be zero. Considered separately, the resulting stochastic processes have transition probabilities $p_{j_1}^{(i_1)}$ and $p_{j_2}^{(i_2)}$ given by:

$$p_{j_1}^{(i_1)} = \left| \frac{U_{j_1 i_1}^{(1)}}{\sqrt{2}} \right|^2, \quad p_{j_2}^{(i_2)} = \left| \frac{U_{j_2 i_2}^{(2)}}{\sqrt{2}} \right|^2$$

By contrast the transition probability matrix of the joint QRN has components:

$$p_{j_1 j_2}^{(i_1 i_2)} = \left| \frac{U_{j_1 i_1}^{(1)} + U_{j_1 i_2}^{(2)}}{\sqrt{2}} \right|^2 \times \left| \frac{U_{j_2 i_1}^{(1)} + U_{j_2 i_2}^{(2)}}{\sqrt{2}} \right|^2$$

Clearly $p_{j_1 j_2}^{(i_1 i_2)} \neq p_{j_1}^{(i_1)} \times p_{j_2}^{(i_2)}$ in general. Thus the two dimensional stochastic attractor generated by a 2-parallel quantum recurrent network is not simply the product of two 1-dimensional stochastic attractors.

Alternative Reset Strategies

So far, we have described the most general quantum recurrent network, i.e., one which involves an *arbitrary* offset state and an *arbitrary* unitary operator. Unfortunately, for such QRNs, the reset operation requires a circuit, like that depicted in Fig. 3, which is exponential in the number of qubits. This is because the required reset is different depending on which of the 2^n possible outcomes was obtained at the last measurement. Moreover, to implement an arbitrary unitary matrix as a quantum circuit could, in the worst case, require an *exponential* number of quantum gates[Knill 95]. Both these shortcomings can be sidestepped, however, by a slight modification to the QRN.

Instead of allowing *any* multi-particle state vector to serve as the offset vector we could allow only product states. This would enable the reset operation to be achieved in only a polynomial number of operations. For example, suppose the offset vector for a 2-qubit QRN is the product state $|\psi\rangle_1|\phi\rangle_2$, then, instead of the reset operation with exponential cost, i.e., if a and b are the most recently measured classical outcomes, $|\psi\rangle_1|\phi\rangle_2 \mapsto \| |\psi\rangle_1|\phi\rangle_2 + |a\rangle_1|b\rangle_2 \|$, we could impose a qubit-by-qubit reset operation $|\psi\rangle_1|\phi\rangle_2 \mapsto \| (|\psi\rangle_1 + |a\rangle_1)(|\phi\rangle_2 + |b\rangle_2) \|$. Although the latter operation is conditional too, the conditioning is with respect to each individual qubit rather than the state of the entire multi-qubit register. Thus, the polynomial cost reset circuit would have the form shown in Fig. 7.

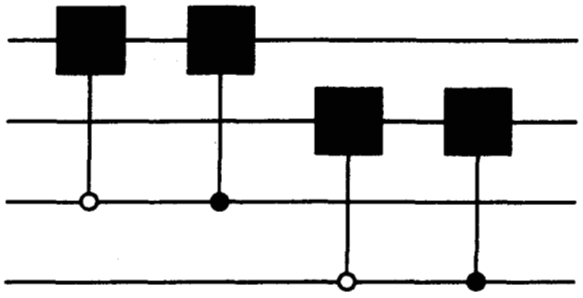

Fig. 7 A polynomially sized reset circuit.

Alternatively, we can imagine another reset strategy inspired by Brassard et al's discovery of a notion of the "closest" product state to an arbitrary entangled state[Brassard & Mor 98]. As we know the ideal entangled state that we should like to feed back into the QRN, we could instead use the closest product state to it. This would perhaps be the best compromise reset, as it would approximate the generic QRN and yet still only require a polynomial cost reset operation.

Likewise, rather than allowing U to be an arbitrary unitary operation, we could require that U be a unitary operation that is implementable in a polynomially sized quantum circuit. That is, we restrict consideration to unitary operators U that can be factored in the form $U = (U_1 \otimes U_2 \otimes \cdots) \cdot (V_1 \otimes V_2 \otimes \cdots) \ldots$ where U_i and V_i are simple unitary operators describing 1-qubit or 2-qubit gates and the total number of terms in the product is bounded by a polynomial in the number of qubits. As a given stochastic attractor can, in general, be obtained from several different transition probability matrices, we have some degree of latitude over the exact choice of U. Thus restricting attention to U's having a special (compact) decomposition ought not to be that limiting.

Summary

Quantum recurrent networks provide a mechanism for generating *true* stochastic attractors. By a process of learning we can tune the free parameters in a QRN to produce stochastic attractors having prescribed characteristics, such as a specific frequency distribution for the states visited. Moreover, as the QRNs operate by interleaving quantum evolution with measurement and reset operations, they are far less sensitive to decoherence than other designs of quantum computers.

Stochastic attractors find a wide range of applications in the physical and computational sciences. For example, one could use quantum recurrent networks as associative memories. Different stimuli, represented by different inputs $|\psi(0)\rangle$ would induce different stochastic attractors. The capacity of such quantum associative memories i.e. the number

of distinct stimuli that can be recognized without unacceptable error, is much higher than for a comparable classical network.

Acknowledgements

The authors gratefully acknowledge financial support for this work from the NASA/JPL Center for Integrated Space Microsystems under contract 277-3R0U0-0 and from the JPL Information & Computing Technologies Research Section under contract 234-8AX24-0.

References

[Brassard & Mor 98] G. Brassard & T. Mor, "Multi-particle Entanglement via 2-particle Entanglement", in Proceedings of the First NASA International Conference on Quantum Computing & Quantum Communications, NASA QCQC'98, Lecture Notes in Computer Science, Springer-Verlag (1998).

[Ferrenberg & Landau 92] A. M. Ferrenberg & D. P. Landau, "Monte Carlo Simulations: Hidden errors from "Good" Random Number Generators", Physical Review Letters, Vol. 69, Number 23, 7th December (1992) pp3382-3384.

[Hertz et al. 91] J. Hertz, A. Krogh & R. G. Palmer, "Introduction to the Theory of Neural Computation", Lecture Notes Volume I, Santa Fe Institute, Studies in Sciences of Complexity, Addison-Wesley Publishing Company (1991).

[Knill 95] E. Knill, "Approximation by Quantum Circuits ", Los Alamos preprint, http://xxx.lanl.gov/archive/quant-ph/9508006, (1995).

[Phatak & Rao 95] S. C. Phatak & S. S. Rao, "Logistics Map: A Possible Random Number Generator", Physical review E, Volume 51, Number 4, April (1995) pp.3670-3678.

[Tuckwell 95] H. Tuckwell, "Elementary Applications of Probability Theory", Second Edition, Chapman & Hall, Texts in Statistical Science (1995).

[Williams & Clearwater 97] C. P. Williams & S. H. Clearwater, "Explorations in Quantum Computing", TELOS/Springer-Verlag, book/CD-ROM, ISBN 0-387-94768-X (1997)

[Zak & Williams 98] M. Zak & C. P. Williams, "Quantum Neural Nets", International Journal of Theoretical Physics, Volume 37, Number 2, (1998) pp651-684.