Dwight Kuo

Chaos and its computing paradigm

Recently, there has been interest in so-called alternative computing paradigms that differ from traditional silicon-based computing architectures. One such alternative is chaos computing which exploits the inherent dynamics of chaotic systems.

In everyday language, the words random and chaotic are often interchanged. However, in the mathematical sense, chaos is not unpredictable or random at all. A given chaotic system behaves in a totally predictable and fully reproducible way (given the same initial conditions). So what is the difference between a chaotic and a non-chaotic system?

Chaos has three defining characteristics. The first characteristic is sensitivity to initial conditions. If one was to input a value x into a system, we might obtain result y. However, in a chaotic system even if we only slightly perturbed x say by 0.0000001 we would obtain a completely different result from y. In fact, such trajectories originating from differing initial conditions separate exponentially fast. This outcome has been most famously stated as the so-called "butterfly effect":

> The flapping of a single butterfly's wing today produces a tiny change in the state of the atmosphere. Over a period of time, what the atmosphere actually does diverges from what it would have done. So, in a month's time, a tornado that would have devastated the Indonesian coast doesn't happen. Or maybe one that wasn't going to happen, does.
>
> (Ian Stewart, *Does God Play Dice? The Mathematics of Chaos*, pg. 141)

This result is clearly different from the ones engineers usually deal with which are typically linear and, thus, have a linear response curve.

A second characteristic of chaos is aperiodic long-term behavior. In other words, there are no fixed points, periodic orbits, or quasi-periodic orbits in the behavior of the system (as $t \to \infty$). Note that this does not mean that the time evolution of the system is random! However, this property has been exploited to generate an approximation to seemingly random numbers in so-called pseudo-random number generators. Such numbers are not truly random since if given the chaotic equation governing the behavior of the system and the *exact* initial conditions, the numbers generated are completely predictable and reproducible. Thus, the system is not really random, since a random signal requires that it be neither predictable nor reproducible.

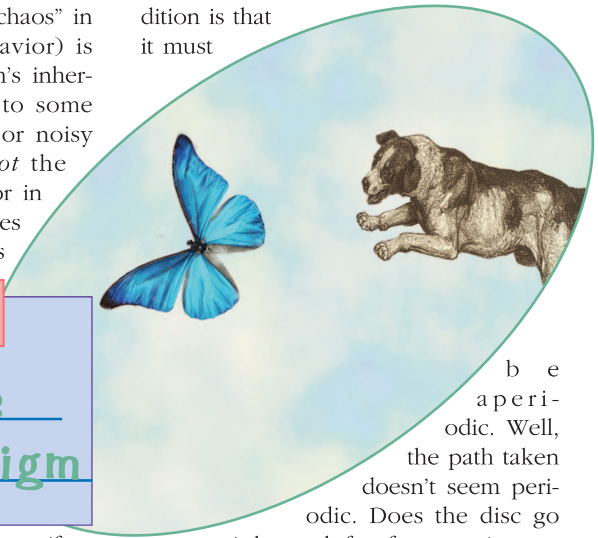
This leads to the third characteristic of a chaotic system. It is wholly deterministic. This means that the "chaos" in the system (its irregular behavior) is solely a function of the system's inherent nonlinearity as opposed to some other driving force. Random or noisy inputs or parameters are *not* the cause of the aperiodic behavior in the system. The system behaves as it does solely based on its own internal dynamics. Therefore, if we were to start a chaotic system twice with the *exact* same initial conditions, we would obtain the *exact* same result just as if we were using any other deterministic system.

Perhaps a simple example will help. If you've ever watched the popular game show, "The Price is Right," you've probably seen the game "Plinko." In this game, the contestant drops a circular disc down a surface that has pegs placed at regular intervals. The disc then seemingly follows a random path down the surface bouncing left and right off the pegs until finally it reaches the bottom (see Fig. 1) of a slot with monetary value. One potential path of the disc is shown by the solid line. So is this system a chaotic one?

Table 1

X_1	X_2	AND(X_1,X_2)
False	False	False
False	True	False
True	False	False
True	True	True

Well, the first condition is that the path of the disc be dependent on the initial conditions. If you've seen this game enough times, you'll know that even if you drop the disc at the same starting point every time with approximately the same starting velocity, the disc doesn't seem to follow the same path. This is because if you change the starting position and velocity even slightly, it will hit the pegs differently during every bounce. This difference becomes larger and larger over time until finally the path of the disc changes from the previous one.



The second condition is that it must be aperiodic. Well, the path taken doesn't seem periodic. Does the disc go right or left of a peg in any repeatable pattern in a long term predictable way? Not really.

The third condition is that the system must be deterministic. If we exclude outside perturbations, like maybe vibrations on the stage, then the system is certainly deterministic. The disc is only subject to the laws of Newtonian physics. Thus, the system has no stochastic or random behaviors. So Plinko does behave as a chaotic system.

In fact, for years scientists have observed a rich variety of behaviors in the natural world with many being chaotic, such as population dynamics, cardiac rhythms and EEG recordings of brain activity. However, most mathematical, computational and modeling tools are linear and, thus, cannot reproduce chaotic behavior (which is inherently nonlinear). The reason for this is the simplicity and significant successes of using linear system models. In contrast, chaotic nonlinear models do not have a unifying theory and are difficult to analyze and model. Thus, it has been common, especially in the engineering community to engineer away chaos in systems and models.

But with the internal processes of living systems often being chaotic (e.g. the brain and heart), there must be benefits to its use. It has been theorized that chaos provides flexibility in the performance of a system and provides a wide range of dynamic behaviors that can be utilized to improve performance. One example where chaotic unpredictability is useful is in escape from predators. Unpredictable patterns of escape behavior might give an advantage to a prey. Another example is in the ability of the brain to continually learn new concepts. It is hypothesized that this is due to the inherent chaotic dynamics in the brain. So what if it were possible to exploit such well understood physical phenomena for the purpose of computations?

William Ditto of the Georgia Institute of Technology and his collaborator, Sudeshna Sinha of the Institute of Mathematical Sciences in Madras, have proposed that useful computations can be performed from coupling chaotic elements, a process that they call "chaos computing." The general idea is to have many chaotic systems whose inputs and outputs are coupled to each other and to use a threshold to convert the output to a logical true/false.

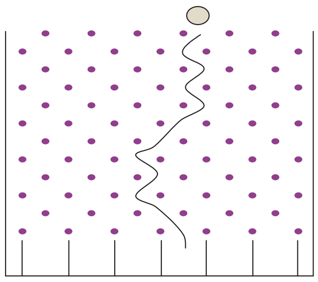


Fig. 1 The game of Plinko

Logistic map

Before we discuss the chaos-computing paradigm, we should formulate the behavior of a chaotic system. Here, we use the Logistic Map, a very simple example of a chaotic system originally used as a simple population (demographic) model. The system is defined as follows:

$$x_{n+1} = \lambda x_n(1 - x_n)$$

where λ is a positive parameter and x is between 0 and 1. The first term, x, denotes the growth due to reproduction that is proportional to the size of the current population. The second term, $(1 - x)$ denotes the loss in the population from starving to death (due to the environment and poor government policy). Therefore, the population decreases at a rate proportional to the theoretical "carrying capacity" (in this case "1") of the environment minus the current population size (x).

The parameter λ controls the combined rate of reproduction and starvation. Its value largely determines the behavior of the system. Figure 2 shows the logistic map's sensitivity to initial conditions for $\lambda = 4.0$. The system is run for 50 iterations with slightly different initial conditions (x_0). It can be seen that system runs approximately the same for a few iterations. However, at about iteration 23, the trajectories of the systems begin to diverge. By iteration 35, the trajectories of the three systems bear little resemblance to each other.

Another way to display the behavior of the logistic map is with a bifurcation diagram. The bifurcation diagram of the logistic map is shown in Fig. 3. This shows the behavior of the system for various values of the parameter λ. For many values of $\lambda \leq 4$ and $\lambda > 3.57$, the logistic map displays chaotic behavior. The dots represent points which have been visited during 100 iterations of the logistic map for a given initial condition. Regions of the plot that are broadly filled in are areas where the system behaves chaotically. In fact, the bifurcation diagram is itself a fractal.

Computing with chaos

Now that we have a definition of chaos, and an example of a chaotic system, how do we use this to perform computations? The following process describes how a chaotic element may be used to perform a computation:

1. Initialize the state of the system and the external inputs (define the initial conditions)

$x = x_0 + X_1 + X_2$ for AND, OR, XOR, and NAND gate

$x = x_0 + X$ for NOT gate

X_1 and X_2 define the input values to the system that are normalized to [0,1].

2. Update the chaotic system (run the system)

$x_{n+1} = f(x_n)$

where $f(x)$ is defined as a chaotic function.

3. Obtain the output of the system by utilizing the threshold

$$Z = \begin{cases} \text{false} & \text{if } f(x) \leq x^* \\ \text{true} & \text{if } f(x) > x^* \end{cases}$$

where Z is the logical output of the chaotic element, and x^* is the threshold.

Note that Z is a logical value (true/false). The actual numerical value of the chaotic system that corresponds to logical true is:

$$\delta = f(x) - x^*$$

This is best illustrated by an example. The AND gate ($Z = \text{AND}(X_1, X_2)$) has four possible input combinations, (0,0), (0,δ), (δ, 0), and (δ,δ) which yield the results 0, 0, 0, and δ respectively which is replaced by logical true in the truth table. In order for a chaotic element to implement the AND gate, it must satisfy this input output relationship.

Thus, for the first combination, $$ (false,false) $\rightarrow$ false, the system is thus initialized: $x = x_0 + 0 + 0$. We require that the output of the system be zero (i.e. Z = false). To accomplish this, we require $f(x_0) \leq x^*$ in step 3.

For the second and third combination of inputs, either X_1 or X_2 is 0 (logical false) while the other value is δ. Therefore, the initial value of the system is $x_0 + \delta$. Since the output value of the system is true, we require $f(x_0 + \delta) \leq x^*$For the final input combination, both X_1 and X_2 are δ. Therefore, the initial value of x is $x + 2\delta$. We must then have $f(x_0 + 2\delta) > x^*$ and also $\delta = f(x + 2\delta) - x^*$. We can combine these two requirements to obtain $f(x_0 + 2\delta) - x^* = \delta$.

Now we know the conditions that must be satisfied simultaneously for a given chaotic element to implement the AND gate:

1. $f(x_0) \leq x^*$
1. $f(x_0 + \delta) \leq x^*$
3. $f(x_0 + 2\delta) - x^* = \delta$.

Similar constraints can be found for other logical operations such as NAND, OR and XOR. In addition, if the value of δ is made to be a common positive constant between these logical operations, the output from one gate can be inputted directly into another. This brings about the possibility that we could couple such elements together to perform useful computations.

Using the logistic map as our chaotic element, by for example selecting $x_0 = 0$ and $x^* = 3/4$, the conditions presented would be:

1. $f(x_0) = f(0) = 0 = 3/4 = x^*$
2. $f(x_0 + \delta) = f(1/4) = 3/4x^*$
3. $f(x_0 + 2\delta) = f(1/2) - 3/4$
$= 1 - 3/4 = 1/4 = \delta$

The parameter values for implementing other logical gates using the logistic maps are shown in the Operations table. Thus by simply changing the threshold value of a chaotic element, it can be made to behave as a logical gate. Of course, we are not limited to using the logistic map. Any chaotic system can be substituted such as the Lorenz, Duffing, or Rössler equations.

The advantages

Eventually, fundamental physical limits will be reached for squeezing even more transistors on a chip. Either it will be technologically impossible or prohibitively expensive. Because of this eventuality, scientists and engineers have been considering alternatives to the traditional silicon based architectures for microchips. This is one of the reasons that schemes such as quantum computing, DNA computing and, now, chaos computing are being investigated.

But why would computing with chaotic elements be any more advantageous than using traditional silicon-based Boolean gates? The key lies in the way chaotic elements are made to behave like logic gates. By changing the threshold value, a given chaotic element can act like a *different* logic element. A completely different circuit could be implemented on the same hardware. But how is this different from standard field programmable gate arrays (FPGAs)? The difference is that the configuration of chaotic elements can be changed simply by changing the threshold.

In practice, this could be accomplished extremely fast, so fast in fact that it may be possible to completely change the function of a given circuit with each tick of the Central Processing Unit's (CPUs) clock. The idea is to try and exploit the complicated inherent dynamics of chaotic systems for computation instead of trying to eliminate them. This could allow computations significantly faster than conventional logical circuits.

In addition, many current devices and systems display chaotic properties. Examples of such systems are lasers, electrical circuits and neurons in the brain. Thus, a wide variety of candidates exist for use as chaotic elements in a chaos computing scheme. In fact, rudimentary chaos computers have been constructed using resistors and capacitors, a pair of leech neurons placed on a microchip, and lasers.

Table 2

Operation	AND	OR	NAND	NOT	XOR
x_0	0	1/8	3/8	_	1/4
x^*	3/4	11/16	11/16	_	3/4

In the specific case of the neurons, they were extracted from leeches, placed in a Petri dish and allowed to form synapses of their own accord and connected through micro-electrodes to a computer. By stimulating the neurons with specific minute electrical inputs and recording the responses, it was possible to get the neurons to perform rudimentary calculations such as the addition of two numbers. In this way, by exploiting the inherent chaos in such devices, we may be able to obtain computations "for free" simply by coupling them together and applying a threshold.

The chaos computing architecture can also easily be parallelized either by executing many "processors" in parallel, or by utilizing the high dimensionality (number of variables) in some chaotic systems. By simply using a device with higher dimensions of chaos, the ideas presented on the logistic map can be extended so as to perform computations in the separate dimensions. (The logistic map has one dimension but other chaotic systems have many more.)

Since research in chaos computing is only in its infancy it is as yet not appropriate to debate the optimality of computing using chaos. However, unlike other alternative computing paradigms such as DNA computing and quantum computing which are designed to handle specific problem instances, chaos computing presents a framework which one day might be appropriate for a general purpose machine. Current work in this area has been focused on increasing the speed of such computations which depend on the particular updating of the chaotic system used and the quick calculation and setting of the aforementioned thresholds.

Read more about it

• Sinha, S., Munakata, T. and Ditto, W.L. (2002) Parallel computing with extended dynamical systems. *Physical Review E*, Vol. 65, 036214

• Munakata, T., Sinha, S., and Ditto, W.L. (2002) Chaos Computing: Implementation of Fundamental Logical Gates by Chaotic Elements. *IEEE Transactions on Circuits and Systems-I: Fundamental Theory and Applications*, Vol. 49(11), 1629-1633

• Strogatz, S. H. (1994) *Nonlinear dynamics and chaos: With applications to physics, biology, chemistry, and engineering*. Reading, MA: Perseus Books, Cambridge MA

About the author

Dwight Kuo obtained a Bachelor's of Applied Science in Systems Design Engineering at the University of Waterloo and is currently pursuing postgraduate studies in Computational Sciences at Memorial University of Newfoundland.

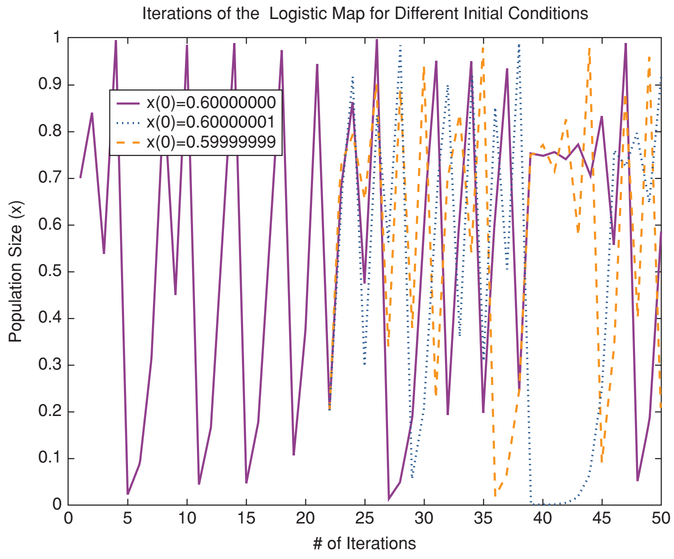


Fig. 2 Sensitivity to initial conditions in the logistic map

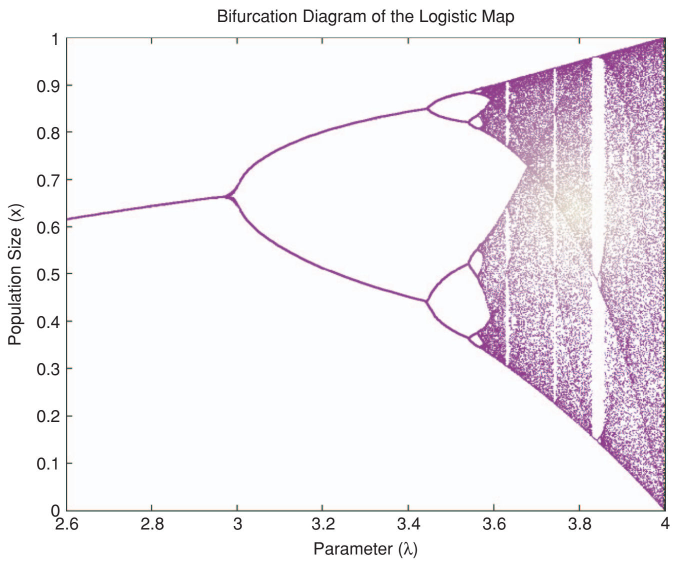


Fig. 3 Bifurcation diagram for the logistic map