Discrete Dynamics Seminar Presentation

Thursday, September 15, 2011
10:28 PM

DDS_seminar1

Inserted from: <file://C:\Users\Public\Documents\VCU\Seminars\DDS_seminar1.pdf>

Δ

DISCRETE DYNAMICS SEMINAR
DEPARTMENT OF MATHEMATICS
VIRGINIA COMMONWEALTH UNIVERSITY

Discrete Dynamical Systems and Difference Equations: A Modern Perspective

Hassan Sedaghat

September 16, 2011

Δ

Classically, a *discrete dynamical system* (DDS) consists of the following components:

- An *underlying space* S such as the set $\mathbb{R}$ of all real numbers or the binary set $\mathbb{Z}_2 = \{0, 1\}$

- A set of *variables* $x_{1,n}, x_{2,n}, \ldots, x_{m,n}$ taking values in S for each $n = 1, 2, 3, \ldots$

- A system of *difference equations* that tracks changes in the status of the variables

$$\begin{aligned} x_{1,n} &= f_1(x_{1,n-1}, x_{2,n-1}, \ldots, x_{m,n-1}) \\ x_{2,n} &= f_2(x_{1,n-1}, x_{2,n-1}, \ldots, x_{m,n-1}) \\ &\vdots \\ x_{m,n} &= f_m(x_{1,n-1}, x_{2,n-1}, \ldots, x_{m,n-1}) \end{aligned}$$

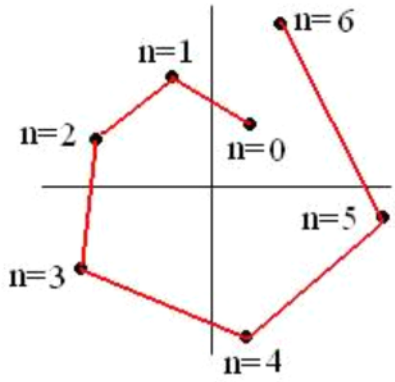
where $f_1, f_2, \ldots, f_m$ are a set of given *update functions*.

Δ

Given any set of *initial status values* $(x_{1,0}, x_{2,0}, \ldots, x_{m,0})$ the above system updates the status of the variables from "*time*" $n = 0$ to $n = 1$

$$\begin{aligned} x_{1,1} &= f_1(x_{1,0}, x_{2,0}, \ldots, x_{m,0}) \\ x_{2,1} &= f_2(x_{1,0}, x_{2,0}, \ldots, x_{m,0}) \\ &\vdots \\ x_{m,1} &= f_m(x_{1,0}, x_{2,0}, \ldots, x_{m,0}) \end{aligned}$$

The new m-tuple $(x_{1,1}, x_{2,1}, \ldots, x_{m,1})$ can then be inserted into the system to update status further to $n = 2$ and continuing this process for higher values of n.



- This iteration process generates an *orbit* (or *solution*) of the DDS

$$\{(x_{1,n}, x_{2,n}, \ldots, x_{m,n}) : n = 0, 1, 2, 3, \ldots\}$$

- The product space S^m of all m-tuples $(s_1, s_2, \ldots, s_m)$ which contains all orbits of the DDS is called its *state-space*. This the discrete analog of "phase space" in differential equations.
- Each m-tuple in S^m is called a *state* of the DDS.

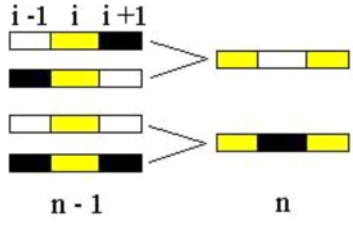
Δ

A cellular automata example:

- $S = \mathbb{Z}_2 = \{0, 1\}$; the state-space S^m has 2^m tuples.
- m variables $x_{1,n}, x_{2,n}, \ldots, x_{m,n}$ denoting the status of nodes in a one-dimensional cellular automata (CA) on a circle (or a line with periodic boundary conditions)
- Uniform, 2-point update rules $f_1, f_2, \cdots, f_m$ defined by "parity" $p(u, v) = (u + v) \bmod 2$ to create the system of difference equations

$$\begin{aligned} x_{1,n} &= (x_{m,n-1} + x_{2,n-1}) \bmod 2 \\ x_{i,n} &= (x_{i-1,n-1} + x_{i+1,n-1}) \bmod 2, \quad 2 \le i \le m-1, \\ x_{m,n} &= (x_{m-1,n-1} + x_{1,n-1}) \bmod 2 \end{aligned}$$

- In CA terminology, the rule is stated as follows: A node i is "on" (has status value 1) at time n if nodes $i-1$ and $i+1$ before and after it have different status values (one is 0 and the other is 1) in time $n-1$. Node i is "off" otherwise.



Δ

Consider $m = 4$ and an initial state $(x_{1,0}, x_{2,0}, x_{3,0}, x_{4,0}) = (1, 0, 0, 0)$. Then

$$\begin{aligned} x_{1,1} &= (x_{4,0} + x_{2,0}) \bmod 2 = (0 + 0) \bmod 2 = 0 \\ x_{2,1} &= (x_{1,0} + x_{3,0}) \bmod 2 = (1 + 0) \bmod 2 = 1 \\ x_{3,1} &= (x_{2,0} + x_{4,0}) \bmod 2 = (0 + 0) \bmod 2 = 0 \\ x_{4,1} &= (x_{3,0} + x_{1,0}) \bmod 2 = (0 + 1) \bmod 2 = 1 \end{aligned}$$

so at time $n = 1$

$$(x_{1,1}, x_{2,1}, x_{3,1}, x_{4,1}) = (0, 1, 0, 1).$$

- Continuing the iteration process yields

  $n = 2 \rightarrow ((1 + 1) \bmod 2, (0 + 0) \bmod 2, (1 + 1) \bmod 2, (0 + 0) \bmod 2) = (0, 0, 0, 0)$
  $n = 3 \rightarrow ((0 + 0) \bmod 2, (0 + 0) \bmod 2, (0 + 0) \bmod 2, (0 + 0) \bmod 2) = (0, 0, 0, 0)$
  $\vdots$

- The orbit locks into the zero state, which is a fixed state that does not change under iteration; the orbit is just 3 steps long:

  $$(1, 0, 0, 0) \rightarrow (0, 1, 0, 1) \rightarrow (0, 0, 0, 0)$$

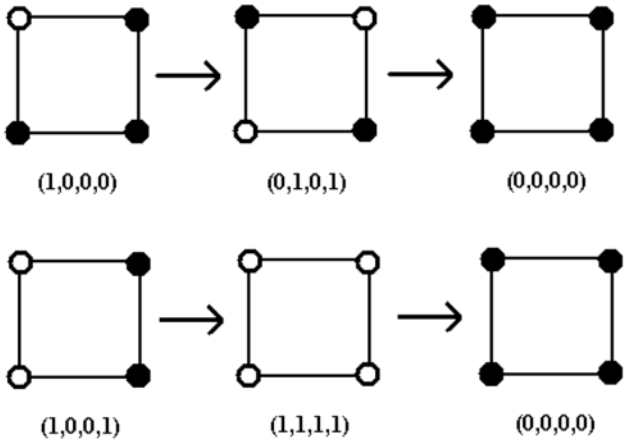
- Clearly, if we started with the state (0,1,0,1) we would reach the fixed state (0,0,0,0) in just one step

- Starting with a different initial state, say (1,0,0,1) the resulting orbit is

  $$(1, 0, 0, 1) \rightarrow (1, 1, 1, 1) \rightarrow (0, 0, 0, 0)$$

Δ

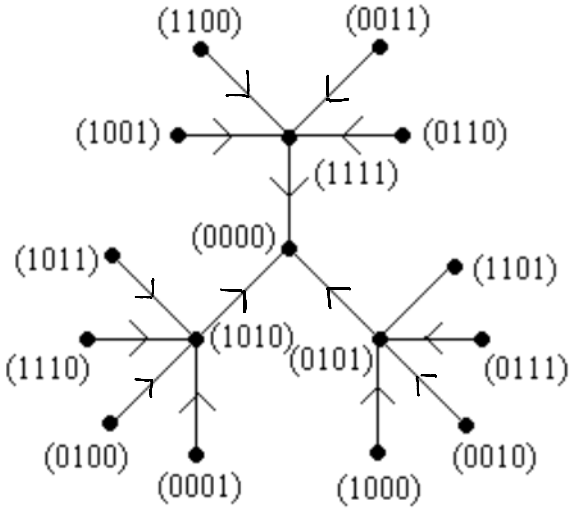
The *space-time diagrams* for the above two orbits may be represented as follows:



Δ

In the previous example with $m = 4$ the finite state-space $\mathbb{Z}_2^4$ has $2^4 = 16$ possible states that are arranged in the form of the digraph below. The zero state (0,0,0,0) is a *global attractor* because every state reaches it.

All of the dynamics of the above CA example in its 16-tuple state-space is captured in a single *state-space digraph*:



Δ

In a classical DDS

$$\begin{aligned} x_{1,n} &= f_1(x_{1,n-1}, x_{2,n-1}, \ldots, x_{m,n-1}) \\ x_{2,n} &= f_2(x_{1,n-1}, x_{2,n-1}, \ldots, x_{m,n-1}) \\ &\vdots \\ x_{m,n} &= f_m(x_{1,n-1}, x_{2,n-1}, \ldots, x_{m,n-1}) \end{aligned}$$

the current-time status $x_{i,n}$ of each variable is determined in terms of one or more of the past-time values $x_{j,n-1}$ as specified by the functions $f_1, \ldots, f_m$.

The updating is done in a *synchronous* (or *parallel*) fashion since inserting the past-time values $x_{1,n-1}, x_{2,n-1}, \ldots, x_{m,n-1}$ into the above system generates all of the current-time values *simultaneously*, or in parallel. Classical DDS are *synchronous or parallel dynamical systems (PDS).*

But...

Synchronously updated DDS are not always the most natural modeling tools!

Δ

An example from cardiac electrophysiology:

The fast-moving reentrant pulse in a loop of cardiac tissue, responsible for the onset of arrhythmia, chases its own tail in a manner that can be represented by a system of difference equations as follows:

$$DI_{i,n} = \sum_{j=1}^{i-1} CT(j, DI_{j,n}) + \sum_{j=i}^{m} CT(j, DI_{j,n-1}) - A(DI_{i,n-1}),\ i = 1, \ldots, m.$$

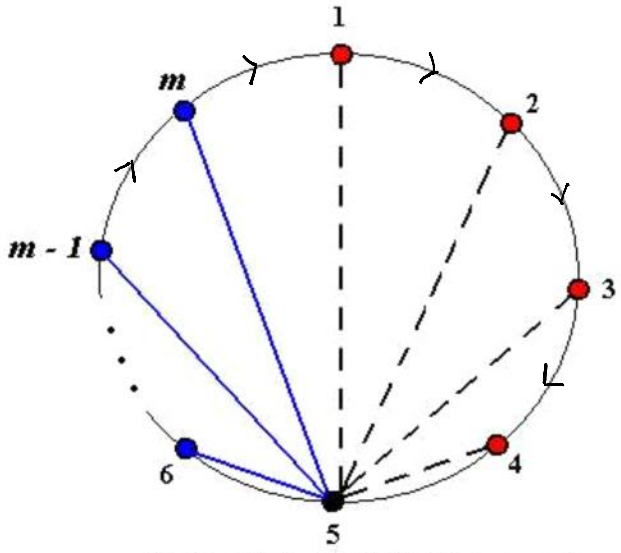
- Variables $DI_{i,n}$ denote "diastolic intervals" (time intervals between successive cell activations) for each of the m cardiac cells (or cell aggregates)
- Functions CT and A represent, respectively, the "restitution" functions for conduction time (propagation time through a cell) and for "action-potential duration" (excitation period of each cell)

OBSERVATION

The first summation contains *current-time* status values of variables $DI_{i,n}$, but only *up to and not including* the i-th cell.

Δ

- Imagine that a pulse is initiated at cell 1 and then proceeds to cell 2, etc till cell m and then starts a new cycle when back at cell 1. This defines an *update order* for the cells, here numbered from 1 to m.
- The cells are NOT updated simultaneously (or synchronously) but serially in a specified order.
- Recall that the reentrant pulse equation contains both current-time status values and past-time values.



Nodes affecting the 5-th node in current time (dashed) and past time (solid)

Δ

A more general definition of discrete dynamical systems that includes cases like the reentrant pulse system can be based on *systems of difference equations across two time levels*: the current time and past.

$$
\begin{aligned}
x_{1,n} &= f_1(a'_{1,1}x_{1,n}, , \ldots, a'_{1,m}x_{m,n}, a_{1,1}x_{1,n-1}, , \ldots, a_{1,m}x_{m,n-1}) \\
x_{2,n} &= f_2(a'_{2,1}x_{1,n}, , \ldots, a'_{2,m}x_{m,n}, a_{2,1}x_{1,n-1}, , \ldots, a_{2,m}x_{m,n-1}) \\
&\vdots \\
x_{m,n} &= f_m(a'_{m,1}x_{1,n}, , \ldots, a'_{m,m}x_{m,n}, a_{m,1}x_{1,n-1}, , \ldots, a_{m,m}x_{m,n-1})
\end{aligned}
$$

where all dependency coefficients $a'_{i,j}, a_{i,j}$ are either 0 or 1.

- $a'_{i,j}, a_{i,j}$ make explicit all interdependencies among the variables
- If $a'_{i,j} = 1$ then the status of variable $x_{i,n}$ is affected by the status of $x_{j,n}$ in the current time period n.
- If $a_{i,j} = 1$ then the status of variable $x_{i,n}$ is affected by the status of $x_{j,n-1}$ in the past time period $n - 1$.
- In the update order $1 \rightarrow 2 \rightarrow 3 \rightarrow \cdots m$ we require that $a'_{i,j} = 0$ if $j \geq i$ because the j-th variable is updated after the i-th one

Δ

The above observations define a particular set of connections among the variables that can be summarized in the $2m \times m$ *dependency matrix*

$$\begin{bmatrix} 0 & 0 & \cdots & 0 & a_{1,1} & a_{1,2} & \cdots & a_{1,m} \\ a'_{2,1} & 0 & \cdots & 0 & a_{2,1} & a_{2,2} & \cdots & a_{2,m} \\ \vdots & & \ddots & & \vdots & & \cdots & \vdots \\ a'_{m,1} & a'_{m,2} & \cdots & 0 & a_{m,1} & a_{m,2} & \cdots & a_{m,m} \end{bmatrix}.$$

- The above matrix of 0's and 1's can be interpreted as the adjacency matrix of a *dependency digraph*.
- If $a'_{i,j} = 1$ for any pair of variables i and j then the DDS is *asynchronous (ADS)*.
- If $a'_{i,j} = 1$ implies that $a_{i,j} = 0$, i.e., the right half of the dependency matrix is upper triangular, then the digraph is proper (no double edges emanating from the same node or vertex). In this case the DDS is a *sequential dynamical system*, or a *SDS*.
- The reentrant pulse system is a SDS since there is no k such that $DI_{k,n}$ and $DI_{k,n-1}$ appear simultaneously in any equation of the system.

Δ

An asynchronous CA with the 2-point parity rule

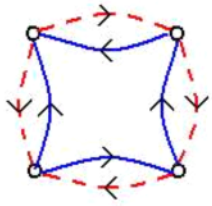
The following is an *asynchronous version* of the 4-variable cellular automata example discussed earlier:

$$\begin{aligned} x_{1,n} &= (x_{4,n-1} + x_{2,n-1}) \bmod 2 \\ x_{2,n} &= (x_{1,n} + x_{3,n-1}) \bmod 2 \\ x_{3,n} &= (x_{2,n} + x_{4,n-1}) \bmod 2 \\ x_{4,n} &= (x_{1,n} + x_{3,n}) \bmod 2 \end{aligned}$$

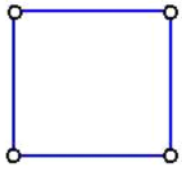
- The dependency matrix of this system is

$$\begin{bmatrix} 0 & 0 & 0 & 0 & 0 & 1 & 0 & 1 \\ 1 & 0 & 0 & 0 & 0 & 0 & 1 & 0 \\ 0 & 1 & 0 & 0 & 0 & 0 & 0 & 1 \\ 1 & 0 & 1 & 0 & 0 & 0 & 0 & 0 \end{bmatrix}$$

- The lower triangular, upper triangular form of the above matrix shows that the DDS is a SDS
- The dependency digraph based on the above matrix is shown below and compared with the same for the earlier synchronous CA
- These graphs show that the parity rule defines a cycle on 4 vertices in the synchronous case but the digraph of the asynchronous case has acyclic orientations in both current and past times



Asynchronous CA
dashed: current time
solid: past time



Synchronous CA
(directed edges merged)

Δ

The asynchronous CA exhibits a very different dynamical behavior than the earlier, synchronous CA

- Seven of the 16 states in the state-space now belong to a 7-cycle:

$$(1,0,1,0) \to (0,1,1,1) \to (0,1,0,0) \to (1,1,1,0) \to (1,0,0,1) \to$$
$$(1,1,0,1) \to (0,0,1,1) \to (1,0,1,0) \to \text{ (cycle repeats)}$$

- The calculations are straightforward; for instance, starting from the state $(x_{1,0}, x_{2,0}, x_{3,0}, x_{4,0}) = (1,0,1,0)$ we have

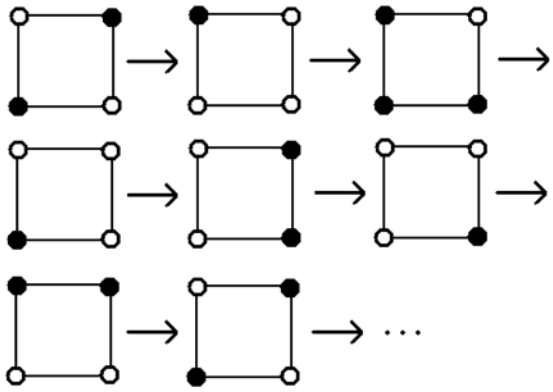
$$\begin{aligned} x_{1,1} &= (x_{4,0} + x_{2,0}) \bmod 2 = (0+0) \bmod 2 = 0 \\ x_{2,1} &= (x_{1,1} + x_{3,0}) \bmod 2 = (0+1) \bmod 2 = 1 \\ x_{3,1} &= (x_{2,1} + x_{4,0}) \bmod 2 = (1+0) \bmod 2 = 1 \\ x_{4,1} &= (x_{1,1} + x_{3,1}) \bmod 2 = (0+1) \bmod 2 = 1 \end{aligned}$$

Therefore, $(1,0,1,0) \to (0,1,1,1)$; recall that in the synchronous case $(1,0,1,0) \to (0,0,0,0)$

- Every *other* state except (1,0,0,0) and (0,0,0,0) reaches the above 7-cycle in one step; and $(1,0,0,0) \to (0,0,0,0)$

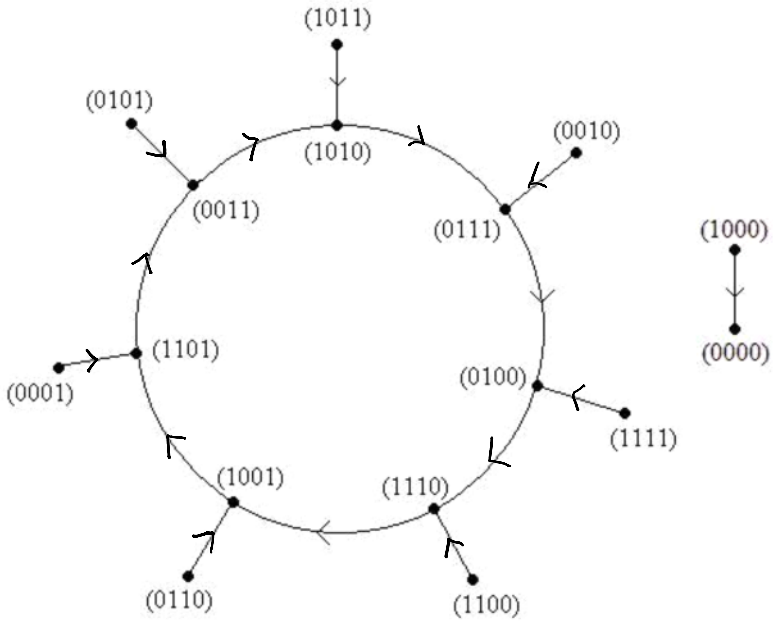
Δ

The space-time diagram of the above orbit is shown below



Δ

The overall state-space digraph of the asynchronous CA is as follows:



Δ

The synchronous equivalent of an ADS

The asynchronous system

$$\begin{aligned} x_{1,n} &= (x_{4,n-1} + x_{2,n-1}) \bmod 2 \\ x_{2,n} &= (x_{1,n} + x_{3,n-1}) \bmod 2 \\ x_{3,n} &= (x_{2,n} + x_{4,n-1}) \bmod 2 \\ x_{4,n} &= (x_{1,n} + x_{3,n}) \bmod 2 \end{aligned}$$

can be transformed into a synchronous one by substituting past time equivalents for each occurrence of current time status values.

- Using the status $x_{1,n}$ from past time values, for $x_{2,n}$ we obtain

$$\begin{aligned} x_{2,n} &= (x_{1,n} + x_{3,n-1}) \bmod 2 \\ &= ((x_{4,n-1} + x_{2,n-1}) \bmod 2 + x_{3,n-1}) \bmod 2 \\ &= (x_{4,n-1} + x_{2,n-1} + x_{3,n-1}) \bmod 2 \end{aligned}$$

- Using the current time value of $x_{2,n}$ we find

$$\begin{aligned} x_{3,n} &= (x_{2,n} + x_{4,n-1}) \bmod 2 \\ &= (x_{2,n-1} + x_{3,n-1} + 2x_{4,n-1}) \bmod 2 \\ &= (x_{2,n-1} + x_{3,n-1}) \bmod 2 \end{aligned}$$

- Using the current time values of $x_{1,n}$ and $x_{3,n}$ we find

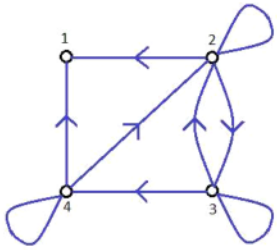
$$\begin{aligned} x_{4,n} &= (x_{1,n} + x_{3,n}) \bmod 2 \\ &= (2x_{2,n-1} + x_{3,n-1} + x_{4,n-1}) \bmod 2 \\ &= (x_{3,n-1} + x_{4,n-1}) \bmod 2 \end{aligned}$$

Δ

The synchronous equivalent of the asynchronous CA is

$$
\begin{aligned}
x_{1,n} &= (x_{2,n-1} + x_{4,n-1}) \bmod 2 \\
x_{2,n} &= (x_{2,n-1} + x_{3,n-1} + x_{4,n-1}) \bmod 2 \\
x_{3,n} &= (x_{2,n-1} + x_{3,n-1}) \bmod 2 \\
x_{4,n} &= (x_{3,n-1} + x_{4,n-1}) \bmod 2
\end{aligned}
$$

which is clearly different from the synchronous CA with a 2-point parity rule; in particular, the digraph of this equivalent PDS now has loops:



Δ

Closing remarks

- A synchronous equivalent PDS may not exist for all ADS or it may not be unique; e.g., ADS defined by non-recursive systems of difference equations (not discussed here)
- If two SDS have the same equivalent synchronous forms they are said to be "functionally equivalent" by some authors
- It may be practically infeasible to calculate the synchronous equivalent for a large SDS whose dependency digraph contains many vertices and edges (e.g., the reentrant pusle equation)
- For practical and theoretical reasons it is necessary to develop methods that are directly applicable to ADS without needing to convert them to an equivalent PDS; very few such methods have been developed so far