Eur. Phys. J. D **42**, 49–59 (2007)
DOI: 10.1140/epjd/e2006-00277-9

THE EUROPEAN
PHYSICAL JOURNAL D

Towards parallel computing: representation of a linear finite state digital logic machine by a molecular relaxation process

F. Remacle[1,2] and R.D. Levine[1,3,a]

[1] The Fritz Haber Research Center for Molecular Dynamics, The Hebrew University of Jerusalem, 91904 Jerusalem, Israel
[2] Département de Chimie, B6c, Université de Liège, 4000 Liège, Belgium
[3] Department of Chemistry and Biochemistry, The University of California Los Angeles, 90095-1569 Los Angeles CA, USA

Received 6 August 2006 / Received in final form 26 November 2006
Published online 22 December 2006 – © EDP Sciences, Società Italiana di Fisica, Springer-Verlag 2006

Abstract. A chemical system displaced not far from equilibrium is shown to offer a physical realization of a linear sequential digital logic machine. The requirement from the system is that its state is described by giving the current values of the concentration of different chemical species. The time evolution is therefore described by a classical master equation. The Landau-Teller process of vibrational relaxation of diatomic molecules in a buffer gas is used as a concrete example where each vibrational level is taken to be a distinct species. The probabilities (= fractional concentrations) of the species of the physicochemical system are transcribed as words composed of letters from a finite alphabet. The essential difference between the finite precision of the logic machine and the seemingly unbounded number of significant figures that could be used to specify a concentration is emphasized. The transcription between the two is made by using modular arithmetic that is, is the arithmetic of congruence. A digital machine corresponding to the vibrational relaxation process is constructed explicitly for the simple case of three vibrational levels. In this exploratory effort we use words of only one letter. Even this is sufficient to achieve an exponentially large number of memory states.

PACS. 31.15.-p Calculations and mathematical techniques in atomic and molecular physics (excluding electron correlation calculations) – 05.10.Gg Stochastic analysis methods (Fokker-Planck, Langevin, etc.) – 02.10.De Algebraic structures and number theory

1 Introduction

The physicochemical processes we here consider are described by the methods of chemical kinetics: the variables are concentrations of different species. When the system is displaced from equilibrium the concentrations will evolve in time. A simple case of such time evolution is first order kinetics when the rate of change of the concentration is proportional to the value of the concentration. An example of a process where there are several species is the vibrational relaxation of diatomic molecules dispersed in a large excess of a monoatomic buffer gas. For this example the different species are the different vibrational levels of the diatomic molecule. Various external perturbations can pump a vibrational ladder of levels away from equilibrium. For example, a sudden shock wave, and for this reason the names of Teller and Bethe and Zeldovich are connected with the temporal evolution of this disturbance. The initial displacement can also be created by a short laser pulse or by an exoergic chemical reaction that selectively disposes the energy as vibration of the products. The time evolution that physical chemists call first order kinetics is what statistical mechanics calls a classical) master equation [1] description. For the particular case of vibrational relaxation the master equation has been thoroughly analyzed in a series of papers beginning with Montroll and Shuler [2,3].

Mathematically, that the rate of change of the state variables depends only on their current values means that we deal with a Markovian stochastic process.

Our purpose is to establish a correspondence between the kinetic evolution as described by a master equation and the sequential operation of a linear logic machine. It is an obvious characteristic of chemical kinetic processes that their evolution depends on their present state. For computer science this is a key consideration because it allows us to define machines whose action depends not only on the input but also on the state of the machine. This is known as a finite state machine [4] or finite automata. A finite state machine is inherently more powerful than the, so-called, combinational machines, such as the familiar AND or OR logic gates, where the outputs are a combination of the inputs only and do not depend on what state the machine is in.

[a] e-mail: `rafi@fh.huji.ac.il`

In this paper, we put the emphasis on the correspondence between the physicochemical process and the logical finite state process. This discussion is needed because the people who deal with logic variables recognize explicitly what natural scientists know very well but tend to forget about namely that the result of a measurement can only be specified to a finite precision. When one takes this into account there are only a finite number of possible states of the machine. These states are the memory of the machine. In simple terms the finite state machine combines the inputs with the contents of its memory to provide an output. We show a very large number of memory states is provided by a set of coupled chemical processes.

We here discuss the correspondence between chemical kinetics and finite state machines by emphasizing first order kinetics because this corresponds to a *linear* machine. Over and above the familiar advantages of a finite state machine, we consider that the linearity of the machine provides a further essential property in that it allows for parallel computation. It is this parallelism that is our ultimate motivation. But to take full advantage of it requires some additional mathematics and so, in our first paper on the subject, we limit consideration to showing that we can establish a linear sequential finite state logic operation and that it operates in a parallel manner.

Combinational circuits for binary (or Boolean) logic based on physicochemical systems not in equilibrium have been discussed for sometime [5]. Like the solid state devices that such circuits seek to emulate, the input or output are assigned to have two possible values. We can think of the possible values as 'on' or 'off' or 0 or 1, etc. In chemical kinetics, if we identify the value with the concentration of a species, the restriction to binary values requires that we confine the observation to the question if a species is present or it is not. It is possible to design kinetic schemes that can robustly meet such desiderata [6]. It is however not an easy condition to satisfy and so in this paper we remove this strong requirement. We show how to use the value of the concentration as measured, as a word of the language of logic. Much of the algebraic part of this paper is the development of the machinery necessary for doing so. In essence we generalize from distinguishing between two alternatives, say, T or F (for true or false) to more alternatives. The concrete example we discuss below has five alternatives that we represent as the numbers 0, 1, 2, 3, 4. To keep this paper as simple as possible we shall use words of just one digit or, equivalently words of one letter. This restriction means that in our example a measured concentration has to be assigned to one of the five possibilities 0, 1, 2, 3, 4. If we measure the fraction of systems in a given physical state (= the probability or the mole fraction of the state), it requires that we break the interval of possible answers, which are in the range 0 to 1, to 5 bins. In principle one can measure to a higher precision and the choice of 5 possibilities is just an example. It can be a higher number or we can use words of more than just one letter so that the probability is specified by two significant digits or even three, etc. What we cannot admit are variables with infinite precision. But the reality that we all recognize is that unlimited precision is a fiction and that a result of a measurement has to be represented by a finite number of significant figures. To discretize the mole fraction we use a procedure motivated by the modular mathematics that is employed to describe the operation of finite state machines, as is discussed in Section 3. However one can devise alternative schemes to 'bin' the concentrations.

2 Vibrational relaxation: the Landau-Teller model

The physical system we discuss is one of the best-studied stochastic processes, the vibrational relaxation of diatomic molecules in a dilute environment [7]. The molecules change their vibrational state by a relatively rare perturbation by the medium. The fraction of molecules occupying the different vibrational states at the time t can be arranged as components of a vector $\mathbf{P}(t)$. The time dependence of the concentrations satisfies a master equation [1]

$$d\mathbf{P}\,(t)/dt = \mathbf{A}\mathbf{P}\,(t)\,. \tag{1}$$

The rate matrix $\mathbf{A}$ describes the rate of transitions between the vibrational states and is based on the early work of Landau and Teller who showed that for weak coupling to the medium only neighboring vibrational levels are coupled. Following Montroll and Shuler [2,3] we truncate the set of states available to the oscillator so that the vibrational quantum number is limited to the range $\nu = 0, \ldots, n-1$. Within the Landau-Teller model, equations (1) are

$$\begin{aligned} dP_\nu/dt = \kappa\big\{\nu e^{-\theta} P_{\nu-1} - \big[\nu + (\nu+1)\,e^{-\theta}\big]\,P_\nu \\ + (v+1)\,P_{\nu+1}\big\}, \qquad \nu = 0, \ldots, n-2 \\ dP_{n-1}/dt = \kappa\,\big\{(n-1)e^{-\theta} P_{n-2} - \big[n - 1 + \alpha n e^{-\theta}\big]\,P_{n-1}\big\} \end{aligned} \tag{2}$$

where $\theta = h\nu/kT$ is the effective temperature and κ is a rate constant that sets the time scale. Physically κ is determined by the strength of the coupling between the molecules and the medium that acts as a heat bath. α is a measure of any loss of probability due to the system being open. The simplest example is loss of vibrationally most excited, $\nu = n$ molecules by dissociation. We intend to describe a closed system so that for us the choice is $\alpha = 0$.

In the numerical example below we allow only three vibrational states, so that the population vector $\mathbf{P}$ has only $n = 3$ components. This is already sufficient to give rise to a rich structure and yet the rate $\mathbf{A}$ matrix is only 3 by 3 so that the matrix operations that we require can be easily checked by hand. A larger number of accessible vibrational levels is perfectly possible and the temporal evolution of vibrational ladder of states that can be mimicked by Landau-Teller kinetics with $n > 30$ states has been reported [8,9].

In the language of chemical kinetics the master equations (2) are of the first order because we take the diatomic molecules to be diluted in their environment so that molecule-molecule collisions are rare and can be neglected. For this case the rate matrix $\mathbf{A}$ is independent of the state of the diatomic molecules and the rate of change of the population of the different states is linearly proportional to the populations themselves. In technical terms the master equations are linear because the rate matrix $\mathbf{A}$ is independent of the state of the system so that if $\mathbf{P}$ and $\mathbf{Q}$ are two vectors of populations then $\mathbf{A}(\mathbf{P}+\mathbf{Q}) = \mathbf{AP}+\mathbf{AQ}$. We reiterate that the linearity is a statement about the physics and that one can discuss a gas composed of diatomic molecules with a lower concentration of the buffer gas or even without a buffer [10,11]. The kinetic equations will then not be linear but they will still provide a basis for logic operations, e.g., [12]. In this paper, we center attention on linear systems because they offer a potential for parallelism.

The master equation (1) is here written for a closed system. In other words, there is no inhomogeneous term representing pumping (or draining) from the outside world. This will be shown below to correspond to a, so-called [13], autonomous mode of operation of the sequential logic machine. To describe input and output to the machine we need to go over to an open system. We intend to do so but not in this introductory paper. Finally, a more technical comment. We here assume that the number of diatomic molecules in our experiment is large enough for the law of large numbers to apply. This means that we will not observe an occupation of a vibrational state that is different from the probability of that state. In the long term we will want to go to small systems containing only a few molecules. Fluctuations are then possible. It is known how to generalize the master equation so as to allow a, so-called, stochastic description [2,3,14]. In this introductory account we assume that the sample is large enough that the conventional tools of chemical kinetics apply.

The transition probability matrix $\mathbf{A}$ does not depend on time. It is a non symmetric matrix specified by equations (1) and (2). Physical considerations restrict the matrix in equation (1) as follows [15]. For any current vector of probabilities $\mathbf{P}(t)$ we want the vector at the next time step to also be normalized. Since $\mathbf{P}(t+\delta t) \cong \mathbf{P}(t) + \delta t\,(d\mathbf{P}(t)/dt)$ the conservation of normalization requires that $\sum_\nu (dP_\nu(t)/dt) = 0$. From the master equation, equation (1), this requires that the columns of the matrix $\mathbf{A}$ have to sum up to unity, $\sum_\nu A_{\nu\mu} = 0$ for all μ. The other key condition is that at long times the system should relax to thermal equilibrium at the temperature θ. This is ensured if the matrix $\mathbf{A}$ satisfies detailed balance $A_{\mu\nu}\exp(-\nu\theta) = A_{\nu\mu}\exp(-\mu\theta)$ where $\exp(-\nu\theta)$ is the Boltzmann factor for state ν. This means that the matrix $\mathbf{A}$ can be brought to a real symmetric form by a similarity transformation and therefore can be diagonalized with real eigenvalues. The equilibrium vector, whose components are $\exp(-\nu\theta)/Q$, Q being the sum over states that insures normalization, is the eigenvector of the matrix $\mathbf{A}$ that corresponds to the eigenvalue zero. Explicit expressions for the eigenvectors as well as the necessary distinction between right and left eigenvectors can be found, e.g., in [15].

From equation (1), the vector, $\mathbf{P}(t)$, whose components are the probabilities of the different vibrational states, $\nu = 0,\ldots,n-1$ at the time t can be expressed as

$$\mathbf{P}(t) = \exp(\mathbf{A}t)\,\mathbf{P}(0) \qquad (3)$$

where $\mathbf{P}(0)$ is the corresponding vector at the initial point in time, $t = 0$. Note that because the matrix $\mathbf{A}$ is independent of the state of the machine, equation (3) is a linear law such that if and p_b are fractions that add to one, $p_a + p_b = 1$,

$$p_a\mathbf{P}_a(t) + p_b\mathbf{P}_b(t) = \exp(\mathbf{A}t)\left(p_a\mathbf{P}_a(0) + p_b\mathbf{P}_b(0)\right).$$

The next two sections prepare the background for establishing a correspondence between the temporal evolution of the relaxation of the mole fraction of the different states of the molecule as described by equation (3) and the sequential operation of the logic machine. We deal with finite state machines where n is the number of states. So the correspondence is between vibrational states of the molecule and individual logic states of the machine. Our task is to discretize the evolution such that time increases in finite steps where each increment in time describes one cycle of operation of the logic machine. Even before that we need to transcribe the probabilities of the different vibrational states of the molecule into the words that specify the logic state of the machine. In this paper we use words of just one letter but even so the number of letters is finite while the mole fractions can be specified by equation (3) to as many digits as we wish. Of course we understand that this seemingly unlimited precision is a fiction limited ultimately by the necessarily finite number, N, of molecules in our system so that a mole fraction can only be known to within $1/N$. Before that there are all sources of noise in the measurement of the concentrations. But in chemical kinetics we do adhere to the fiction that concentrations can be known to any desired precision. The first task we address is to remedy this and to do so we need to carry out arithmetic in a finite field.

3 On finite fields

Linear sequential machines [4,13,16,17] are digital machines that operate on finite fields of integers. Many of us a re more used to work with the fiction of observables whose values can be specified to as many significant figures as we care to. Finite fields, while not as familiar, seem to us to offer a better representation of physical reality because numbers are only known to a finite precision. Finite fields of integers are also known as Galois fields and the most common Galois Field is GF(2), the finite field of the binary numbers 0 and 1. In GF(2) addition and multiplication are defined modulo 2. Addition corresponds to an XOR logic gate and multiplication to an AND gate. For a finite field of order p, GF(p), addition and multiplication are defined modulo p. When p is a prime number one can also define an inverse so that the integers modulo p form a finite field with p elements, $0, 1, \dots, p-1$. $a = b$ in GF(p) means that $a \equiv b \bmod p$ which says that $a - b$ is divisible by p, [18,19]. Then, a and b are said to be "congruent modulo p". To make a web search for the procedure note that the arithmetic of congruence is what is known as modular arithmetic. x, the modular inverse of a is defined by $ax = 1 \bmod p$. Note that in modular arithmetic the inverse of p is 0. Several readily available packages for doing symbolic operations implement modular arithmetic. We chose to work with the Mathematica (version 5.1) software [20] but for the example that we implement below all the operations are easy enough that they can be done by hand.

Galois fields can be extended to p^m elements, where m is the number of integers in a sequence where each entry in the sequence is drawn from the integers modulo p. As an example consider the field GF(2^3). There are eight possible sequences of 3 integers, each of them being 0 or 1, namely $(0,0,0)$, $(0,0,1)$, $(0,1,0)$, $(1,0,0)$, $(0,1,1)$, $(1,0,1)$, $(1,1,0)$, $(1,1,1)$. In this paper, we keep things as simple as the subject allows by developing an example using sequences of a single integer, $m = 1$. As we will discuss below with n physical levels and a Galois field of order p there are p^n states of the machine. This number can be quite large. We are however aware that one can have m larger than unity and so end up with the even larger number of p^{nm} states at our disposal. At the same time we should note that a large number of states is not, by itself, sufficient for a fast throughput. We also want rapid transitions between the states and this is governed by the constant κ of the kinetic equations (2).

In the example below we take the very conservative estimate of $p = 5$ different values for the state of the logic machine. It is a conservative estimate because all that we need to ask for from the experiment is to distinguish between 5 alternative values for the logical state. This requires that the experiment determines the first digit in the value of the mole fraction of a vibrational level. Of course, it can be possible to achieve better experimental precision but for distinguishing between five alternatives the ten possible one digit fractions are enough. There are several ways whereby one can establish the correspondence between the experimental reading of the mole fraction of the vibrational state ν and one of the 5 values of the logical state of the state ν of the machine. Intuitively it is clear that the value $p = 5$ allows for some margin of error in the experimental value and that there are different ways of binning the experimental values that are proper fractions into five mutually exclusive and inclusive bins. We chose the mathematical procedure known as affine rationalization bracketing [18] to a specified number of digits. It is implemented in the Mathematica 5.1 software package [20]. What this procedure does is to provide a fraction in a rational form that approximates the given fraction to a pre-specified number of digits. Why affine? Because the experimental state vectors are normalized and we want to preserve this property. To get the logic vectors we need to take one more step. Given a rational fraction we generate an integer between 0 and p by expressing the fraction modulo p. We shall use $p = 5$ and, as an example, the rational fraction 1/9 has the value, modulo 5, of 4 because $9 = 4 \bmod 5$. The transcription is easy for such rational fractions that are of the form $a/4^k$, because modulo 5, $1/4 = 4$. We cannot use fractions of the form $a/5^k$ because in modular arithmetic the inverse of 5 modulo 5 is 0.

To make the numerical example simple enough to be checked by hand we limit the machine to three logic states $n = 3$. This requires from the experiment to detect the mole fractions of the first three vibrational levels, $\nu = 0, 1, 2$. For $p = 5$ there are therefore $p^n = 5^3 = 125$ possible states of the machine. Each logical state vector of the machine, written as a row vector of $n = 3$ components, has the form (m, n, o) where each letter is one of the 5 integers between 0 and 4. As an example we list a few states as column vectors

$$\begin{pmatrix}1\\0\\2\end{pmatrix};\quad \begin{pmatrix}1\\4\\2\end{pmatrix};\quad \begin{pmatrix}2\\0\\3\end{pmatrix};\quad \begin{pmatrix}1\\4\\1\end{pmatrix};\quad \begin{pmatrix}0\\3\\1\end{pmatrix}. \tag{4}$$

Only a subset of logical states correspond to physical states that are normalized, that is whose sum of components = 1 modulo p. Normalized states are shown in equation (16) below. It is therefore worthwhile to comment that the machine uses all the logical states and that this is made clear when the action of the machine operating in a parallel mode is examined. As an example, equation (20) below shows three very useful logic state vectors in column format and the first two vectors are normalized to zero.

4 On the exponential number of logic states

Temporal evolution of vibrational ladders of states that can be mimicked by Landau-Teller kinetics with $n > 30$ states have been reported [8,9]. Even for the conservative values of $p = 5$ and $m = 1$ this corresponds to over 10^{21} memory states of the machine!

Quantum computing [19,21] is also linear but it is linear in the amplitudes rather than the linearity in the probabilities in the quasiclassical scheme that we use (we [12] refer to it as quasiclassical because we use discrete quantum states but neglect interference so the time evolution is described by a master equation). Quantum computing uses two state systems and achieves massive parallelism by coupling systems so that instead of a single integer a word is made up of m integers that are the quantum numbers that are needed to specify a basis state for the combined system. We here use just a single integer but we allow more than 2 levels. Even for $m = 1$, this allows the exponentially large p^n number of states. Quantum aficionados will object that the energy of the system increases with the number of levels. But I_2^- in solution or in clusters [9] does exhibit fully reproducible relaxation kinetics up to $n = 30$ or even higher. It is possible to couple vibrational ladders by intermolecular transfer [11] and this allows us to use words of more than one letter [12]. It is also possible to couple different vibrational ladders of the same molecule, what chemists know as IVR (intramolecular vibrational redistribution) [22]. A word is then the sequence of quantum numbers needed to specify a (zero order) state of the molecule. Time evolution via IVR also allows for faster transitions between states. We reserve the doubly exponential potentialities of p^{nm} states to future work. Another option is to allow not only for energy but also for charge and/or atom transfer [7,23] as means of enlarging the repertoire of states that are available.

An essential difference in the scaling between quantal and quasiclassical number of states should be noted. A quantum computer operating on an n level system, (a qubit is $n = 2$) and using p addressable states will scale as n^p to be contrasted with the p^n scaling that we achieve.

5 Linear sequential machines

A sequential machine is defined [4,13,24] in terms of how, in any given step, the current input and the present state (vector) of the machine determine the output and the next state of the machine. Machine time τ is measured by integer values where one unit is one step in the operation of the machine. As shown by the examples in equation (4) it is convenient to represent the input $\mathbf{i}(\tau)$, the state $\mathbf{q}(\tau)$ and the output $\mathbf{z}(\tau)$ as vectors. The machine is a finite state machine because the state vector $\mathbf{q}(\tau)$ has a finite number, n, of components. The values of components of each column vector are selected from the Galois field, namely the values are integers modulo p. The correspondence with the physical stochastic process is made by taking the dimension of the vectors to be equal to the number of levels of the physical machine, i.e., the number, n, of states coupled by the kinetic scheme, cf. equation (2). The entries in the state vector are established by the procedure of affine rationalization as discussed above.

The three operations of a linear logic circuit are multiplying by a constant, addition and a delay element that steps the value of the integer-valued time τ. All these operations are performed modulo p. Since the operations are linear, a linear sequential machine obeys the law of superposition, like a linear filter, of which it is the digital analog. The equations for the next state $\mathbf{q}(\tau+1)$ and the output $\mathbf{z}(\tau)$ can be written in matrix form. For the special case of a closed system, meaning that there is no external input, these equations are

$$\mathbf{q}(\tau+1) = \mathbf{T}\mathbf{q}(\tau), \quad \mathbf{z}(\tau) = \mathbf{C}\mathbf{q}(\tau) \quad \text{no input.} \qquad (5)$$

The matrix $\mathbf{T}$ is the state transition matrix and $\mathbf{C}$ is state-output matrix. The dimensions of the matrices in equation (5) must conform to the dimensions of the input and of the state vectors. For example, the state transition matrix $\mathbf{T}$ must be n-by-n. For a linear system the matrices $\mathbf{T}$ and $\mathbf{C}$ are given and are independent of the state of the machine.

A linear sequential machine operated as in equation (5) is said to be in an autonomous mode. Physically it means that once the initial state of the machine is specified there is no further input. The corresponding stochastic process operates as a closed system and it is the change of state that is experimentally being monitored. Both the physical process and the machine can also (linearly) respond to an external input that is a function of time.

In this paper, we limit consideration to the autonomous mode of a sequential finite state machine for which

$$\mathbf{q}(\tau+1) = \mathbf{T}\mathbf{q}(\tau). \qquad (6)$$

It is to be emphasized however that it is physically possible to apply an input during the time evolution and/or to monitor an output (e.g., the IR fluorescence from the excited, $\nu \geq 1$, vibrational states). It is only in this preliminary report that we limit considerations to the autonomous response.

6 The state transition matrix: physics and logic

It is now necessary to distinguish between the physical and logical definitions of the state transition matrix $\mathbf{T}$. The matrix $\mathbf{T}$ is defined in equations (5) or (6) as the matrix that is to operate on the logical states of the machine, states that we represent as vectors $\mathbf{q}(\tau)$. The components of these logical states are integers modulo p. In order that the next logical state is also defined over the same finite field the entries of the logical state transition matrix $\mathbf{T}$ are integers modulo p and the addition and multiplication as required by equations (5) or (6) are also modulo p. In doing such computations it is useful to take advantage of the result that

$$\left(\left(\tilde{\mathbf{T}} \bmod p\right)(\mathbf{P} \bmod p)\right) \bmod p = \left(\tilde{\mathbf{T}}\mathbf{P}\right) \bmod p. \qquad (7)$$

The physical machine works with probability vectors $\mathbf{P}(t)$ whose components are the fraction of molecules in the different levels that we can monitor. We use the notation $\tilde{\mathbf{T}}$ for the state transition matrix for the physical vector states where the tilde identifies the physical matrix. To establish a correspondence with the logic equation (6), the physical matrix $\tilde{\mathbf{T}}$ is to act for a time interval corresponding to the step of the logic machine

$$\mathbf{P}(\tau+1) = \tilde{\mathbf{T}}\mathbf{P}(\tau). \qquad (8)$$

The matrix $\tilde{\mathbf{T}}$ has dimension n by n where n is the number of physical levels of the system. It is obtained from equation (3) as

$$\tilde{\mathbf{T}} = \exp(\mathbf{A}\delta t) \qquad (9)$$

where δt is the physical time step that corresponds to one logical step of the machine. For the physical and logical transition matrices to be conformable the finite number n of internal states of the logic machine is the same as the number n of levels of the physical system. From equation (9) the elements of the physical matrix $\tilde{\mathbf{T}}$ are non-negative numbers in the range 0 to 1. On the other hand the definition of a sequential machine, requires that the matrix elements of the logic matrix $\mathbf{T}$ are integers modulo p where, as discussed above, p is a prime number. So the logical $\mathbf{T}$ and physical $\tilde{\mathbf{T}}$ cannot be the same matrix. To determine these matrices for a machine based on a physical system that is described master equation we first evaluate the physical $\tilde{\mathbf{T}}$ matrix and then, in a straightforward procedure, transcribe it to the logical state-transition matrix $\mathbf{T}$ by the procedure of affine rationalization, with an explicit example provided below.

In a closed system, which is what we are concerned with when discussing the autonomous response, the probability vectors $\mathbf{P}(t)$ are normalized. This immediately requires that the sum of the elements in any column of the physical matrix must be unity. Below we discuss the rounding-off of the elements of the $\tilde{\mathbf{T}}$ matrix. Also after rounding we require that the column sum remains 1 because this is what insures that probability is conserved at every cycle of the machine. Because this conservation is so central to us we next prove the well-known requirement on the column sum. Say we are at the machine time τ so that the physical vector state is the normalized $\mathbf{P}(\tau)$. The state after the next cycle of the machine is $\mathbf{P}(\tau+1)$ and it is determined by equation (8). To show that if the column sums of $\tilde{\mathbf{T}}$ are 1 also the next state is normalized we compute the sum of the components of $\mathbf{P}(\tau+1)$

$$\begin{aligned}\sum_{\nu=0}^{n-1} P_\nu(\tau+1) &= \sum_{\nu=0}^{n-1}\sum_{\mu=0}^{n-1} \tilde{T}_{\nu\mu} P_\mu(\tau)\\ &= \sum_{\mu=0}^{n-1} P_\mu(\tau)\sum_{\nu=0}^{n-1}\tilde{T}_{\nu\mu} = \sum_{\mu=0}^{n-1} P_\mu(\tau).\end{aligned}$$

To relate the physical and logical machines we need to transcribe the probability vectors $\mathbf{P}(t)$ to the logical vectors whose components are integers modulo p. At the same time we need to relate the physical state transition matrix $\tilde{\mathbf{T}}$ to the logical matrix $\mathbf{T}$.

We propose to use 'binning' via the affine rationalization procedure as a way of transcribing the physical vector states of the machine to the logical vector states of the machine. In the same way we transcribe the elements of the physical matrix $\tilde{\mathbf{T}}$, elements that are non-negative and are in the interval 0 to 1, to obtain the logic matrix $\mathbf{T}$. The procedure is implemented in two steps. First we determine a rational approximation that recovers a fraction to within a specified number of digits. Then, to go to logic states we express all rational fractions modulo 5.

After k steps the logical state of the machine for an autonomous (no input), mode of operation is given by a k-fold iteration of equation (6). When $\mathbf{q}(0)$ is the initial state of the machine

$$\mathbf{q}(k) = \mathbf{T}^k \mathbf{q}(0). \tag{10}$$

7 Implementation of a linear sequential machine by a 3-level Landau-Teller model

In order to keep the arithmetic relatively easy to follow the implementation of the Landau-Teller transition matrix is on a three level system, $\nu = 0$, 1 and 2. The structure of the matrix $\mathbf{A}$ (see Eq. (2) above) is:

$$\mathbf{A} = \kappa\begin{pmatrix} -\exp(-\theta) & 1 & 0\\ \exp(-\theta) & -(1+2\exp(-\theta)) & 2\\ 0 & 2\exp(-\theta) & -(2+3\alpha\exp(-\theta))\end{pmatrix}. \tag{11}$$

This matrix $\mathbf{A}$ is of rank 3, all its eigenvalues are negative and when there is no loss $\alpha = 0$ and the highest eigenvalue is zero. The eigenvector corresponding to the eigenvalue zero is the vector of populations at equilibrium.

The matrix $\tilde{\mathbf{T}} = \exp(\mathbf{A}\delta t)$ is specified by two parameters, the value of $\exp(-\theta)$, that depends on the reduced temperature, and the value of the unit time interval. The matrix quoted in equation (13) below uses $\exp(-\theta) = 0.05$ and $\kappa\delta t = 0.7$.

Lastly we need to choose the precision to within which the machine operates. This is specified by the value of the prime integer p. It needs to be large enough that the elements are not too much rounded off so that the physics of the process is preserved. $p = 5$ appears to be a small yet acceptable value. The physical matrix $\tilde{\mathbf{T}} = \exp(\mathbf{A}\delta t)$ with $\mathbf{A}$ as above with elements rounded to two significant figures is

$$\tilde{\mathbf{T}} = \begin{pmatrix} 0.98 & 0.49 & 0.25\\ 0.02 & 0.49 & 0.49\\ 0 & 0.02 & 0.26\end{pmatrix}. \tag{12}$$

It is transcribed to the logical form by first applying an affine rationalization to each (normalized) column of the matrix leading to the rational fraction form for the physical matrix

$$\tilde{\mathbf{T}} = \begin{pmatrix} 1 & 1/2 & 1/4\\ 0 & 1/2 & 1/2\\ 0 & 0 & 1/4\end{pmatrix}. \tag{13}$$

For either equations (12) or (13) the sum of each column of the $\tilde{\mathbf{T}}$ matrix is unity. As a check of the procedure of rounding-off one can compute the matrix $\tilde{\mathbf{T}}$ to three significant figures and applied an affine rationalization to one significant figure. This procedure recovers equation (13). The procedure guarantees that the resulting logical $\mathbf{T}$ matrix has its column sum as unity. Then, if we operate with the logical matrix on a vector normalized to 1 mod 5, we keep normalization.

The logical $\mathbf{T}$ matrix is the physical transition matrix modulo 5. Making use of $2\times3 = 1$ mod 5, $4\times4 = 1$ mod 5 and $2 \times 4 = 3$ mod 5, equation (13) yields

$$\mathbf{T} = \left(\tilde{\mathbf{T}}\right) \bmod 5 = \begin{pmatrix} 1 & 3 & 4 \\ 0 & 3 & 3 \\ 0 & 0 & 4 \end{pmatrix}. \tag{14}$$

For higher values of p than $p = 5$ one can check the arithmetic using the Mod and PowerMod commands of for example the Mathematica 5.1 software [20].

The physical equilibrium state is the eigenvector of $\tilde{\mathbf{T}}$ that has the eigenvalue 1. Written in transpose form the equilibrium state vector of the matrix $\tilde{\mathbf{T}}$ is found to be $(1, 0, 0)$. We have computed the matrix for a not high temperature, $\exp(-\theta) = \exp(-h\nu/kT) = 0.05$, the partition function $Q = 1+\exp(-\theta)+\exp(-2\theta) \simeq 1$ is rounded off to unity and the rounded-off Boltzmann vibrational distribution for three vibrational states has all the population in the state $\nu = 0$. In order that we do not at long times lose the population in the higher vibrational states by round-off we need to take the value of $\exp(-\theta)$ to be far larger. This implies operating at higher temperatures and then it does not make physical sense to truncate the population vector to three levels. For a harmonic system the equilibrium population in level ν is $\exp(-\nu\theta)/Q$. So no matter by how much we decrease θ, there will be higher vibrational states whose equilibrium population will be small.

Any initial population vector that is not an equilibrium state upon repeated action by the $\tilde{\mathbf{T}}$ matrix will approach equilibrium. For example, starting from the state most in disequilibrium (0, 0, 1), the sequence of steps of action by the transition matrix $\tilde{\mathbf{T}}$ of equation (13) is

$$\begin{pmatrix} 0 \\ 0 \\ 1 \end{pmatrix} \xrightarrow{\tilde{\mathbf{T}}} \begin{pmatrix} 1/4 \\ 1/2 \\ 1/4 \end{pmatrix} \xrightarrow{\tilde{\mathbf{T}}} \begin{pmatrix} 9/16 \\ 6/16 \\ 1/16 \end{pmatrix} \xrightarrow{\tilde{\mathbf{T}}} \begin{pmatrix} 49/64 \\ 14/64 \\ 1/64 \end{pmatrix} \xrightarrow{\tilde{\mathbf{T}}} \begin{pmatrix} 225/256 \\ 30/256 \\ 1/256 \end{pmatrix}. \tag{15}$$

We actually do not need to go on because to the specified round-off precision the population has reached equilibrium.

What we now show is a concrete example of the transcription between the physical system and the logic machine. Specifically we show that the sequence of logic states obtained by action of the logic $\mathbf{T}$ matrix on an initial logic state vector of the machine is the same as the set of logic state obtained by transcribing the physical state vectors as given in equation (15) to logic state vectors. This equivalence is guaranteed by the rules of modular arithmetic and specifically the definition of an inverse and the rule given in equation (7). Even so, the result is rather startling and the reason we chose the low value of 5 for the modulo and used vectors of only three states is to enable the reader to check our arithmetic by hand. Modulo 5, 4 is the inverse of 4, $4 \times 4 = 1$, so equation (15) mod 5 generates the sequence of vectors, that are, as expected, normalized mod 5

$$\begin{pmatrix} 0 \\ 0 \\ 1 \end{pmatrix} \xrightarrow{\mathbf{T}} \begin{pmatrix} 4 \\ 3 \\ 4 \end{pmatrix} \xrightarrow{\mathbf{T}} \begin{pmatrix} 4 \\ 1 \\ 1 \end{pmatrix} \xrightarrow{\mathbf{T}} \begin{pmatrix} 1 \\ 1 \\ 4 \end{pmatrix} \xrightarrow{\mathbf{T}} \begin{pmatrix} 0 \\ 0 \\ 1 \end{pmatrix}. \tag{16}$$

One can easily check that the same sequence is obtained by acting on the initial vector with the logic $\mathbf{T}$ matrix of equation (14).

Unlike the physical machine where the vector states evolve towards equilibrium, the logic states recur. This has to be so since there are only a finite number, p^n, logic states. Therefore after some finite number K of logic cycles equation (10) must return us to the starting point. In our example there are 125 logic states but the state $(0, 0, 0)$ does not change under $\mathbf{T}$. Therefore $K = 124$ or a divisor of 124. From equation (16) we see that for our machine there is a cycle of length 4. There is also a cycle of length 2, for example

$$\mathbf{T}^2 \begin{pmatrix} 2 \\ 3 \\ 1 \end{pmatrix} = \begin{pmatrix} 1\;2\;4 \\ 0\;4\;1 \\ 0\;0\;1 \end{pmatrix} \begin{pmatrix} 2 \\ 3 \\ 1 \end{pmatrix} = \begin{pmatrix} 2 \\ 3 \\ 1 \end{pmatrix}. \tag{17}$$

While the iteration of the physical machine as specified by equation (8) can go on and on producing new physical states, beyond a certain point these are not regarded by the machine as new states. The reason is the keeping numbers to modulo p and the round-off that it implies. After K iterations the physical states become indistinguishable modulo p. We can see that by comparing the 0th and 4th physical states in equation (15).

8 Towards parallelism: the companion matrix

The linearity of the machine we discuss means that it can operate on a linear combination of states and the result is a linear combination of the operation on the individual states. Specifically, if there are n physical levels so that the state vectors $\mathbf{P}(t)$ are n dimensional, there are n independent vectors that provide a basis meaning that any state vector $\mathbf{P}(t)$ can be expanded as a linear combination of n components

$$\mathbf{P}(t) = \sum_{j=1}^{n} p_j(t)\mathbf{e}_j. \tag{18}$$

Or, in component form

$$P_\nu(t) = \sum_{j=1}^{n} p_j(t)(\mathbf{e}_j)_\nu. \tag{19}$$

The action of $\tilde{\mathbf{T}}$ on a physical vector is thus a linear combination of its action on the basis vectors $\mathbf{e}_j, j = 1, \ldots, n$. The n weights p_j can be computed by taking the n scalar products $\mathbf{e}_j^T \cdot \mathbf{P}(t)$ where the superscript T denotes the transpose. If the basis vectors can be taken to be orthonormal we have the familiar simple result $p_j(t) = \mathbf{e}_j^T \cdot \mathbf{P}(t)$. If we take the basis vectors to be eigenvectors of the $\mathbf{T}$ matrix then, because the transition matrix is not symmetric but does satisfy detailed balance one needs to work with the biorthogonal right and left eigenvectors [15]. Then, if $\mathbf{f}_j$ is the jth left eigenvector (and as such it is necessarily a row vector), $p_j(t) = \mathbf{f}_j \cdot \mathbf{P}(t)$. By the properties of congruence, modulo p the physical matrix in rational form and logic matrix have the same set of eigenvectors and eigenvalues.

Equation (19) defines what is parallelism: monitoring the temporal evolution of the population of a particular physical level is equivalent to n logic simultaneous computations. Each individual computation is the sequence of vectors generated by the action of the logic matrix on a basis vector. For the logic matrix given by equation (14) the three eigenvalues are 4, 3 and the logic right and left eigenvectors are

$$\underbrace{\begin{pmatrix}3\\4\\3\end{pmatrix}, \ \begin{pmatrix}0\ 0\ 3\end{pmatrix};}_{\text{eigenvalue 4}} \ \underbrace{\begin{pmatrix}3\\2\\0\end{pmatrix}, \ \begin{pmatrix}0\ 4\ 3\end{pmatrix};}_{\text{eigenvalue 3}} \ \underbrace{\begin{pmatrix}1\\0\\0\end{pmatrix}, \ \begin{pmatrix}1\ 1\ 1\end{pmatrix}.}_{\text{eigenvalue 1}} \quad (20)$$

The format in equation (20) is chosen such that if the coma is replaced by a matrix product then one gets the projection matrices on the three eigenvalues.

Progress in experimental technology allows us to go beyond parallelism to massive parallelism. This is made possible by the experimental ability, using an array detector, to simultaneously detect the population of several, up to n, physical levels. This means that it is possible to perform up to n^2 computations in parallel. Even beyond this it is a reality to think of the application of time resolved multidimensional vibrational spectroscopy [25] to logic.

There are at least three obvious choices for a set of basis vectors. One possibility as discussed above are the eigenvectors of the $\tilde{\mathbf{T}}$ matrix. These are necessarily the same as the eigenvectors of the $\mathbf{A}$ matrix. These vectors have been discussed in detail [1,2,15] and we hope to return to their concrete application as a computational basis in the present context in a forthcoming paper. A second choice is the, so-called, Krylov basis as used extensively in Lanczos based applications [26]. These vectors are defined by starting with some vector $\mathbf{v}$ and generating the n independent vectors $\mathbf{v}, \tilde{\mathbf{T}}\mathbf{v}, \tilde{\mathbf{T}}^2\mathbf{v}, \ \ldots$

A third and related choice of the basis used to describe the computation is the, so-called, standard basis ε_j, $j = 1, \ldots, n$ where the components of ε_j are all zeros except for a unity in position j. This is the basis that we adopt in order to discuss the logic circuit of our machine. This choice is a natural one because the $\mathbf{T}$ matrix for the Landau-Teller process has not degenerate eigenvalues. Therefore the characteristic polynomial $\varphi(x)$ of $\mathbf{T}$, defined as $|\mathbf{T} - x\mathbf{I}| = 0$ has n distinct roots. Given $\mathbf{T}$ we compute the polynomial and write it as

$$\varphi(x) = x^n - a_{n-1}x^{n-1} - \ldots - a_1 x - a_0. \quad (21)$$

From the coefficients of the characteristic polynomial as written in equation (21) one constructs the companion matrix $\mathbf{T}_c$

$$\mathbf{T}_c = \begin{pmatrix} 0 & 1 & 0 & \cdots & 0 \\ 0 & 0 & 1 & 0\cdots & 0 \\ & & \ddots & \ddots & \\ 0 & & \cdots & 0 & 1 \\ a_0 & a_1 & a_2 & \cdots & a_{n-1} \end{pmatrix}. \quad (22)$$

The matrix $\mathbf{T}$ and its companion matrix are related by a similarity transformation: $\mathbf{T} = \mathbf{S}^{-1}\ \mathbf{T}_c\ \mathbf{S}$. Therefore rather than regarding $\mathbf{T}$ as the state transition matrix for the states $\mathbf{q}$ of the machine, we can regard $\mathbf{T}_c$ as the state transition matrix for the states $\hat{\mathbf{q}} = \mathbf{S}^{-1}\mathbf{q}$ of the machine. The companion matrix plays a key role because $\mathbf{T}_c$ is the transition matrix for the machine working with the computational basis that we chose. What this means is that the action of the machine is represented as the matrix equation

$$\begin{pmatrix} \hat{q}_1(k+1) \\ \hat{q}_2(k+1) \\ \vdots \\ \hat{q}_{n-1}(k+1) \\ \hat{q}_n(k+1) \end{pmatrix} = \begin{pmatrix} 0 & 1 & 0 & \cdots & 0 \\ 0 & 0 & 1 & 0\cdots & 0 \\ & & \ddots & \ddots & \\ 0 & & \cdots & 0 & 1 \\ a_0 & a_1 & a_2 & \cdots & a_{n-1} \end{pmatrix} \begin{pmatrix} \hat{q}_1(k) \\ \hat{q}_2(k) \\ \vdots \\ \hat{q}_{n-1}(k) \\ \hat{q}_n(k) \end{pmatrix}. \quad (23)$$

In component form this reads

$$\begin{aligned} \hat{q}_1(k+1) &= \hat{q}_2(k) \\ \hat{q}_2(k+1) &= \hat{q}_3(k) \\ &\vdots \\ \hat{q}_{n-1}(k+1) &= \hat{q}_n(k) \\ \hat{q}_n(k+1) &= a_0\hat{q}_1(k) + a_1\hat{q}_2(k) + \ldots \\ &\quad + a_{n-2}\hat{q}_{n-1}(k) + a_{n-1}\hat{q}_n(k). \end{aligned} \quad (24)$$

We are now in a position to construct an actual logical circuit. In the literature of linear machines the circuit we introduce is known as the 'canonical form' [13,17] of the linear machine. What we do not do in this paper is to connect to the theory of factorization of polynomials. The motivation for making this connection is that factorization is regarded as a 'hard' computational problem. Yet we believe that we know how to make the connection and we intend to do so in a separate paper where we allow for input to the machine.

9 A logic circuit

The three operations of a linear logic circuit are multiplying by a constant and addition both of which are performed modulo p and shifting the cycle time τ by one or more units. All these operations are linear.

To write down the circuit we make the convention that the arrows are the order of operations. With reference to equations (24) we see that after one cycle $\hat{q}_1(k+1) = \hat{q}_2(k)$ and similarly for all others up to $n-1$. Each of these actions is known technically as a (unit) delay [4,16,24]. Therefore in one cycle the circuit shifts the content of the memory cells by one unit, an action known as a 'shift register' [4,13,17,24]. For the last memory cell the updated value is obtained by multiplication of the content of each memory entry and addition of the results $\hat{q}_n(k+1) = a_0\hat{q}_1(k)+a_1\hat{q}_2(k)+\ldots+a_{n-2}\hat{q}_{n-1}(k)+a_{n-1}\hat{q}_n(k)$. The sum is fed back into the last memory cell. This is schematically shown in Figure 1. In summary of Figure 1, the companion matrix is the (so-called, canonical) representation of the sequential linear machine and the corresponding circuit is as shown. Each of the n physical levels plays the role of a memory unit for the machine, capable of storing an integer modulo 5. Each memory unit is shown as a delay element in Figure 1. The delay sequence is drawn in Figure 1 as an arrow leading from $\hat{q}_j(k)$ to $\hat{q}_{j-1}(k)$. The contents of the memory cells for $j = 1,\ldots,n-1$ are shown multiplied by the required coefficient, added and fed back to the last memory cell. Only the nth delay element is affected by the feedback information.

Another simple representation of the logic circuit is expressed in terms of the transpose of the companion matrix (Eq. (22)), $\mathbf{T}_c^T$,

$$\mathbf{T}_c^T = \begin{pmatrix} 0 & 0 & 0 & \cdots & a_0 \\ 1 & 0 & 0 & 0\cdots & a_1 \\ & 1 & \ddots & \ddots & \\ 0 & & \ddots & 0 & a_{n-2} \\ 0 & 0 & 0 & 1 & a_{n-1} \end{pmatrix}. \tag{25}$$

When we write down the action $\mathbf{T}_c^T$ on the current state vector, as is done for $\mathbf{T}_c$ in equations (24), we see that the transpose matrix leads to a circuit known as a multi-adder shift register [24] shown in Figure 2. $\mathbf{T}_c^T$ is also related to the matrix $\mathbf{T}$ by a similarity transformation. Therefore in general, the matrix $\mathbf{T}$ can be realized either as a shift register or as a multi adder shift register.

The characteristic polynomial, $\varphi(x)$, the logic $\mathbf{T}$ matrix, equation (14), is

$$\varphi(x) = 4x^3 + 3x^2 + x + 2 \tag{26}$$

$\varphi(x)$ can be factorized as

$$\varphi(x) = (x+4)(x+3)(x+1) = (x-1)(x-2)(x-4). \tag{27}$$

The result (26) can be checked by finding first the characteristic polynomial of the physical matrix, $\varphi(x) = -x^3 + 1.75x^2 - 0.875x + 0.125$, and taking it modulo 5 recalling that, for example, $(-1) = 4 \bmod 5$, etc.

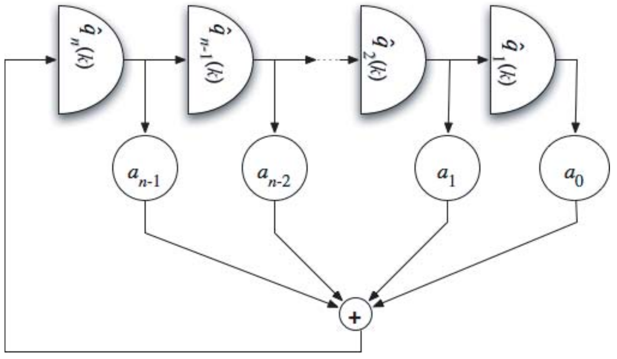

Fig. 1. A shift register circuit for the realization of the companion matrix $\mathbf{T}_c$ (Eq. (22)). The n delays, represented as a half circle, correspond to the n internal state of the linear sequential machine. The $\oplus$ means addition modulo p and the symbol $\textcircled{a_i}$ means multiplication modulo p by a_i, $i = 0,\ldots,n-1$. The arrows show the direction of operations.

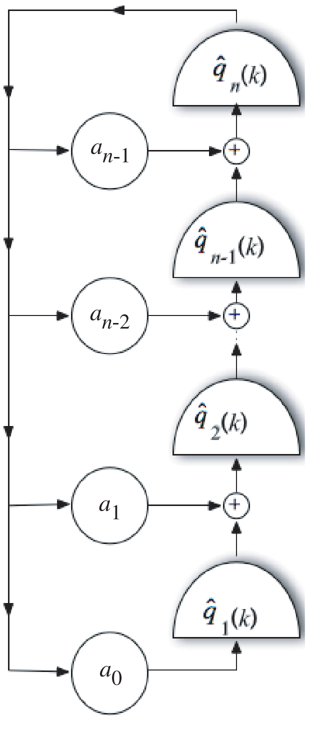

Fig. 2. The realization of the transpose of the companion matrix $\mathbf{T}_c$ (Eq. (25)) as a multi-adder shift register circuit. The symbols used are the same as in Figure 1.

To get the companion matrix, the polynomial must be expressed as in equation (21), or $\varphi(x) = x^3 - 3x^2 - x - 2$. The companion matrix, equation (25), for the 3 vibrational level relaxation process is therefore given as

$$\mathbf{T}_c = \begin{pmatrix} 0 & 1 & 0 \\ 0 & 0 & 1 \\ 2 & 1 & 3 \end{pmatrix}. \tag{28}$$

The corresponding linear machine is shown in Figure 3.

So far we asked how to express the circuit that corresponds to a given relaxation process. The limitations on what a linear machine can do are more clearly evident

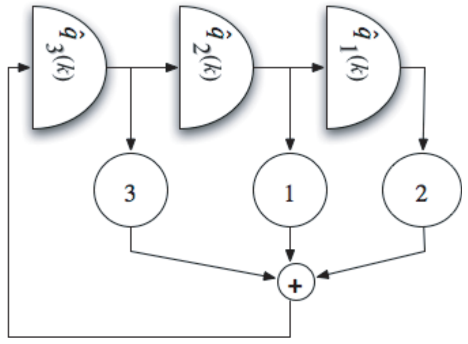

Fig. 3. The realization of the companion matrix $\mathbf{T}_c$ (Eq. (22)) derived from the rounded off physical matrix $\tilde{\mathbf{T}}$ (Eq. (13)) that describes the Landau-Teller relaxation of a truncated 3 level harmonic oscillator. The symbols used are as in Figure 1.

when we ask the complementary question, how to identify a physical process that will realize a given circuit. The circuit is fully specified when we give its companion matrix. The matrix is equivalent to a polynomial. If the eigenvalues of the relaxation process are distinct the polynomial is the characteristic polynomial and can be written in product form as in equation (27). Otherwise it is the minimal polynomial where every distinct eigenvalue appears once. It follows that there are three types of variations in the circuit that we can achieve by changing the physical system. First is the order, p, of the Galois field and that is determined by the precision within which we can determine concentrations. A one significant figure is certainly reasonable so $p = 10$ is realistic. Then there is the number n, the degree of the polynomial or the size of the companion matrix. n is equal to or smaller than the number of distinct species that participate in the process. We need to allow for smaller because it can be that not all the eigenvalues of the relaxation process are distinct. Last, for a given p and n we can control the values of the n coefficients a_i, $i = 0, \ldots, n-1$ that specify the polynomial $\varphi(x)$, equation (21) or the companion matrix, equation (22). The physicochemical relaxation process is specified by the symmetric (detailed balanced) n-by-n rate matrix $\mathbf{A}$, cf. equation (1). But it is only through the changes in the n coefficients a_i of the companion matrix that changing the nature of the relaxation process can lead to a different machine.

10 Concluding remarks

A system evolving in time under a master equation, a Markovian stochastic process, was shown to offer a physical realization of a linear sequential computing machine. The Landau-Teller process as analyzed by Montroll and Shuler is used as a concrete example. Such a system not only computes in parallel but can do doubly so, sometimes known as massively parallel. Establishing the connection with the fundamentals of linear finite state machines requires a transcription between the probabilities of physical states of the system and words composed of letters from a finite alphabet. We did so using basic notions from the algebraic theory of Galois fields. Even machine words only one letter long are sufficient to achieve doubly parallel processing because a state of the machine is equivalent to an entire probability vector of the stochastic process and the machine has an exponentially large number of memory states. Different logic circuits that implement the physics are presented.

This work was supported by the US-Israel Binational Science Foundation, BSF, Jerusalem, Israel and by the EC STREP FET-OPEN project MOLDYNLOGIC. We thank an anonymous referee for emphasizing the different scaling behavior of quantum and quasiclassical schemes.

References

1. I. Oppenheim, K.E. Shuler, G.H. Weiss, *Stochastic Processes in Chemical Physics: The Master Equation* (The MIT Press, Cambridge, 1977)
2. E.W. Montroll, K.E. Shuler, Adv. Chem. Phys. **1**, 361 (1958)
3. E.W. Montroll, K.E. Shuler, J. Chem. Phys. **26**, 454 (1957)
4. Z. Kohavi, *Switching and Finite Automata Theory* (Tata McGraw-Hill, New Delhi, 1999)
5. A.P. deSilva, Nature Mat. **4**, 15 (2005); X.F. Guo, D.Q. Zhang, G.X. Zhang, D.B. Zhu, J. Phys. Chem. B **108**, 11942 (2004); D. Margulies, G. Melman, C.E. Felder, R. Arad-Yellin, A. Shanzer, J. Am. Chem. Soc. **126**, 15400 (2004); F. Remacle, S. Speiser, R.D. Levine, J. Phys. Chem. A **105**, 5589 (2001); D. Steinitz, F. Remacle, R.D. Levine, Chem. Phys. Chem. **3**, 43 (2002); F. Remacle, I. Willner, R.D. Levine, Chem. Phys. Chem. **6**, 1 (2005); M.N. Stojanovic, D. Stefanovic, J. Am. Chem. Soc. **125**, 6673 (2003); J. Andreasson, G. Kodis, Y. Terazono, P.A. Liddell, S. Bandyopadhyay, R.H. Mitchell, T.A. Moore, A.L. Moore, D. Gust, J. Am. Chem. Soc. **126**, 15926 (2004); F.M. Raymo, Adv. Mat. **14**, 401 (2002); V. Balzani, A. Credi, M. Venturi, Chem. Phys. Chem. **4**, 49 (2003); C. Joachim, J.K. Gimzewski, A. Aviram, Nature **408**, 541 (2000)
6. A. Hjelmfelt, E.D. Weinberger, J. Ross, Proc. Natl. Acad. Sci. (USA) **88**, 10983 (1991); A. Hjelmfelt, E.D. Weinberger, J. Ross, Proc. Natl. Acad. Sci. (USA) **89**, 383 (1992); A. Hjelmfelt, J. Ross, Proc. Natl. Acad. Sci. (USA) **89**, 388 (1992)
7. R.D. Levine, *Molecular Reaction Dynamics* (Cambridge University Press, Cambridge, 2005)
8. A.E. Johnson, N.E. Levinger, P.F. Barbara, J. Phys. Chem. **96**, 7841 (1992); D.A.V. Kliner, J.C. Alfano, P.F. Barbara, J. Chem. Phys. **98**, 5375 (1993); T. Kuhne, P. Vohringer, J. Chem. Phys. **105**, 10788 (1996)
9. A.V. Davis, R. Wester, A.E. Bragg, D.M. Neumark, J. Chem. Phys. **117**, 4282 (2002)
10. C.C. Rankin, J.C. Light, J. Chem. Phys. **46**, 1305 (1967); S.R. Leone, R.J. McDonald, C.B. Moore, J. Chem. Phys. **63**, 4735 (1975)
11. C.B. Moore, in *Fluorescence*, edited by G.G. Guilbault (Dekker, New York, 1967), p. 133; H.-L. Chen, C.B. Moore, J. Chem. Phys. **54**, 4072 (1971)

12. F. Remacle, R.D. Levine, Proc. Natl. Acad. Sci. USA **101**, 12091 (2004)
13. T.L. Booth, *Sequential Machines and Automata Theory* (Wiley, New York, 1968)
14. I. Oppenheim, K.E. Shuler, G.H. Weiss, J. Chem. Phys. **50**, 460 (1969); I.N. Krieger, P.J. Gans, J. Chem. Phys. **32**, 247 (1960)
15. R.I. Cukier, K.E. Shuler, J. Chem. Phys. **57**, 302 (1972)
16. A. Gill, *Introduction to the Theory of Finite-State Machines* (McGraw-Hill, New York, 1962)
17. M.A. Harrison, *Lectures on Linear Sequential Machines* (Academic Press, New York, 1969)
18. E.W. Weisstein, *Concise Encyclopedia of Mathematics*, 2nd edn. (Chapman and Hall, London, 2002); E.W. Weisstein `http://mathworld.wolfram.com/Congruence.html` (2006)
19. A. Ekert, P. Hayden, H. Inamori, e-print `arXiv:quant-ph/0011013` (2000)
20. S. Wolfram, *The Mathematica Book*, 5th edn. (Wolfram Media, 2003)
21. M.A. Nielsen, I.L. Chuang, *Quantum Computation and Quantum Information* (Cambridge University Press, Cambridge, 2000); D. Deutsch, Proc. R. Soc. Lond. A **425**, 73 (1989); D. Deutsch, R. Jozsa, Proc. R. Soc. Lond. A **439**, 553 (1992); A. Barenco, C.H. Bennett, R. Cleve, D.P. Divincenzo, N. Margolus, P. Shor, T. Sleator, J.A. Smolin, H. Weinfurter, Phys. Rev. A **52**, 3457 (1995)
22. D.J. Nesbitt, R.W. Field, J. Phys. Chem. **100**, 12735 (1996); M. Gruebele, P.G. Wolynes, Acc. Chem. Res. **37**, 261 (2004)
23. S. Hammes-Schiffer, Acc. Chem. Res. **34**, 273 (2001); R.I. Cukier, D.G. Nocera, Ann. Rev.Phys. Chem. **49**, 337 (1998); F.M. Raymo, M. Tomasulo, Chem. Soc. Rev. **34**, 327 (2005)
24. A. Gill, *Linear Sequential Circuits* (McGraw Hill, New York, 1966)
25. J. Zheng, K. Kwak, J. Asbury, X. Chen, I. Piletic, M.D. Fayer, Science **309**, 1338 (2005); Y.S. Kim, R.M. Hochstrasser, PNAS **102**, 11185 (2005)
26. C. Lung, R.E. Wyatt, J. Chem. Phys. **99**, 2261 (1993); R.B. Lehoucq, S.K. Gray, D.H. Zhang, J.C. Light, Comp. Phys. Com. **109**, 15 (1998)