Standardizing the set of states and the neighborhood of asynchronous cellular automata

Thomas Worsch worsch@kit.edu

Karlsruhe Institute of Technology Institute for Cryptography and Security

Classical result

Alvy Ray Smith III. Cellular automata complexity trade-offs. Information and Control, 18:466–482, 1971.

For synchronous CA it is true that

- each CA with k = |Q| states can be simulated by a CA with 2 = |{0,1}| states
- each CA with N = {−r,..., −1, 0, 1, ... r} can be simulated by a CA with N = {−1, 0, 1}.

Asynchronous CA (ACA)

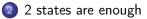
- w.l.o.g. consider one-dimensional CA
- Z set of cells
- Q set of states
- N neighborhood
- f local transition function $Q^N \to Q$
- A ⊆ Z "activity set"
- F_A global transition function $Q^{\mathbf{Z}} \rightarrow Q^{\mathbf{Z}}$

$$F_A(c)(i) = \begin{cases} f(c_{i+N}) & \text{iff } i \in A \\ c(i) & \text{iff } i \notin A \end{cases}$$

• F global transition relation $F = \bigcup_{A \subseteq \mathbf{Z}} F_A$

Outline

An idea by Lee et al.



3 Neighborhood radius 1 is enough



Generalization of the idea by Lee et al.

- An idea by Lee et al.
 - 2 2 states are enough
- 3 Neighborhood radius 1 is enough
- 4 Generalization of the idea by Lee et al.

Idea by Lee et al.

how to simulate a synchronous CA on an asynchronous CA

- Lee/Adachi/Peper/Morita (2004), Schumacher (2012)
- synchronous CA $C_s = (Q_s, N_s, f_s)$
- asynchronous CA $C_a = (Q_a, N_a, f_a)$

 \sim

$$N_a = N_s \cup -N_s \cup \{0\}$$

where

given c_s ∈ Q_s^Z
define c_a ∈ Q_a^Z

$$\forall i \in \mathbf{Z} : c_a(i) = (c_s(i), B)$$

An idea by Lee et al. (2) local transition function for the asynchronous CA

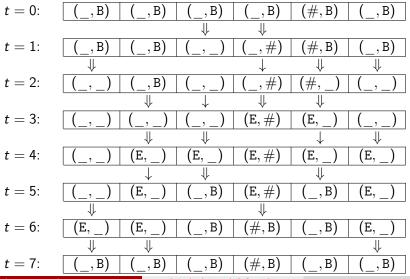
 $B \rightarrow T$: if all neighbors are in $Q_B \cup Q_T$ then transition $(q, B) \mapsto (q, f_s(q_1, \dots, q_k))$ where q_i first components of the neighboring states

$$\mathtt{T} o \mathtt{E}: ext{ if all neighbors are in } Q_{\mathtt{T}} \cup Q_{\mathtt{E}} \\ ext{ then transition } (q,q') \mapsto (\mathtt{E},q')$$

 $extsf{E}
ightarrow extsf{B}$: if all neighbors are in $Q_{ extsf{E}} \cup Q_{ extsf{B}}$ then transition $(extsf{E}, q') \mapsto (q', extsf{B})$

otherwise no change

An idea by Lee et al. (3) example: shift left



1 An idea by Lee et al.

2 states are enough

- 3 Neighborhood radius 1 is enough
- 4 Generalization of the idea by Lee et al.

Theorem

Each ACA with $k = |Q_a| \ge 3$ states can be simulated by an ACA with $2 = |\{0, 1\}|$ states.

From k states to 2 states (1) Encoding of one state

- "one-hot" encoding
- represent one state as

		mai	rker				ol	d sta	te			ne	ate		
0	1	1	1	1	0	01	<i>o</i> ₂		0 _k	XE	n_1	<i>n</i> ₂	• • •	n _k	XB

- the marker cannot appear anywhere else
- use large neighborhood each cell can observe all neighboring segments

From k states to 2 states (2) Transitions

marker		ol	d sta	te		new state					corresponds
		Oi	0j			nj					to
011110	0	1	0	0	0	0	0	0	0	В	(q_i, B)
011110	0	1	0	0	0	0	0	1	0	В	
011110	0	1	0	0	0	0	0	1	0	0	(q_i,q_j)
011110	0	1	0	0	0	0	0	1	0	0	(q_i,q_j)
011110	0	1	0	0	E	0	0	1	0	0	
011110	0	0	0	0	E	0	0	1	0	0	(E, q_j)
011110	0	0	0	0	E	0	0	1	0	0	(E, q_j)
011110	0	0	0	0	E	0	0	1	0	В	
011110	0	0	1	0	E	0	0	1	0	В	
011110	0	0	1	0	E	0	0	0	0	В	
011110	0	0	1	0	0	0	0	0	0	В	(q_j, B)

Thomas Worsch worsch@kit.edu

Standard N and Q for ACA

From k states to 2 states (3) Transitions

- for the simulation of synchronous CA
 - use Lee's idea
- for the simulation of asynchronous CA
 - $\bullet\,$ start B \to T transitions only if all neighboring segments represent an unmodified B state
 - · do not care about neighboring segments afterwards

Optimization

segments of length $O(\log |Q|)$

- encode each state using $\log |Q| + O(1)$ bits
- represent each "logical" bit as 01 resp. 10
 - use 00 indicating "undefined"
 - $\bullet\,$ useful for checking progress of state copying during $E \to B$ transitions

- 1 An idea by Lee et al.
- 2 2 states are enough
- 3 Neighborhood radius 1 is enough
 - 4 Generalization of the idea by Lee et al.

Theorem

Each ACA with $N = \{-r, \ldots, -1, 0, 1, \ldots, r\}$ can be simulated by an ACA with $N = \{-1, 0, 1\}$.

Construction

given ACA C

- construct ACA C'
 - each cell has additional activity bit a
 - if a = 1: active cell will use original f of C
 - if a = 0: active cell will not change its state
- construct ACA C["]
 - apply to C' construction by Smith (1971) for the synchronous case
 - no need to copy the *a* bits from neighbors
- construct C'''
 - apply to $C^{\prime\prime}$ the construction to make the local transition function robust for asychronous updating

construct C''''

- "plug in" local rules for computing a from data present in $N = \{-1, 0, 1\}$
- such that any arbitrary subset of cells can have its activity bits set

- 1 An idea by Lee et al.
- 2 2 states are enough
- 3 Neighborhood radius 1 is enough
- Generalization of the idea by Lee et al.

Generalization of the idea

• extend the set of states further

$$\mathit{Q_{a}} = \mathit{Q_{B}} \cup \mathit{Q_{R}} \cup \mathit{Q_{A}} \cup \mathit{Q_{T}} \cup \mathit{Q_{E}}$$

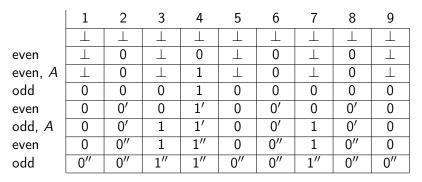
• where for example

$$\begin{array}{l} Q_{\rm B} = \{(q,{\rm B}) \mid q \in Q\} & \times \{\bot\} & \times \{\bot\} \\ Q_{\rm R} = \{(q,{\rm R}) \mid q \in Q\} & \times \{0,1\}^m \times \{\bot\} \\ Q_{\rm A} = \{(q,{\rm A}) \mid q \in Q\} & \times \{0,1\}^m \times \{0,1\} \\ Q_{\rm T} = \{(q,q') \mid q,q' \in Q\} \times \{\bot\} & \times \{\bot\} \\ Q_{\rm E} = \{({\rm E},q') \mid q' \in Q\} & \times \{\bot\} & \times \{\bot\} \end{array}$$

"auxiliary" bits "activity" bit

Generalization of the idea (2) local transition function

for $A \subseteq \mathbf{Z}$ realize corresponding activity bits for example as follows



Outlook

Summary and Outlook

- ACA with k states can be simulated by ACA with 2 states (using larger N)
- ACA with N_r can be simulated by ACA with N₁ (using larger Q)

Open problems:

- different plugins for computing activity bits
- "speed-up" by constant factor for ACA?

Outlook

Summary and Outlook

- ACA with k states can be simulated by ACA with 2 states (using larger N)
- ACA with N_r can be simulated by ACA with N₁ (using larger Q)

Open problems:

- different plugins for computing activity bits
- "speed-up" by constant factor for ACA?

Thank you very much for your attention!