Five Lectures on CA

5. Asynchronous and probabilistic CA

Thomas Worsch

Department of Informatics
Karlsruhe Institute of Technology
http://liinwww.ira.uka.de/~thw/vl-hiroshima/

at Hiroshima University, January 2012

Contents

- Asynchronous CA
  - Basics
  - Examples
- Probabilistic CA
  - Basics
  - Density classification

- Asynchronous CA
  - Basics
  - Examples
- Probabilistic CA
  - Basics
  - Density classification

Why asynchronous CA?

- implementation of CA in nanotechnology
  - no global clock for synchronous operation
- simulation of natural phenomena
  - asynchronous operation may give more realistic results

Until now

- deterministic CA $C = (R, Q, N, f)$
- synchronous mode of operation
  - $F : Q^R \to Q^R$ induced by f according to

$$\forall x \in R : F(c)(x) = f(L(c, x))$$

  - F is a function.

Definition: asynchronous CA

- asynchronous mode of operation
  - (sub)set $A \subseteq R$ of active cells
  - $F_A : Q^R \to Q^R$ induced by f according to

$$\begin{array}{rl} \text{active cells} & \forall x \in A\colon F_A(c)(x) = f(L(c,x)) \\ \text{passive cells} & \forall x \in R \setminus A : F_A(c)(x) = c(x) \end{array}$$

  - F_A is a function
- one step relation $\vdash \subseteq Q^R \times Q^R$

$$c \vdash c' \iff \exists A\colon F_A(c) = c'$$

Example

rule 170: $f(\ell) = \ell(1)$

Example

rule 170: $f(\ell) = \ell(1)$ shift left

Example

rule 170: $f(\ell) = \ell(1)$ shift left

synchronous operation

1	0	1	1	0	1	1	1	0	...
0	1	1	0	1	1	1	0	0	...
1	1	0	1	1	1	0	0	0	...
1	0	1	1	1	0	0	0	0	...
0	1	1	1	0	0	0	0	0	...

active

Example

rule 170: $f(\ell) = \ell(1)$ shift left

synchronous operation

1	0	1	1	0	1	1	1	0	...
0	1	1	0	1	1	1	0	0	...
1	1	0	1	1	1	0	0	0	...
1	0	1	1	1	0	0	0	0	...
0	1	1	1	0	0	0	0	0	...

active

Example

rule 170: $f(\ell) = \ell(1)$ shift left

synchronous operation

1	0	1	1	0	1	1	1	0	...
0	1	1	0	1	1	1	0	0	...
1	1	0	1	1	1	0	0	0	...
1	0	1	1	1	0	0	0	0	...
0	1	1	1	0	0	0	0	0	...

active

Example

rule 170: $f(\ell) = \ell(1)$ shift left

synchronous operation

1	0	1	1	0	1	1	1	0	...
0	1	1	0	1	1	1	0	0	...
1	1	0	1	1	1	0	0	0	...
1	0	1	1	1	0	0	0	0	...
0	1	1	1	0	0	0	0	0	...

active

Example

rule 170: $f(\ell) = \ell(1)$ shift left

synchronous operation

1	0	1	1	0	1	1	1	0	...
0	1	1	0	1	1	1	0	0	...
1	1	0	1	1	1	0	0	0	...
1	0	1	1	1	0	0	0	0	...
0	1	1	1	0	0	0	0	0	...

Example

rule 170: $f(\ell) = \ell(1)$ shift left

synchronous operation

1	0	1	1	0	1	1	1	0	...
0	1	1	0	1	1	1	0	0	...
1	1	0	1	1	1	0	0	0	...
1	0	1	1	1	0	0	0	0	...
0	1	1	1	0	0	0	0	0	...

asynchronous operation, for example:

1	0	1	1	0	1	1	1	0	...
1	1	1	0	0	1	1	0	0	...
1	1	1	0	1	1	1	0	0	...
1	1	0	0	1	1	0	0	0	...
1	0	0	1	1	0	0	0	0	...

Example

rule 170: $f(\ell) = \ell(1)$ shift left

synchronous operation

1	0	1	1	0	1	1	1	0	...
0	1	1	0	1	1	1	0	0	...
1	1	0	1	1	1	0	0	0	...
1	0	1	1	1	0	0	0	0	...
0	1	1	1	0	0	0	0	0	...

asynchronous operation, for example:

1	0	1	1	0	1	1	1	0	...
1	1	1	0	0	1	1	0	0	...
1	1	1	0	1	1	1	0	0	...
1	1	0	0	1	1	0	0	0	...
1	0	0	1	1	0	0	0	0	...

active

Example

rule 170: $f(\ell) = \ell(1)$ shift left

synchronous operation

1	0	1	1	0	1	1	1	0	...
0	1	1	0	1	1	1	0	0	...
1	1	0	1	1	1	0	0	0	...
1	0	1	1	1	0	0	0	0	...
0	1	1	1	0	0	0	0	0	...

asynchronous operation, for example:

1	0	1	1	0	1	1	1	0	...
1	1	1	0	0	1	1	0	0	...
1	1	1	0	1	1	1	0	0	...
1	1	0	0	1	1	0	0	0	...
1	0	0	1	1	0	0	0	0	...

active

Example

rule 170: $f(\ell) = \ell(1)$ shift left

synchronous operation

1	0	1	1	0	1	1	1	0	...
0	1	1	0	1	1	1	0	0	...
1	1	0	1	1	1	0	0	0	...
1	0	1	1	1	0	0	0	0	...
0	1	1	1	0	0	0	0	0	...

asynchronous operation, for example:

1	0	1	1	0	1	1	1	0	...
1	1	1	0	0	1	1	0	0	...
1	1	1	0	1	1	1	0	0	...
1	1	0	0	1	1	0	0	0	...
1	0	0	1	1	0	0	0	0	...

active

Example

rule 170: $f(\ell) = \ell(1)$ shift left

synchronous operation

1	0	1	1	0	1	1	1	0	...
0	1	1	0	1	1	1	0	0	...
1	1	0	1	1	1	0	0	0	...
1	0	1	1	1	0	0	0	0	...
0	1	1	1	0	0	0	0	0	...

asynchronous operation, for example:

1	0	1	1	0	1	1	1	0	...
1	1	1	0	0	1	1	0	0	...
1	1	1	0	1	1	1	0	0	...
1	1	0	0	1	1	0	0	0	...
1	0	0	1	1	0	0	0	0	...

active

Example

rule 170: $f(\ell) = \ell(1)$ shift left

synchronous operation

1	0	1	1	0	1	1	1	0	...
0	1	1	0	1	1	1	0	0	...
1	1	0	1	1	1	0	0	0	...
1	0	1	1	1	0	0	0	0	...
0	1	1	1	0	0	0	0	0	...

asynchronous operation, for example:

1	0	1	1	0	1	1	1	0	...
1	1	1	0	0	1	1	0	0	...
1	1	1	0	1	1	1	0	0	...
1	1	0	0	1	1	0	0	0	...
1	0	0	1	1	0	0	0	0	...

How to tame asynchronous CA

- using a local function designed for the synchronous case
- designing local functions directly for the asynchronous case

Nakamura's technique

- given: a CA (R, Q, N, f) designed for the synchronous case
- wanted: a CA (R, Q', N, f') which does "more or less the same" when used in the asynchronous mode of operation

Nakamura's technique

- given: a CA (R, Q, N, f) designed for the synchronous case
- wanted: a CA (R, Q', N, f') which does "more or less the same" when used in the asynchronous mode of operation
- Nakamura's idea (1970):
  - let some cells be ahead of others,
  - but never more than one step
    compared to any other cell in the neighborhood
  - and remember the old state for those behind

Nakamura's technique

- given: a CA $C = (R, Q, N, f)$ for the synchronous case
  - assume that $0 \in N$
- wanted: a CA $C' = (R, Q', N, f')$ for the asynchronous case
- $Q' = Q \times Q \times \{0, 1, 2\}$,
  components of $q' \in Q'$ are denoted
  $\mathit{current}(q')$, $\mathit{old}(q')$, and $\mathit{time}(q')$

Nakamura's technique

- given: a CA $C = (R, Q, N, f)$ for the synchronous case
  - assume that $0 \in N$
- wanted: a CA $C' = (R, Q', N, f')$ for the asynchronous case
- $Q' = Q \times Q \times \{0, 1, 2\}$,
  components of $q' \in Q'$ are denoted
  $\mathit{current}(q')$, $\mathit{old}(q')$, and $\mathit{time}(q')$

- define f' for local configuration ℓ' as follows:
  - If for all $n \in N$: $\mathit{time}(\ell'(n)) = \mathit{time}(\ell'(0))$ or
    $\mathit{time}(\ell'(n)) = \mathit{time}(\ell'(0)) + 1 \pmod 3$:
    then $f'(\ell') = (f(\ell), \mathit{current}(\ell'(0)), \mathit{time}(\ell'(0)) + 1 \pmod 3)$,
    where

$$\ell(n) = \begin{cases} \mathit{current}(\ell'(n)) & \text{if } \mathit{time}(\ell'(n)) = \mathit{time}(\ell'(0)) \\ \mathit{old}(\ell'(n)) & \text{if } \mathit{time}(\ell'(n)) = \mathit{time}(\ell'(0)) + 1 \pmod 3 \end{cases}$$

  - otherwise: $f'(\ell') = \ell'(0)$.

Nakamura's technique

- given: a CA $C = (R, Q, N, f)$ for the synchronous case
  - assume that $0 \in N$
- wanted: a CA $C' = (R, Q', N, f')$ for the asynchronous case
- $Q' = Q \times Q \times \{0, 1, 2\}$,
  components of $q' \in Q'$ are denoted
  $\mathit{current}(q')$, $\mathit{old}(q')$, and $\mathit{time}(q')$

- define f' for local configuration ℓ' as follows:
  - If for all $n \in N$: $\mathit{time}(\ell'(n)) = \mathit{time}(\ell'(0))$ or
    $\mathit{time}(\ell'(n)) = \mathit{time}(\ell'(0)) + 1 \pmod 3$:
    then $f'(\ell') = (f(\ell), \mathit{current}(\ell'(0)), \mathit{time}(\ell'(0)) + 1 \pmod 3)$,
    where

$$\ell(n) = \begin{cases} \mathit{current}(\ell'(n)) & \text{if } \mathit{time}(\ell'(n)) = \mathit{time}(\ell'(0)) \\ \mathit{old}(\ell'(n)) & \text{if } \mathit{time}(\ell'(n)) = \mathit{time}(\ell'(0)) + 1 \pmod 3 \end{cases}$$

  - otherwise: $f'(\ell') = \ell'(0)$.

Nakamura's technique

- given: a CA $C = (R, Q, N, f)$ for the synchronous case
  - assume that $0 \in N$
- wanted: a CA $C' = (R, Q', N, f')$ for the asynchronous case
- $Q' = Q \times Q \times \{0, 1, 2\}$,
  components of $q' \in Q'$ are denoted
  $\mathit{current}(q')$, $\mathit{old}(q')$, and $\mathit{time}(q')$

- define f' for local configuration ℓ' as follows:
  - If for all $n \in N$: $\mathit{time}(\ell'(n)) = \mathit{time}(\ell'(0))$ or
    $\mathit{time}(\ell'(n)) = \mathit{time}(\ell'(0)) + 1 \pmod 3$:
    then $f'(\ell') = (f(\ell), \mathit{current}(\ell'(0)), \mathit{time}(\ell'(0)) + 1 \pmod 3)$,
    where

$$\ell(n) = \begin{cases} \mathit{current}(\ell'(n)) & \text{if } \mathit{time}(\ell'(n)) = \mathit{time}(\ell'(0)) \\ \mathit{old}(\ell'(n)) & \text{if } \mathit{time}(\ell'(n)) = \mathit{time}(\ell'(0)) + 1 \pmod 3 \end{cases}$$

  - otherwise: $f'(\ell') = \ell'(0)$.

Example

synchronous operation:

1	0	1	1	0	1	1	1	0	$t = 0$
0	1	1	0	1	1	1	0	0	$t = 1$
1	1	0	1	1	1	0	0	0	$t = 2$
1	0	1	1	1	0	0	0	0	$t = 3$
0	1	1	1	0	0	0	0	0	$t = 4$
1	1	1	0	0	0	0	0	0	$t = 5$

simulated by the asynchronous CA storing this:

0	1	1	0	1	0	0	0	0	$old(q')$
1	0	0	1	1	0	0	0	0	$current(q')$
2	0	2	2	0	1	2	1	0	$time(q')$

Example

synchronous operation:

1	0	1	1	0	1	1	1	0	$t = 0$
0	1	1	0	1	1	1	0	0	$t = 1$
1	1	0	1	1	1	0	0	0	$t = 2$
1	0	1	1	1	0	0	0	0	$t = 3$
0	1	1	1	0	0	0	0	0	$t = 4$
1	1	1	0	0	0	0	0	0	$t = 5$

simulated by the asynchronous CA storing this:

0	1	1	0	1	0	0	0	0	$old(q')$
1	0	0	1	1	0	0	0	0	$current(q')$
2	0	2	2	0	1	2	1	0	$time(q')$

Example

synchronous operation:

1	0	1	1	0	1	1	1	0	$t = 0$
0	1	1	0	1	1	1	0	0	$t = 1$
1	1	0	1	1	1	0	0	0	$t = 2$
1	0	1	1	1	0	0	0	0	$t = 3$
0	1	1	1	0	0	0	0	0	$t = 4$
1	1	1	0	0	0	0	0	0	$t = 5$

simulated by the asynchronous CA storing this:

0	1	1	0	1	0	0	0	0	$old(q')$
1	0	0	1	1	0	0	0	0	$current(q')$
2	0	2	2	0	1	2	1	0	$time(q')$

Example

synchronous operation:

1	0	1	1	0	1	1	1	0	$t = 0$
0	1	1	0	1	1	1	0	0	$t = 1$
1	1	0	1	1	1	0	0	0	$t = 2$
1	0	1	1	1	0	0	0	0	$t = 3$
0	1	1	1	0	0	0	0	0	$t = 4$
1	1	1	0	0	0	0	0	0	$t = 5$

simulated by the asynchronous CA storing this:

0	1	1	0	1	0	0	0	0	$old(q')$
1	0	0	1	1	0	0	0	0	$current(q')$
2	0	2	2	0	1	2	1	0	$time(q')$

Example

synchronous operation:

1	0	1	1	0	1	1	1	0	$t = 0$
0	1	1	0	1	1	1	0	0	$t = 1$
1	1	0	1	1	1	0	0	0	$t = 2$
1	0	1	1	1	0	0	0	0	$t = 3$
0	1	1	1	0	0	0	0	0	$t = 4$
1	1	1	0	0	0	0	0	0	$t = 5$

simulated by the asynchronous CA storing this:

0	1	1	0	1	0	0	0	0	$old(q')$
1	0	0	1	1	0	0	0	0	$current(q')$
2	0	2	2	0	1	2	1	0	$time(q')$

Example

synchronous operation:

1	0	1	1	0	1	1	1	0	$t = 0$
0	1	1	0	1	1	1	0	0	$t = 1$
1	1	0	1	1	1	0	0	0	$t = 2$
1	0	1	1	1	0	0	0	0	$t = 3$
0	1	1	1	0	0	0	0	0	$t = 4$
1	1	1	0	0	0	0	0	0	$t = 5$

simulated by the asynchronous CA storing this:

0	1	1	0	1	0	0	0	0	$old(q')$
1	0	0	1	1	0	0	0	0	$current(q')$
2	0	2	2	0	1	2	1	0	$time(q')$

Example

synchronous operation:

1	0	1	1	0	1	1	1	0	$t = 0$
0	1	1	0	1	1	1	0	0	$t = 1$
1	1	0	1	1	1	0	0	0	$t = 2$
1	0	1	1	1	0	0	0	0	$t = 3$
0	1	1	1	0	0	0	0	0	$t = 4$
1	1	1	0	0	0	0	0	0	$t = 5$

simulated by the asynchronous CA storing this:

0	1	1	0	1	0	0	0	0	$old(q')$
1	0	0	1	1	0	0	0	0	$current(q')$
2	0	2	2	0	1	2	1	0	$time(q')$

after activation of cell 3 (only):

0	1	0	0	1	0	0	0	0	$old(q')$
1	0	1	1	1	0	0	0	0	$current(q')$
2	0	0	2	0	1	2	1	0	$time(q')$

Example

synchronous operation:

1	0	1	1	0	1	1	1	0	$t = 0$
0	1	1	0	1	1	1	0	0	$t = 1$
1	1	0	1	1	1	0	0	0	$t = 2$
1	0	1	1	1	0	0	0	0	$t = 3$
0	1	1	1	0	0	0	0	0	$t = 4$
1	1	1	0	0	0	0	0	0	$t = 5$

simulated by the asynchronous CA storing this:

0	1	1	0	1	0	0	0	0	$old(q')$
1	0	0	1	1	0	0	0	0	$current(q')$
2	0	2	2	0	1	2	1	0	$time(q')$

after activation of cell 3 (only):

0	1	0	0	1	0	0	0	0	$old(q')$
1	0	1	1	1	0	0	0	0	$current(q')$
2	0	0	2	0	1	2	1	0	$time(q')$

Designing rules for asynchronous CA

- may be more complicated
  - use a trick like the "otherwise" case in Nakamura's f'
  - simulate some other kind of device, e. g.
    embed delay insensitive circuits into a two-dimensional CA
- may be easier

Definition: α-asynchronous CA

- useful for simulations
  - of CA on a computer
  - of real phenomena
- α: a probability $0 \leq \alpha \leq 1$
- make set A of active cells a random variable:
  - in each step each cell x is included in A with probability α
  - independently of the others
  - then use F_A

now:

- first a few examples
- then let's look at something more general

- Asynchronous CA
  - Basics
  - Examples
- Probabilistic CA
  - Basics
  - Density classification

Let's change the discrete space

- $N = \{-1, 0, 1\}$
- from now on a finite ring of n cells:
  $R = \mathbb{Z}_n = \{0, 1, \ldots, n-1\}$
  - $x + n$ computed $\mod n$
  - neighbor -1 of cell 0 is cell $n-1$
  - neighbor 1 of cell $n-1$ is cell 0

Deterministic rule: R232

- rule 232: "majority rule" (see simulation)

$$f(\ell) = \begin{cases} 0 & \text{if} \quad \ell(-1) + \ell(0) + \ell(1) \leq 1 \\ 1 & \text{if} \quad \ell(-1) + \ell(0) + \ell(1) \geq 2 \end{cases}$$

Deterministic rule: R232

- rule 232: "majority rule" (see simulation)

$$f(\ell) = \begin{cases} 0 & \text{if} \quad \ell(-1) + \ell(0) + \ell(1) \leq 1 \\ 1 & \text{if} \quad \ell(-1) + \ell(0) + \ell(1) \geq 2 \end{cases}$$

- synchronous updating:
  - blocks of at least two identical consecutive bits are conserved
  - "isolated" bits are destroyed

- asynchronous updating, small α:
  - blocks of at least two identical consecutive bits are conserved
  - "isolated" bits are destroyed after some time

Deterministic rule: R184

- rule 184: “traffic rule” (see simulation)
- a “car” 1
  - moves to the right if there is an “empty street” 0 leaving an “empty street” 0 behind
  - stays where it is, otherwise

Deterministic rule: R184

- rule 184: "traffic rule" (see simulation)
- a "car" 1
  - moves to the right if there is an "empty street" 0 leaving an "empty street" 0 behind
  - stays where it is, otherwise
- the number of 0 and 1 is never changed

- synchronous updating:
  - separated cars move to the right
  - traffic jams move to the left

- asynchronous updating, very large α:
  - "basically the same"
    (sorry for not being precise)

Two-dimensional minority rule

- Regnault/Schabanel/Thierry (2009):
  Progresses in the analysis of stochastic 2D cellular automata:
  A study of asynchronous 2D minority
  Theoretical Computer Sience **410**, pp. 4844–4855.
- significant changes when changing α from 1 to 0
- different behavior for the 5- and 9-neighborhoods

α-asynchronous CA are difficult to analyze

- Asynchronous CA
  - Basics
  - Examples

- Probabilistic CA
  - Basics
  - Density classification

- Asynchronous CA
  - Basics
  - Examples
- Probabilistic CA
  - Basics
  - Density classification

Definition

- write $[0;1]$ for the interval of real numbers between 0 and 1
- The local function of a probabilistic CA is a function

$$f\colon Q^N \to [0;1]^Q$$

- $f(\ell)(q)$ is the probability that a cell observing ℓ in its neighborhood chooses q as its new state.
- therefore additional requirement:

$$\text{for each} \quad \ell \in Q^N\colon \sum_{q\in Q} f(\ell)(q) = 1$$

- f is applied using the synchronous mode of operation.

- avoid "strange" probabilities (uncomputable numbers, ...)

- deterministic CA using synchronous updating: special case

α-asynchronous CA

are a special case:

- given deterministic $f : Q^N \to Q$
- define $f' : Q^N \to [0;1]^Q$ like this

α-asynchronous CA

are a special case:

- given deterministic $f : Q^N \to Q$
- define $f' : Q^N \to [0;1]^Q$ like this

$$f'(\ell) = \begin{cases} f(\ell) & \text{with probability } \alpha \\ \ell(0) & \text{with probability } 1-\alpha \end{cases}$$

(the two cases may give the same state)

- claim:
  - using f in the α-asynchronous mode of operation and
  - using f' in the synchronous mode of operation

result in the same global behaviors, even quantitatively

- Asynchronous CA
  - Basics
  - Examples
- Probabilistic CA
  - Basics
  - Density classification

Problem

- design a CA with
  - $Q = \{0, 1\}$ fixed! no additional states allowed
  - arbitrary N
  - arbitrary f
- such that for each *odd* number n
- using the CA on $R = \mathbb{Z}_n$
- for each initial configuration $c \in R^Q$
- after a finite number of steps the following configuration is reached:
  - all cells in state 0, if the majority of cells in c was in state 0
  - all cells in state 1, if the majority of cells in c was in state 1

Theorem (Land/Belew, 1995)

There is no deterministic CA solving the density classification problem with synchronous updating.

Theorem (Land/Belew, 1995)

There is no deterministic CA solving the density classification problem with synchronous updating.

- some people try to find rules that "most of the time"
- empirical work

Theorem (Fukś, 1995)

The density classification problem can be solved using two CA:

- first run $n/2$ steps with rule 184
- then run $n/2$ steps with rule 232

Theorem (Fukś, 1995)

The density classification problem can be solved using two CA:

- first run $n/2$ steps with rule 184
- then run $n/2$ steps with rule 232

explanation:

- after $n/2$ steps of the traffic rule either
  - all traffic jams have dissolved or
  - all "empty streets" of length ≥ 2 are gone
  - depending on what had the majority in the beginning
  - leaving only exactly one type of blocks of length ≥ 2
- then the majority rule
  - will change all checker board patterns
  - to the state of cells which is in the majority

Probabilistic CA for density classification

Fatès (2011):

- use a probabilistic CA
- "$f_{Fates} = \alpha \cdot f_{184} + (1 - \alpha) \cdot f_{232}$"
- for all $\ell \in Q^N$:

$$f_{Fates}(\ell) = \begin{cases} f_{184}(\ell) & \text{with probability } \alpha \\ f_{232}(\ell) & \text{with probability } 1 - \alpha \end{cases}$$

Probabilistic CA for density classification

Fatès (2011):

- use a probabilistic CA
- "$f_{Fates} = \alpha \cdot f_{184} + (1 - \alpha) \cdot f_{232}$"
- for all $\ell \in Q^N$:

$$f_{Fates}(\ell) = \begin{cases} f_{184}(\ell) & \text{with probability } \alpha \\ f_{232}(\ell) & \text{with probability } 1 - \alpha \end{cases}$$

- Theorem: For each n there is a (large) probability α such that f_{Fates} works "very well" (in some precise sense).
- experimental result: it seems that α has not to be *that* large

...

Summary

- aynchronous CA have “no gloal clock”
  - any subset of cells can be updated in a global step
- probabilistic CA allow each cell choose between different new states (if wanted)
  - this can help to solve – with high probability – problems which are otherwise unsolvable