Introduction Computational Science

Introduction Computational Science

by Lera (Valeria) Krzhizhanovskaya

V.Krzhizhanovksya@uva.nl

L3. CA. Langton parameter. Measuring complexity.

Computational Science

L1. Intro Computational Sci (recap)

- **The 3rd pillar of science**
  - Experiment
  - Theory
  - Modelling & simulation
- **Why model? and what?**
- **System, experiment, model, simulation**
  - Only an idiot uses simulation in place of <.?.>
  - Don't fall in love with your model! Danger!
- **Validation, verification**
- **Types of models**

L2. Cellular Automata. 1D (recap)

- **What is CA? Why study? Applications?**
- **Range *r*, Neighbourhood $N=2r+1$, *k* States Σ,**
- **Input alphabet α, Size $m=k^N$**
- **k^m Transition functions Δ (a.k.a. Rules)**
- **Rule numbering by "Wolfram code"**

  Rule 30: $30 = 00011110_2$

111	110	101	100	011	010	001	000
0	0	0	1	1	1	1	0

- **Wolfram classes**

  1 homogeneous, 2 stable pattern, 3 chaos, 4 complex

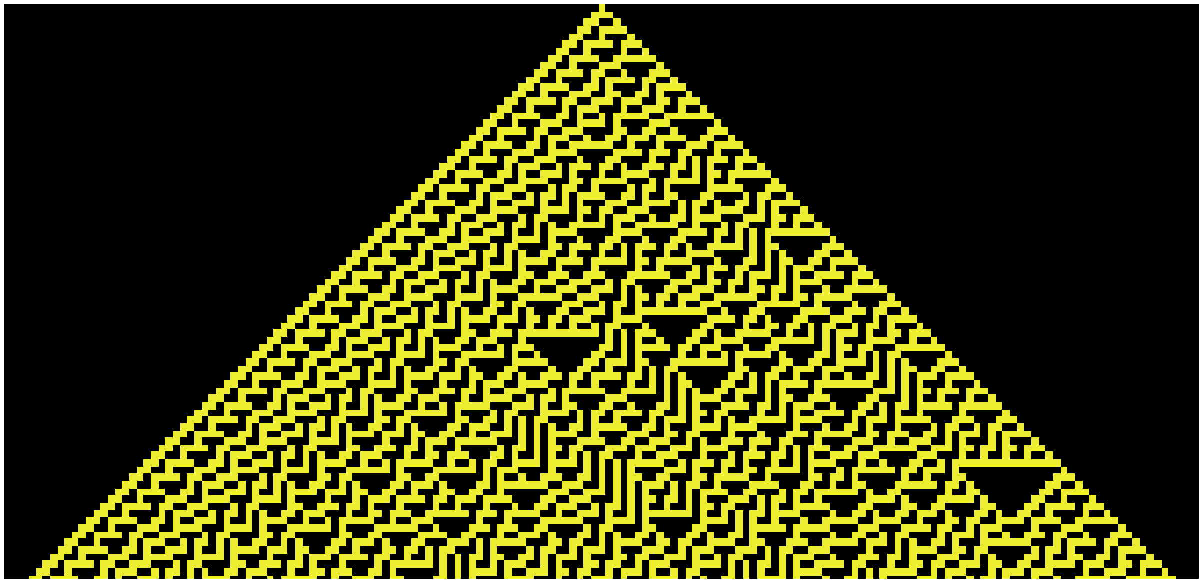
- **Transients and cycles** 001 → 011 → 101 → 110 → 011 → 101 →…

Langton. “Computation at the edge of chaos”

NOW:
QUANTIFY COMPLEXITY

Rule 30

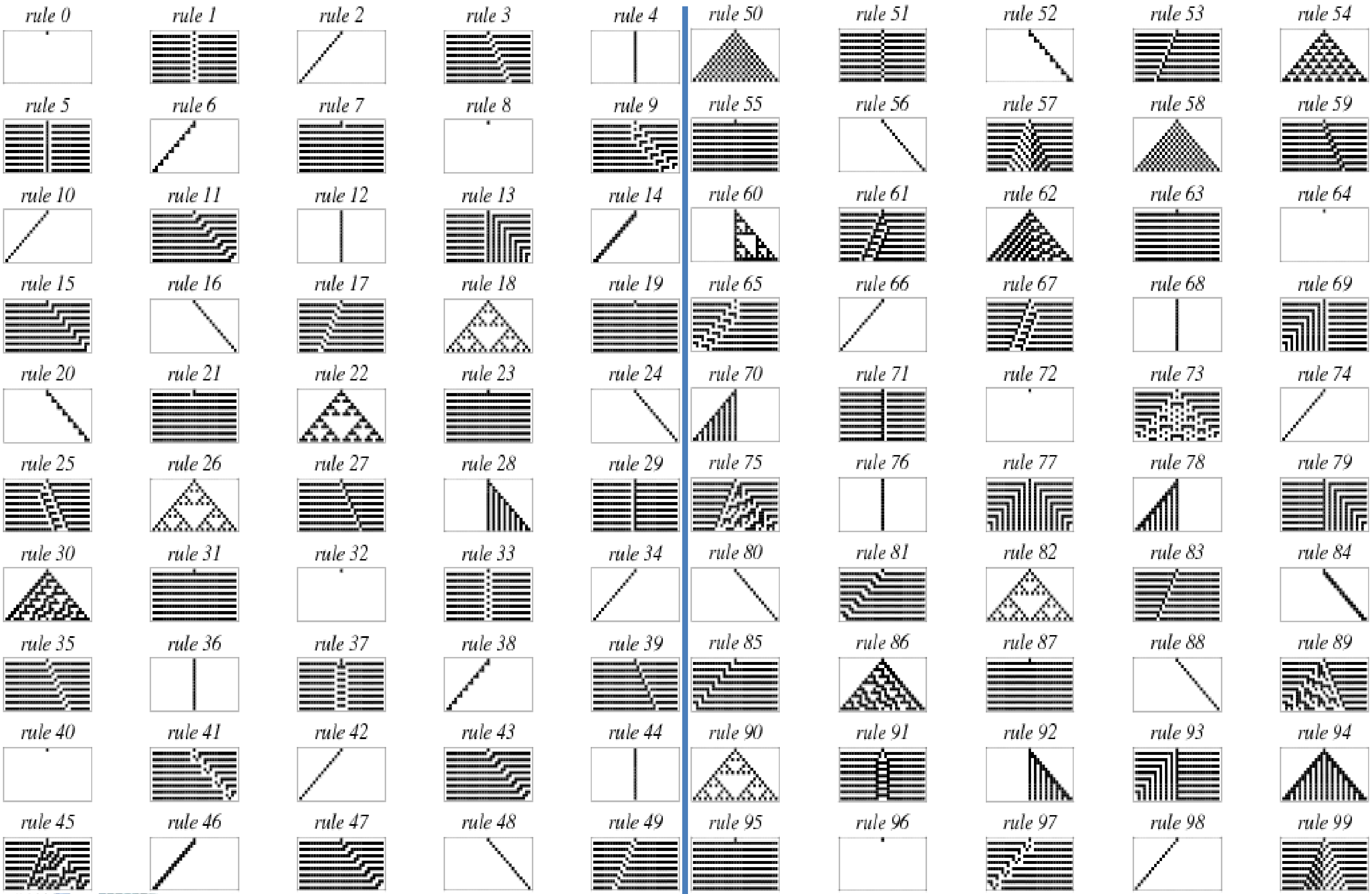
- **Wolfram Class 3 automaton (random)**



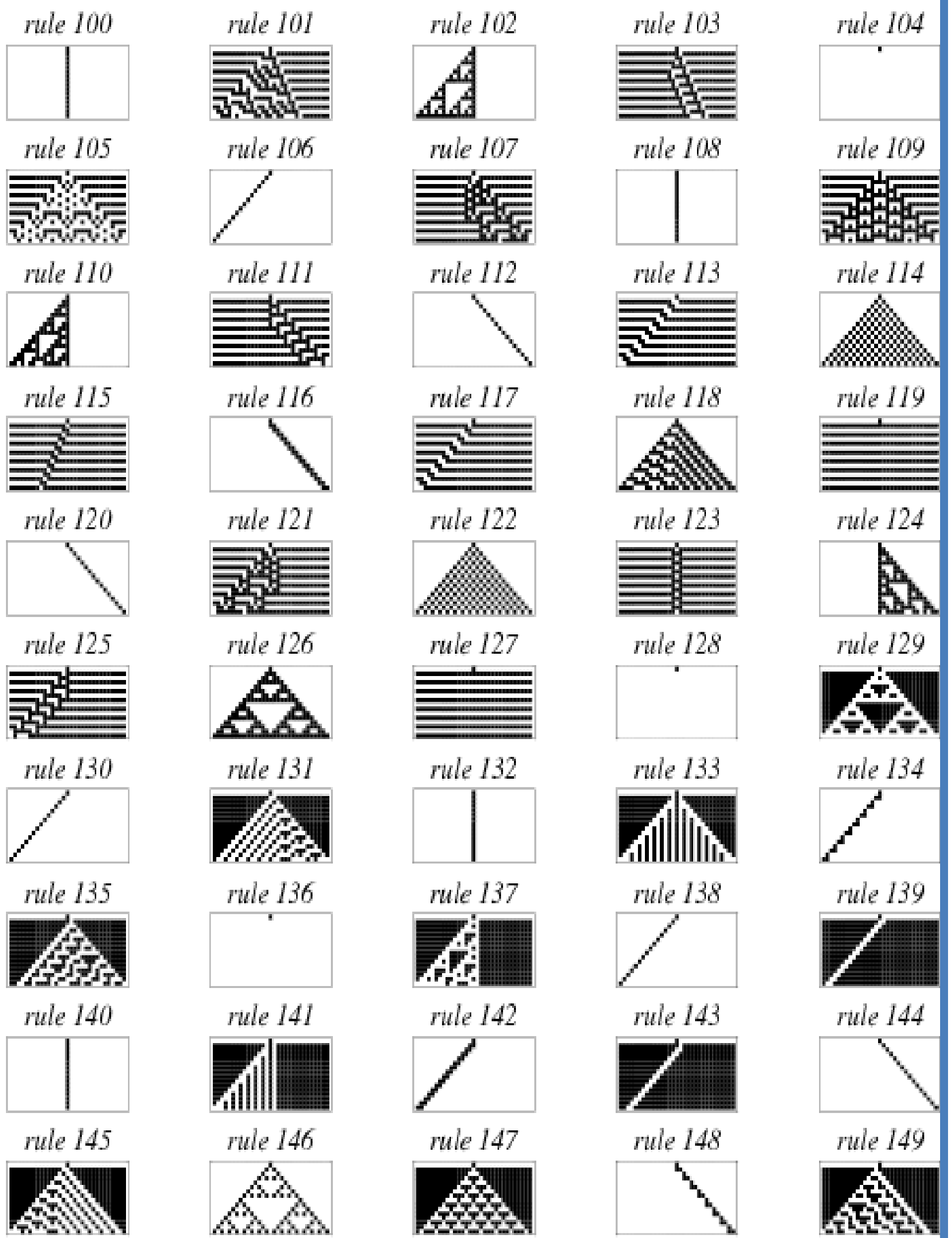
1D CA Rules Netlogo

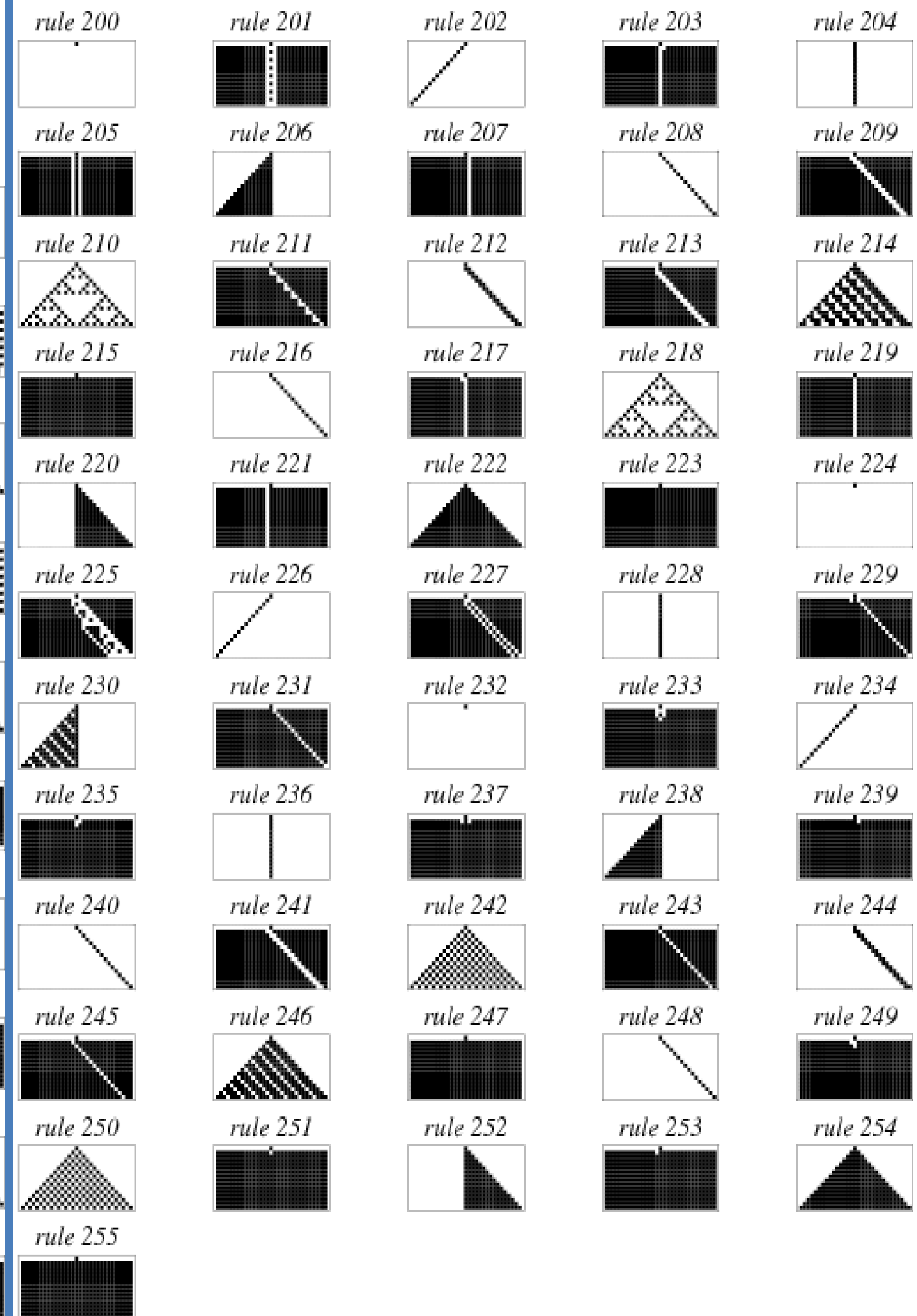
111	110	101	100	011	010	001	000
0	0	0	1	1	1	1	0

All 1D CA



rule 0 rule 1 rule 2 rule 3 rule 4 rule 50 rule 51 rule 52 rule 53 rule 54
rule 5 rule 6 rule 7 rule 8 rule 9 rule 55 rule 56 rule 57 rule 58 rule 59
rule 10 rule 11 rule 12 rule 13 rule 14 rule 60 rule 61 rule 62 rule 63 rule 64
rule 15 rule 16 rule 17 rule 18 rule 19 rule 65 rule 66 rule 67 rule 68 rule 69
rule 20 rule 21 rule 22 rule 23 rule 24 rule 70 rule 71 rule 72 rule 73 rule 74
rule 25 rule 26 rule 27 rule 28 rule 29 rule 75 rule 76 rule 77 rule 78 rule 79
rule 30 rule 31 rule 32 rule 33 rule 34 rule 80 rule 81 rule 82 rule 83 rule 84
rule 35 rule 36 rule 37 rule 38 rule 39 rule 85 rule 86 rule 87 rule 88 rule 89
rule 40 rule 41 rule 42 rule 43 rule 44 rule 90 rule 91 rule 92 rule 93 rule 94
rule 45 rule 46 rule 47 rule 48 rule 49 rule 95 rule 96 rule 97 rule 98 rule 99


rule 100
rule 101
rule 102
rule 103
rule 104
rule 105
rule 106
rule 107
rule 108
rule 109
rule 110
rule 111
rule 112
rule 113
rule 114
rule 115
rule 116
rule 117
rule 118
rule 119
rule 120
rule 121
rule 122
rule 123
rule 124
rule 125
rule 126
rule 127
rule 128
rule 129
rule 130
rule 131
rule 132
rule 133
rule 134
rule 135
rule 136
rule 137
rule 138
rule 139
rule 140
rule 141
rule 142
rule 143
rule 144
rule 145
rule 146
rule 147
rule 148
rule 149


rule 200
rule 201
rule 202
rule 203
rule 204
rule 205
rule 206
rule 207
rule 208
rule 209
rule 210
rule 211
rule 212
rule 213
rule 214
rule 215
rule 216
rule 217
rule 218
rule 219
rule 220
rule 221
rule 222
rule 223
rule 224
rule 225
rule 226
rule 227
rule 228
rule 229
rule 230
rule 231
rule 232
rule 233
rule 234
rule 235
rule 236
rule 237
rule 238
rule 239
rule 240
rule 241
rule 242
rule 243
rule 244
rule 245
rule 246
rule 247
rule 248
rule 249
rule 250
rule 251
rule 252
rule 253
rule 254
rule 255

WOLFRAM CLASSIFICATION

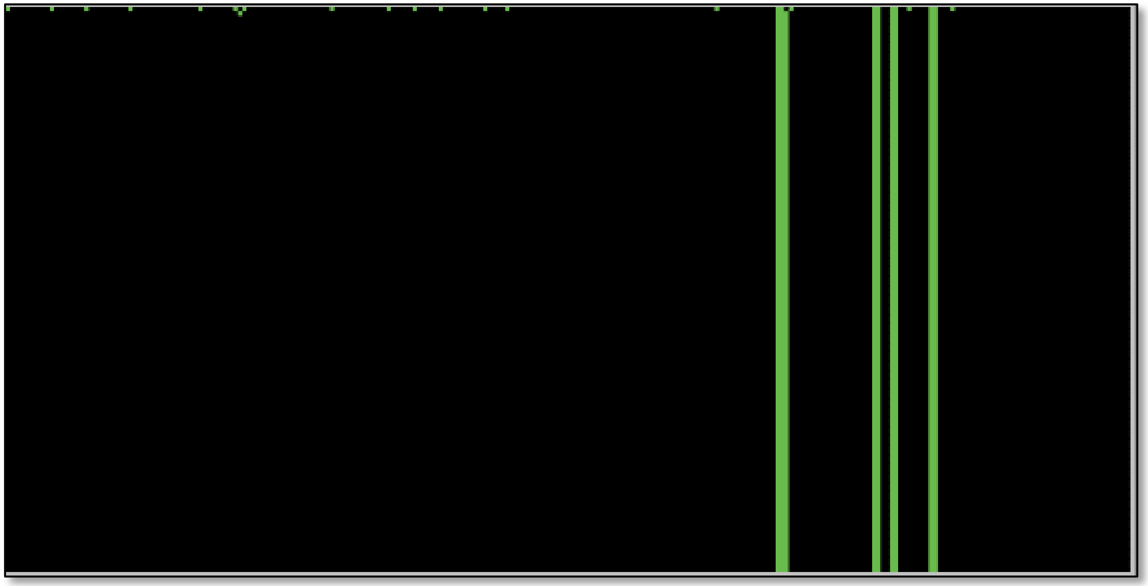
Behavioral Classes of CA

- **Class 1:**
  - Evolution leads to a stable homogeneous state, in which all cells eventually attain the same value.

  - Examples are rules 0, 32, 160 and 250.

Behavioral Classes of CA

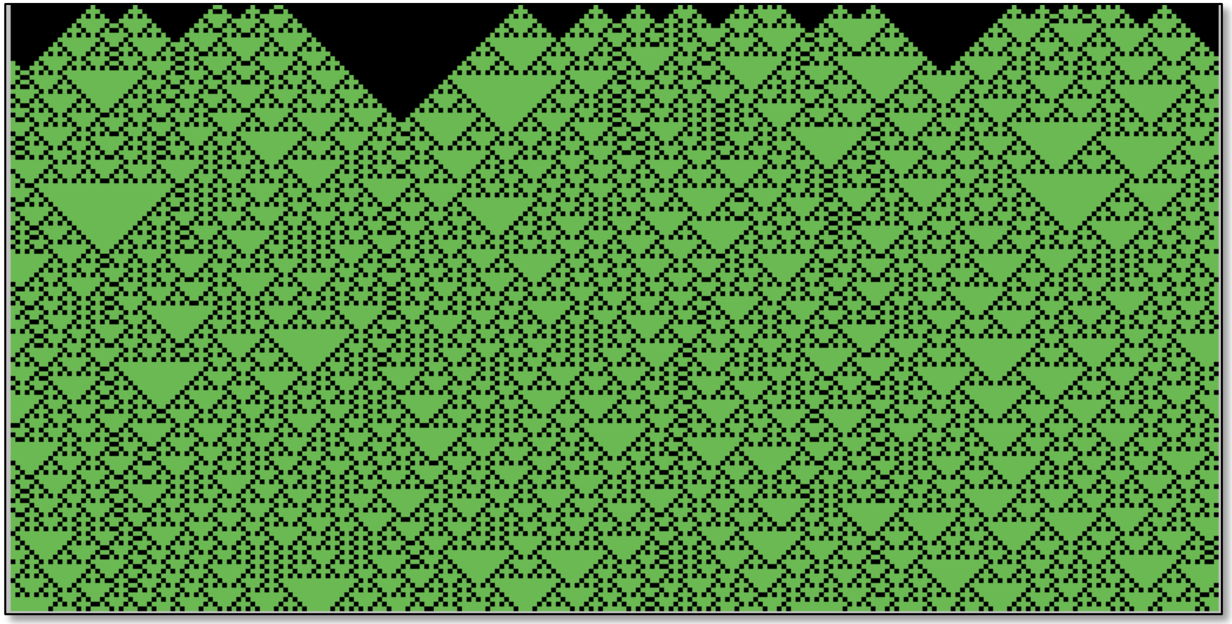
- **Class 2:**
  - Evolution leads to inhomogeneous state: either simple stable states or periodic and separated structures.



  - Examples are rules 4, 108, 218 and 232.

Behavioral Classes of CA

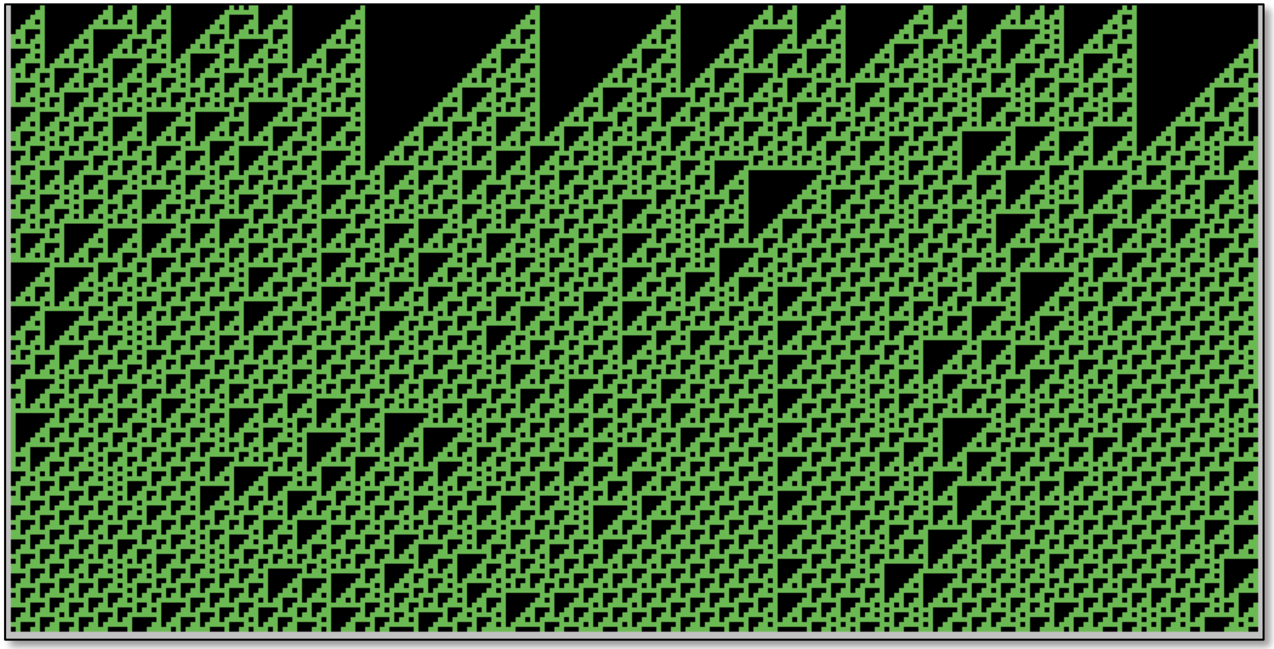
- **Class 3:**
  - Evolution leads to chaotic nonperiodic patterns.



  - Examples are rules 22, 30, 126, 150, 182.

Behavioral Classes of CA

- **Class 4:**
  - Evolution leads to complex, localized propagating structures.



  - Rule 110

Each class is useful!

- **Class 1,2: deterministic computation on input and then 'halts'**
- **Class 3: pseudo-random numbers, cryptography**
- **Class 4: emergent, interacting structures; universal computation; digital physics; life?**

…But how to 'find' them? (Especially for larger *k, r*)

- Can you look at a state transition table and predict which class of behaviour it will exhibit? Or design one for a class?

LANGTON PARAMETER

“Computation and the edge of chaos”

Langton's λ-parameter

- **A single parameter to differentiate behavior of CAs**
- **λ used to specify the rule set Δ of the CA. '*How random is the rule set*'**

- **PDF available on blackboard – good to read:**

  Chris G. Langton. 1990. **Computation at the edge of chaos: phase transitions and emergent computation**. Physica D 42 (1990) 12-37

Langton's λ-parameter

- **Pick an arbitrary state s ∈ Σ, and call it the *quiescent* state s_q**
- **Count the number of transitions in Δ that produce this quiescent state, and call it *n***
- **The other k^N-n transitions must produce the non-quiescent states of Σ - s_q, but may otherwise be chosen at random.**

$$l(\mathrm{D}) = \frac{k^N - n}{k^N}$$

Langton's λ-parameter

$$\lambda(\Delta) = \frac{k^N - n}{k^N}$$

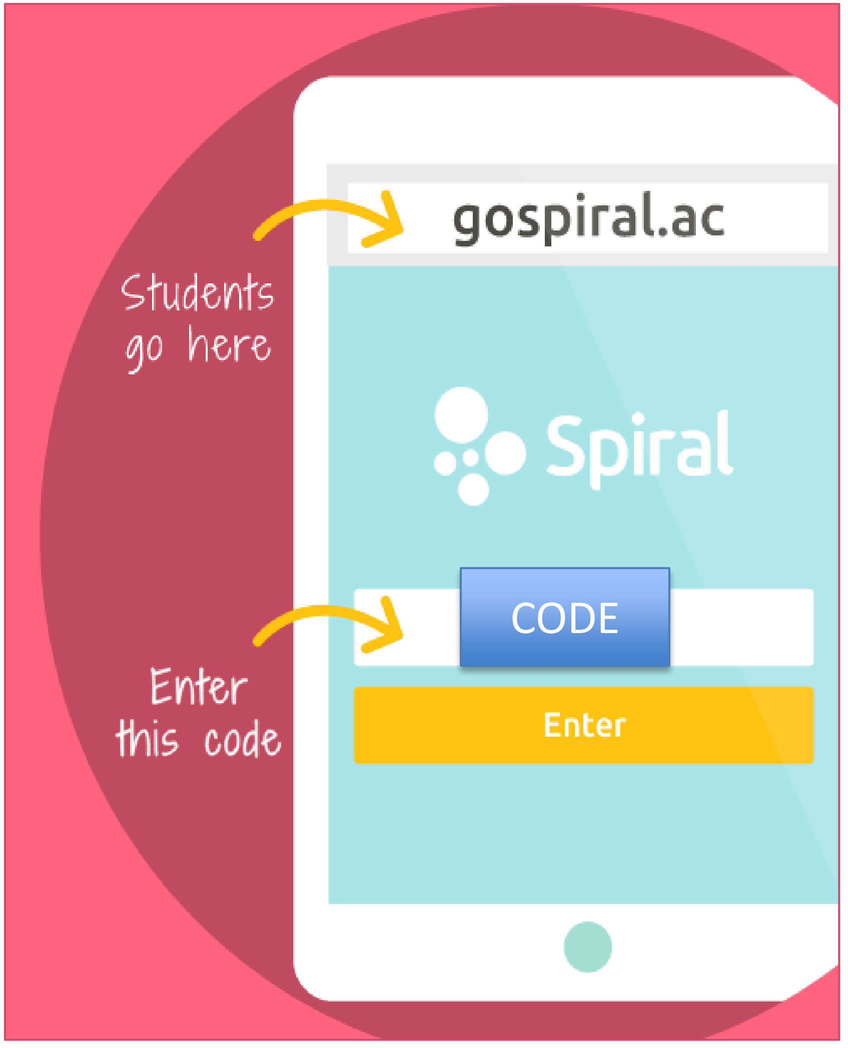
- **If n = k^N, all transitions lead to s_q, λ = 0**

- **If n = 0, no transitions lead to s_q, λ = 1**

- **If all states are represented equally:**

  n = k^N/k, λ = 1-(1/k)

Quiz: Q1, Q2


gospiral.ac
Students go here
Spiral
CODE
Enter
Enter this code

Building Δ from λ

- **Two methods to build rule table Δ for a particular value of λ:**
  1. **Random table**: λ is a bias on the random selection of states from Σ
  2. **Table walk-through**: start with table entirely set to s_q and change some to random according to λ

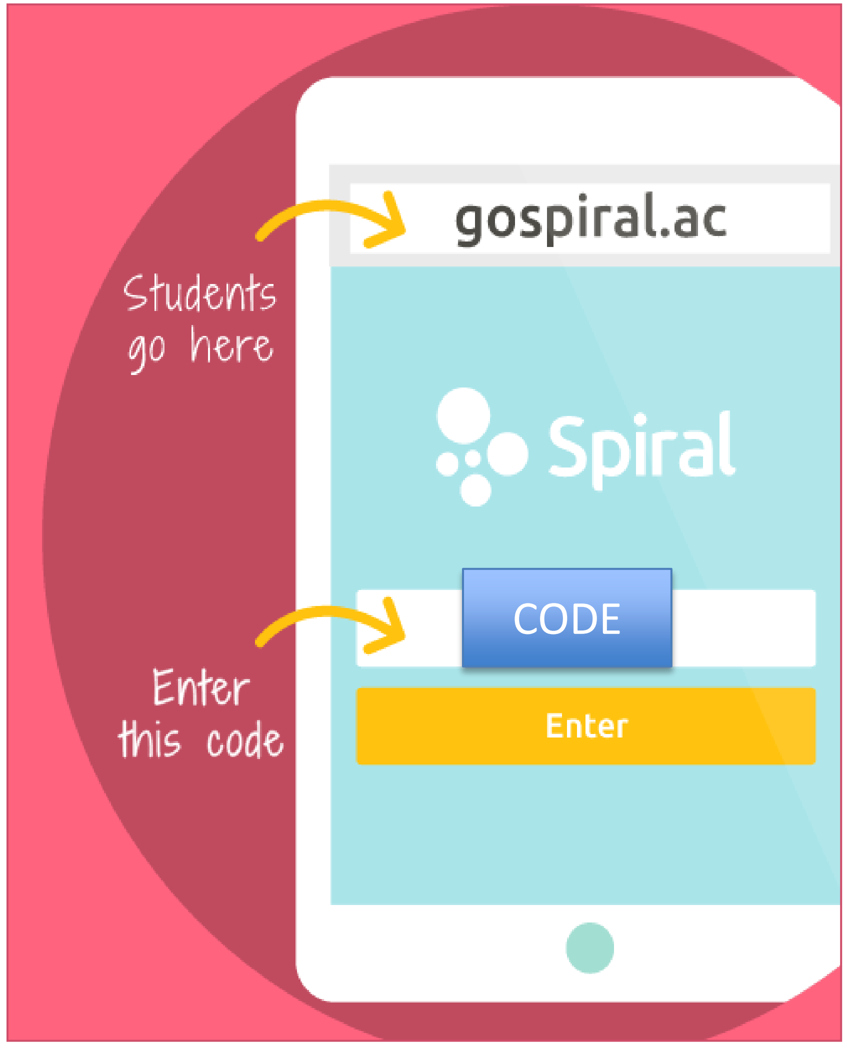
1. Random Table for λ

- **For $0.0 \leq \lambda \leq 1/k$ in discrete steps:**

For each table of the k^N cells (with input states r_i)

1. Generate uniform random number *g* in [0, 1]
2. *if* $g>\lambda$ *then* set output state to be s_q
3. *else* set output to another random state $s_p \in S$, $p \neq q$

Blackboard + Quiz: Q3, Q4



2. Table-walk-through for λ

- **Initialize all k^N states with s_q (λ=0)**
- **Progressively change the Rules (starting from the previous table, and not generating randomly each time)**
- **Algorithm to increase from λ to λ' :**

1 select $(\lambda' - \lambda)k^N$ input states at random

2 set output states to $s_p \in S$, $p \neq q$
(choose one of the p states randomly)

Langton's λ-parameter

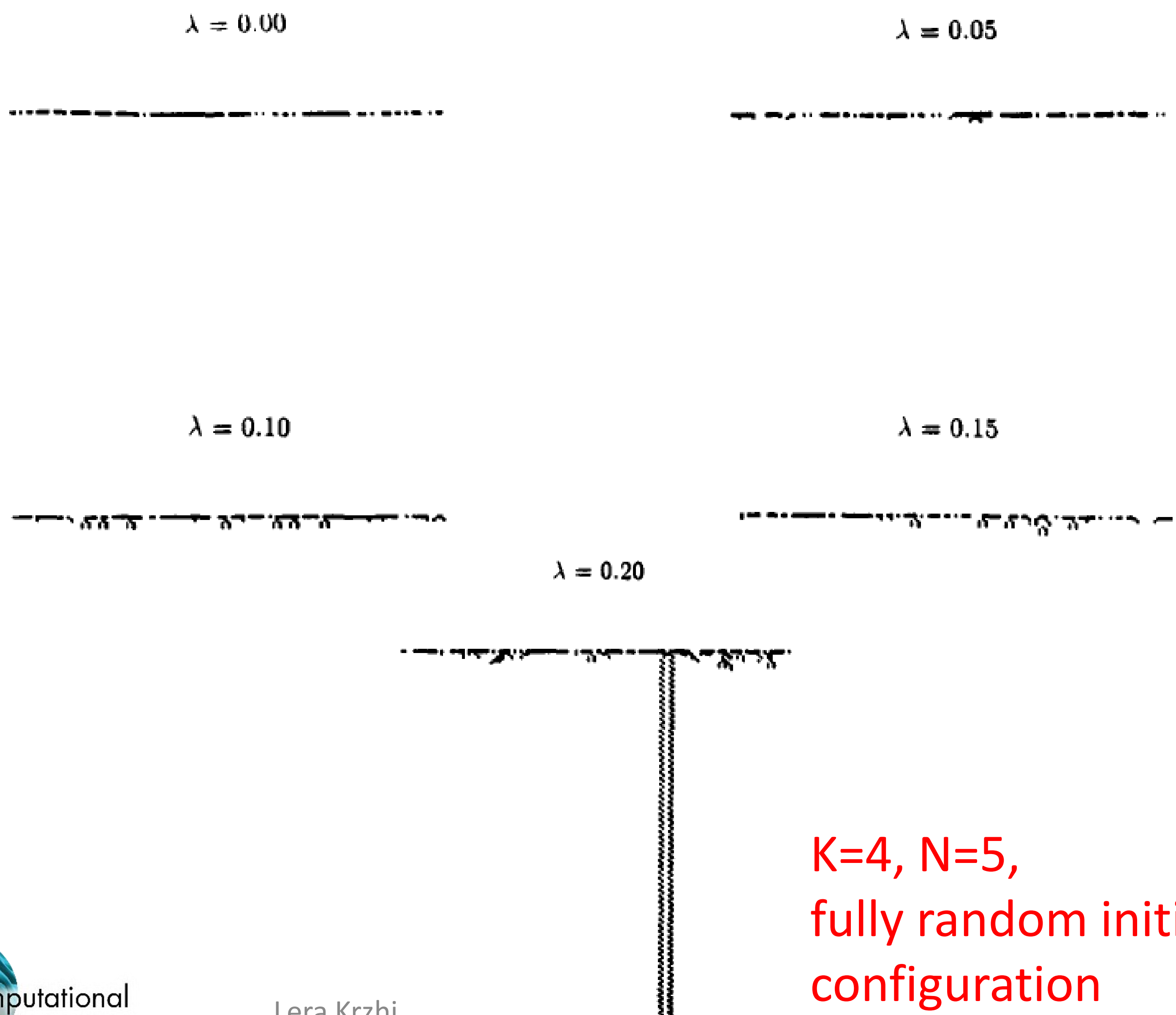
- Other parameterizations of CA rule space exist, but the simplicity and single-dimensionality of λ make it attractive

- λ discriminates well between dynamical regimes for "large" values of K and N, but not for small dimensional spaces. For example, λ is only roughly correlated with dynamical behavior for 1-D CAs with K=2 and N=3

- Langton sticks to CAs with K ≥ 4 and N ≥ 5 (r = 2), which results in transition tables of size $4^5 = 1024$ or larger, a total rules space of 4^4^5 = 3.23*10^{616}

NB: The Universe: "only" 10^{80} atoms!

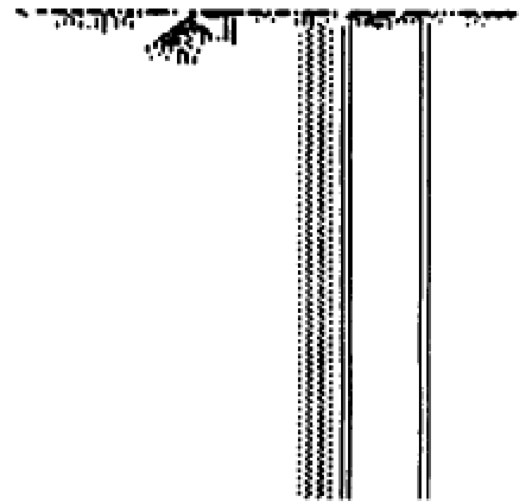
Figure 1 from Langton's paper



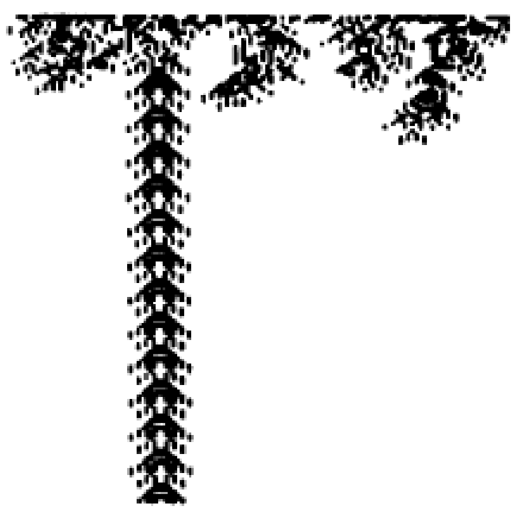
K=4, N=5,
fully random initial configuration

$\lambda = 0.25$

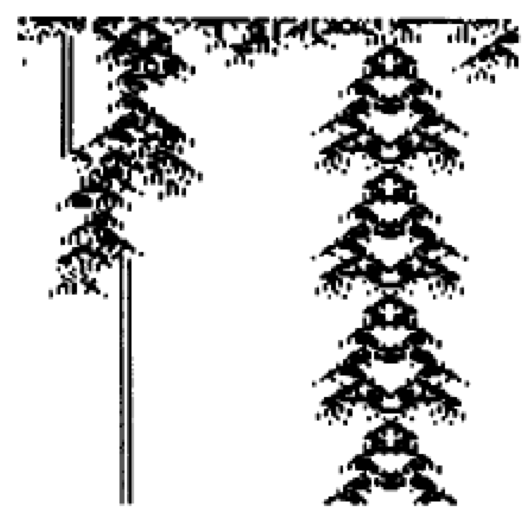
$\lambda = 0.30$

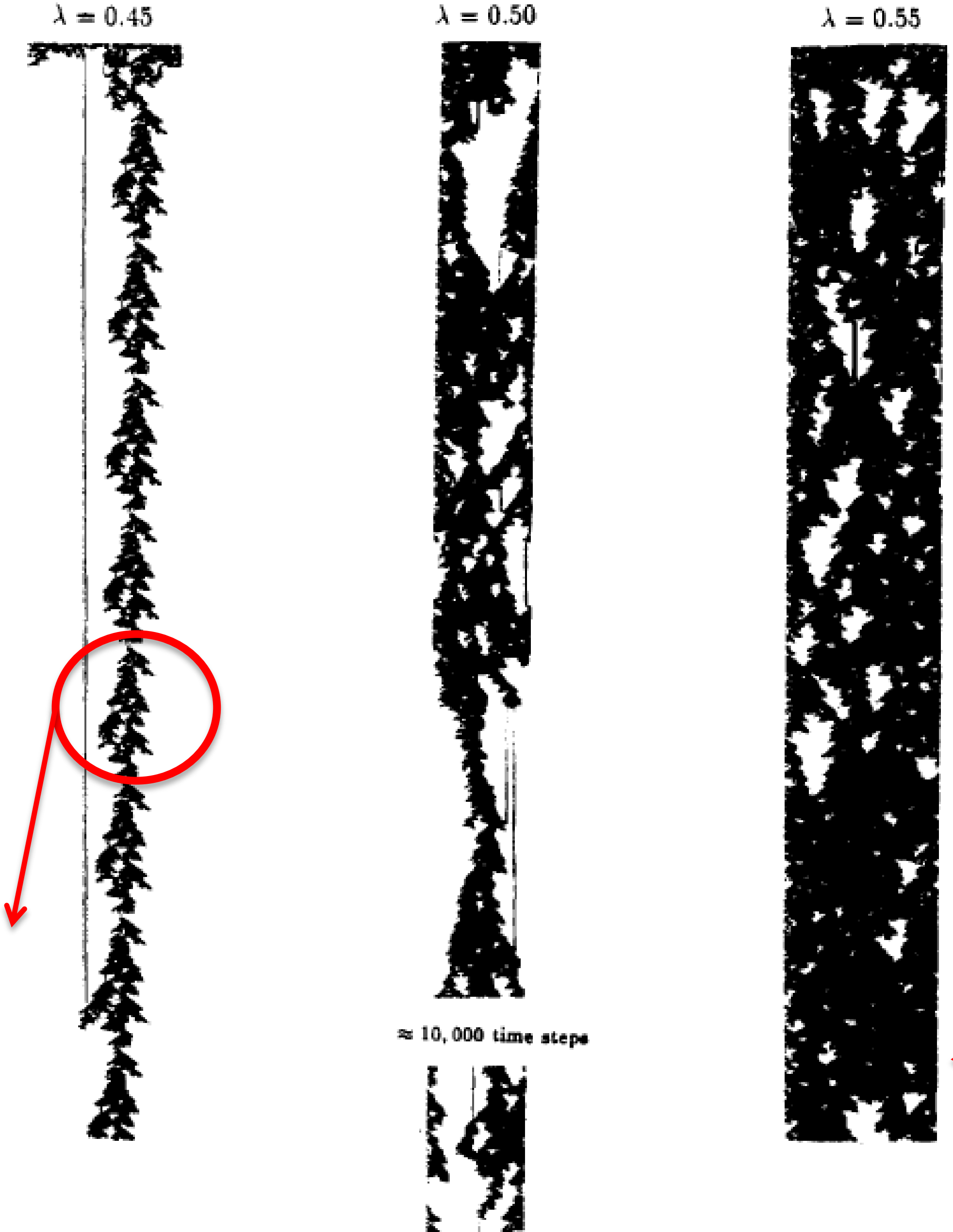


$\lambda = 0.35$



$\lambda = 0.40$





Rotation period = 14848

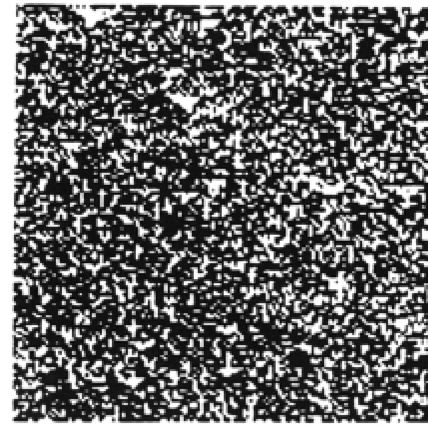
Complex structures

Chaos sets in

$\lambda = 0.60$

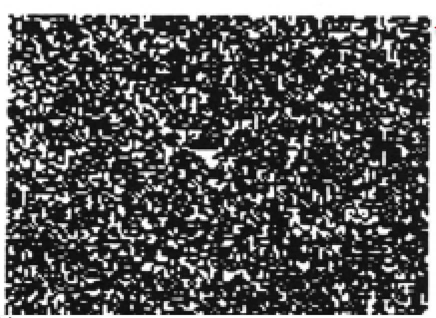
Chaos

$\lambda = 0.65$



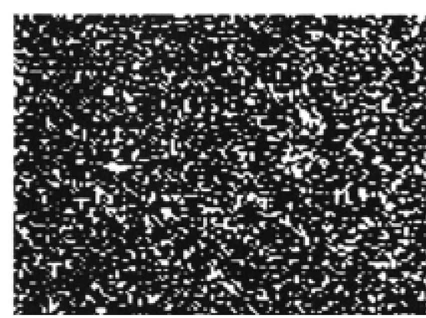
Chaos

$\lambda = 0.70$



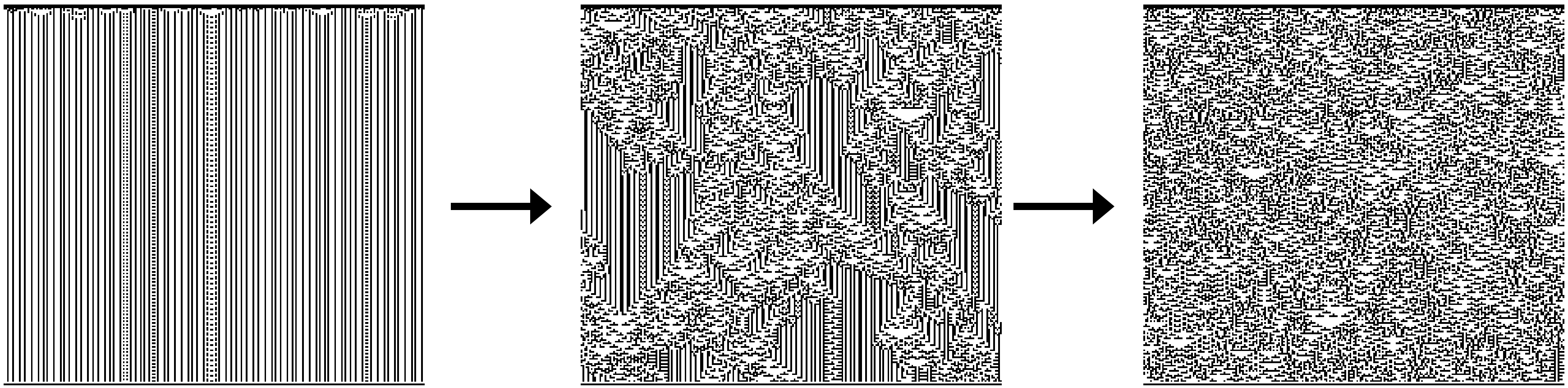
Chaos

$\lambda = 0.75$



Chaos

Variation in λ



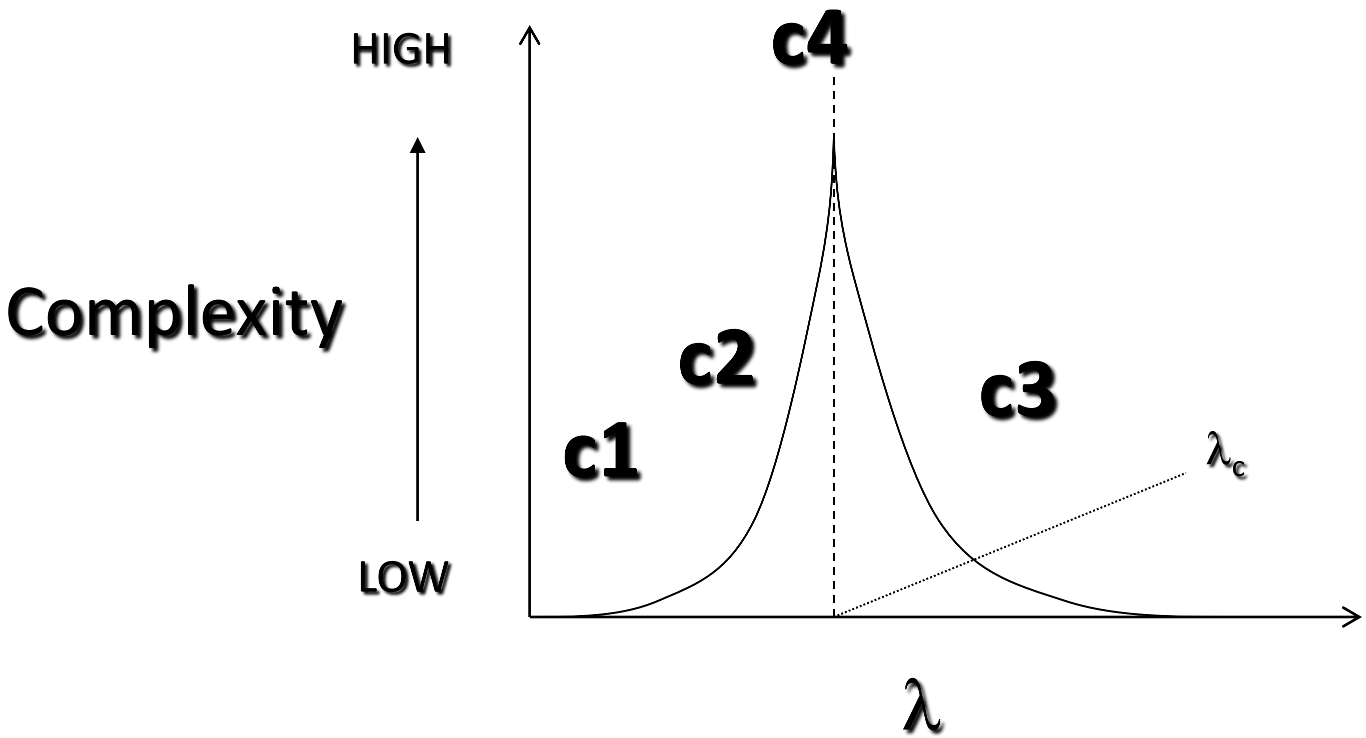
- **Dynamical behavior a function of increasing λ:**

  fixed-point → periodic → “complex” → chaotic

- **Analogous to Wolfram’s classes:**

  c1 → c2 → c4 → c3

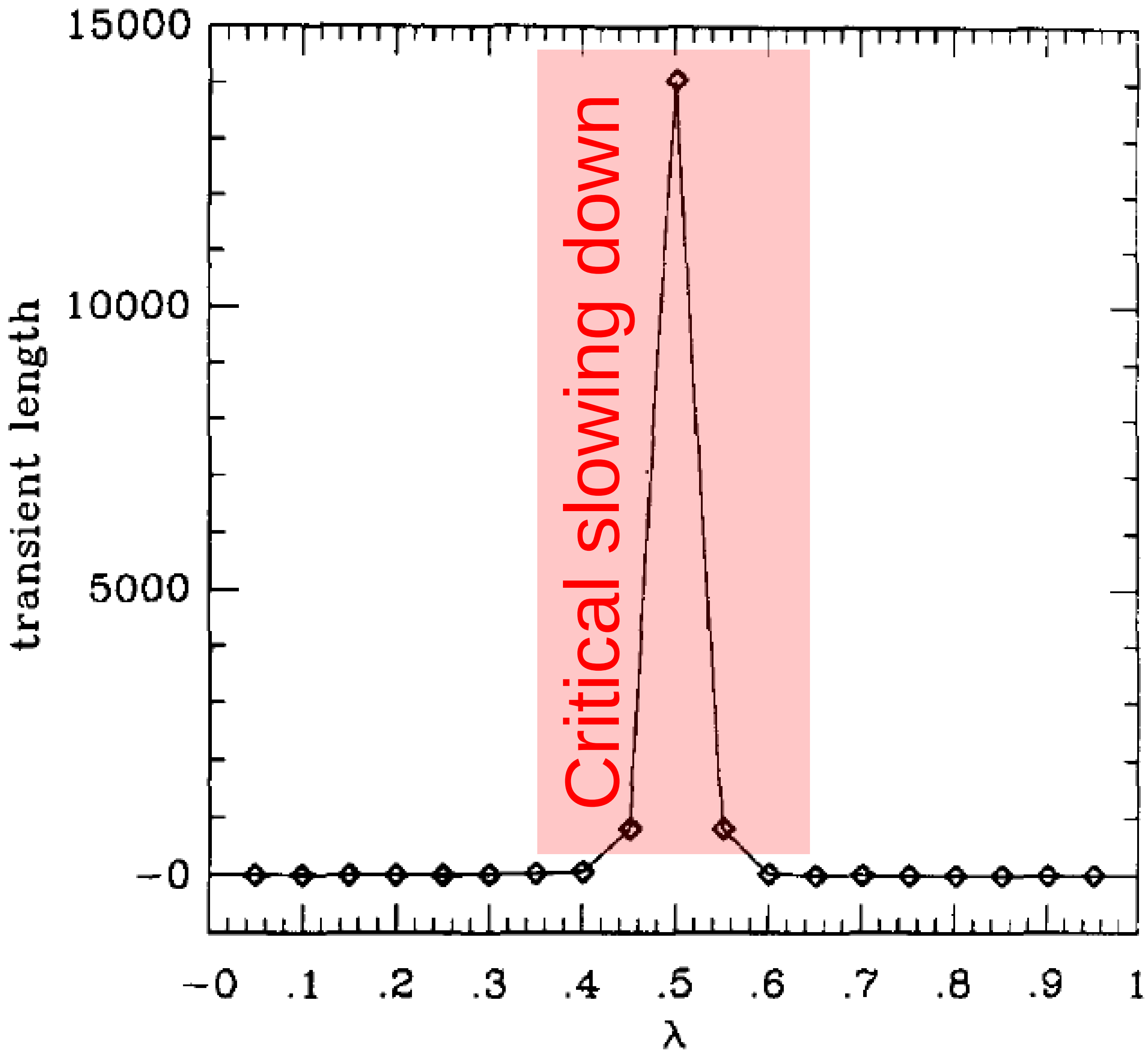
“Edge of chaos”



What is complex?

MEASURING COMPLEXITY

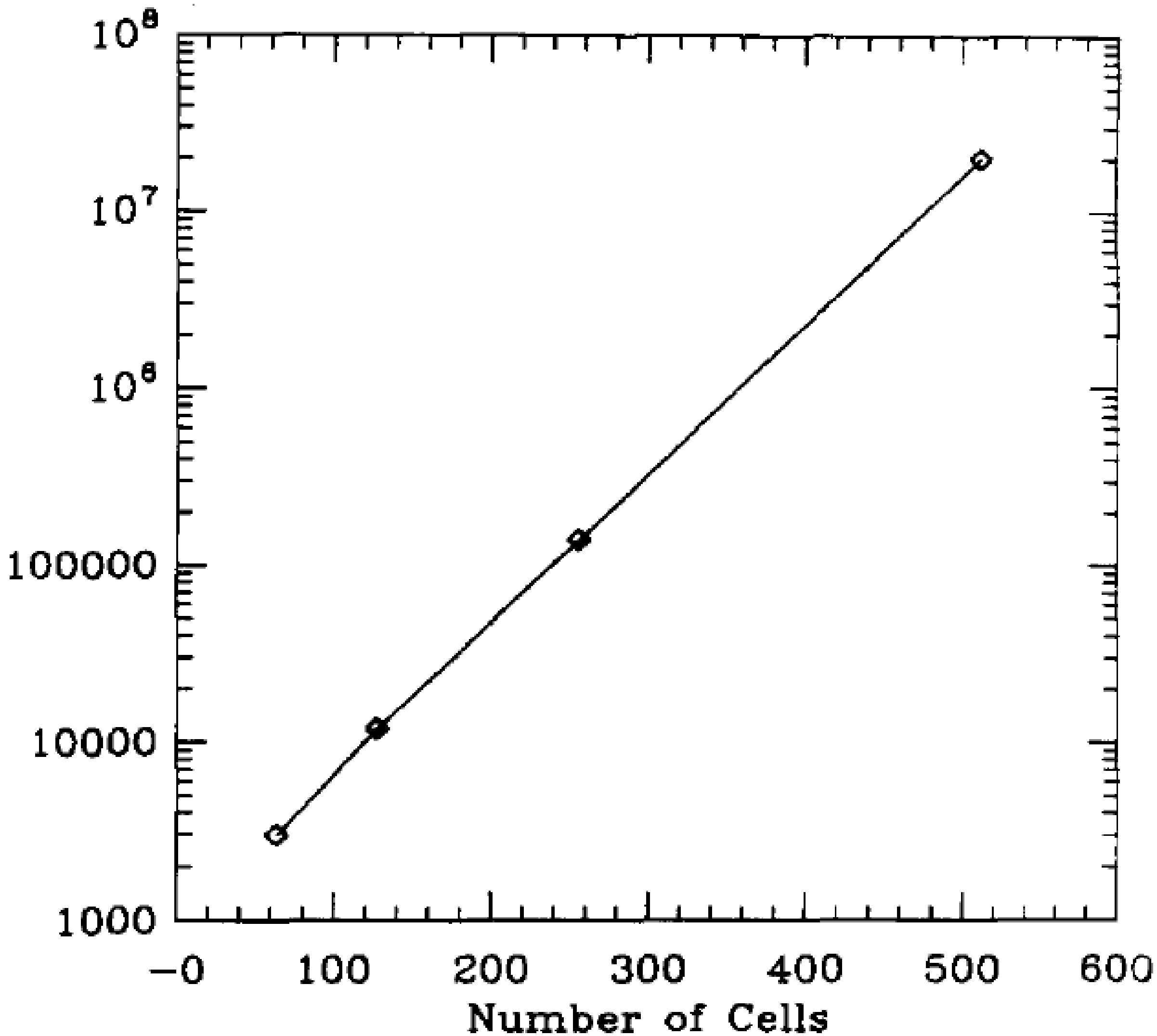
Transient length (Langton, 1990)



See also
Lindgren, Kristian, and Mats G. Nordahl. "Complexity measures and cellular automata." Complex Systems 2.4 (1988): 409-440.
https://www.complex-systems.com/pdf/02-4-2.pdf

Computational Science

Transient length grows with the system size



Shannon entropy

- **Information Theory:**
  - (Efficiently) **store** information
  - (Efficiently) **communicate** information
- **C.E. Shannon, "A Mathematical Theory of Communication", Bell System Technical Journal, vol. 27, 1948**

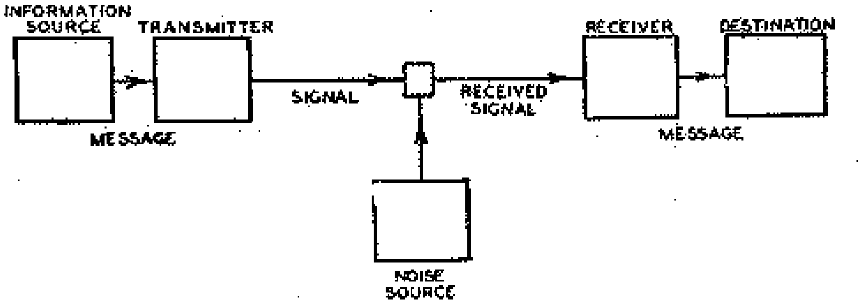


Fig. 1—Schematic diagram of a general communication system.

- **how to convert efficiently a message into a signal (transmitter)**
- **how to decipher efficiently the signal back into a message (receiver)**
- **how to cope with noisy environments which alter the signal**

Shannon entropy

- **Example: N coin flips**

x N

- **Represent a binary string… 1=H, 0=T**

$$b_1, b_2, \ldots, b_N$$

- **We want to communicate the outcome (binary string length N) to someone. What is the minimum of bits we can transmit so that we can reconstruct the message?**

Shannon entropy

- **Say: N = 1000**

- **Write down 01000100…011 representing result as bits***

- **If N = 10^{9}? Is there a more efficient way? Perhaps not…**
  - Each event is independent
  - Two equally likely outcomes
  - So really need to provide information for every event

- **If you're lucky and get all heads…**
  - can say we got 10^9 heads

***bit** = binary digit, coined down in 1948 by Shannon (originally Tukey in '37)

Computational Science

Shannon entropy

- **What if coin is biased? $p_0 = 1/3$ (Tail) and $p_1 = 2/3$ (Head)**
  - Most likely that our string has twice as many 1's as 0's

- **Consider 2 subsequent tosses (could look a 8 x 3-tuples also):**

$$p(b_i b_{i+1} = 00) = \frac{1}{3} \times \frac{1}{3} = \frac{1}{9}$$

a=00

$$p(b_i b_{i+1} = 01) = \frac{1}{3} \times \frac{2}{3} = \frac{2}{9}$$

b=01

$$p(b_i b_{i+1} = 10) = \frac{2}{3} \times \frac{1}{3} = \frac{2}{9}$$

c=10

$$p(b_i b_{i+1} = 11) = \frac{2}{3} \times \frac{2}{3} = \frac{4}{9}$$

d=11

- Rewrite in a,b,c,d form...
- **110010110111 ➔ dacdbd**
- No real gain… 500 x 2 bits

Computational Science

Shannon entropy

- **Intuition: give likely outcomes short codes, unlikely ones long codes**
  - d= 0, c =10, b=110, a=111

  - **1100101101111111 ➔ dacdbddd ➔ 0111100110100**
  - 16 bits ➔ 13 bits

- **On average we need:**

$$\frac{N}{2}\times(1p_d+2\times p_c+3\times p_b+3\times p_a)=\frac{N}{2}\times\left(\frac{4}{9}+\frac{4}{9}+\frac{6}{9}+\frac{3}{9}\right)=\frac{17}{18}N$$

Don't forget this is for just pairs, we could try encoding triplets, etc.

Shannon entropy

- **For a random variable *X* taking values in a finite set *X* with probability *p*, we call the entropy of *X*,**

$$H(\mathrm{X}) = -\sum_{x \in X} p(x) \log_2 p(x)$$

N i.i.d. (independent and identically distributed) random variables each with entropy H(X) can be compressed into more than NH(X) bits with negligible risk of information loss, as N tends to infinity; but conversely, if they are compressed into fewer than NH(X) bits it is virtually certain that information will be lost.

$H(b) = -(p_1 \log_2 p_1 + p_0 \log_2 p_0) = -(2/3 \log_2 (2/3) + 1/3 \log_2 (1/3)) = 0.918296..$
vs. $17/18 = 0.9444\ldots$

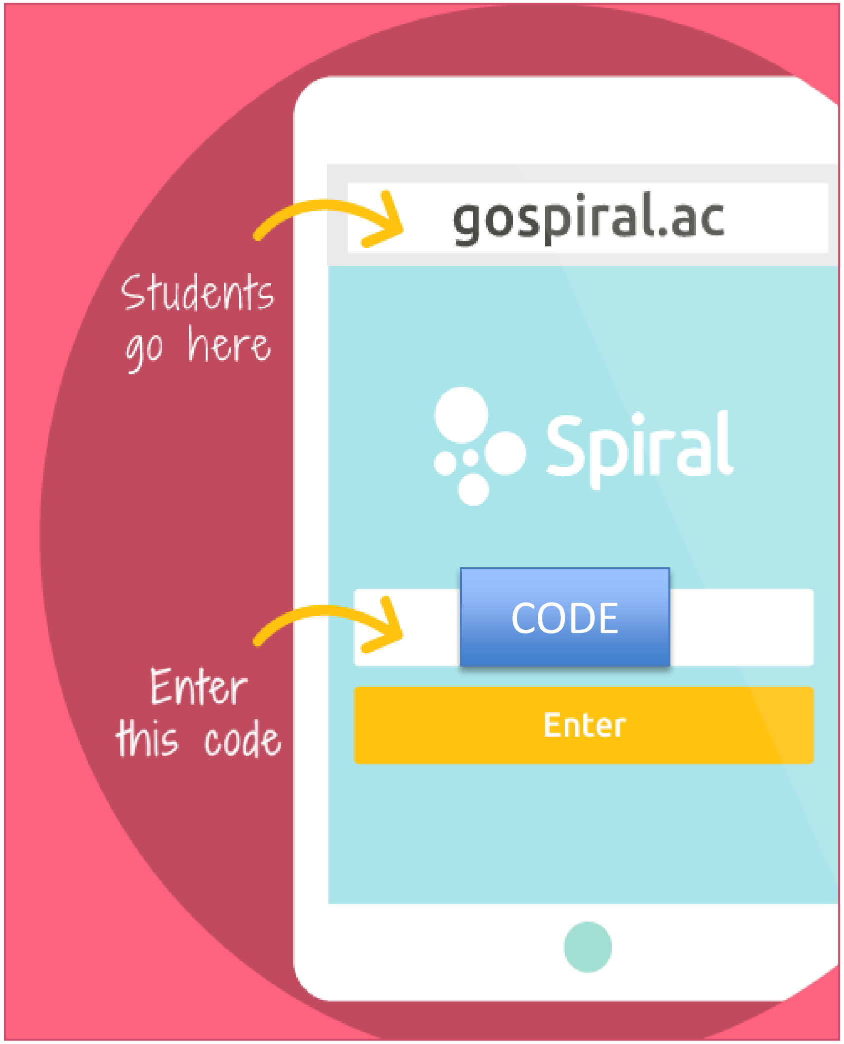
Shannon entropy

- **The amount of information being transmitted through the evolution of a CA could be a good measure of complexity.**
  - Or measure information at state t, t+1, t+2, t+3 … somehow
- **i.e., given the state *s(t)*, count neighborhood configurations**

Neighborhood	Count	p_i	$p_i \log_2 p_i$
111	1	1/12	-0.298746875
110	3	3/12	-0.5
101	1	1/12	-0.298746875
100	2	2/12	-0.430827083
011	3	3/12	-0.5
010	0	0/12	NA
001	2	2/12	-0.430827083
000	0	0/12	NA

$$H(\mathrm{X}) = -\sum_{i=1}^{k} p_i \log p_i = 2.459$$

Blackboard + Quiz: Q5



Shannon entropy (Langton 1990)

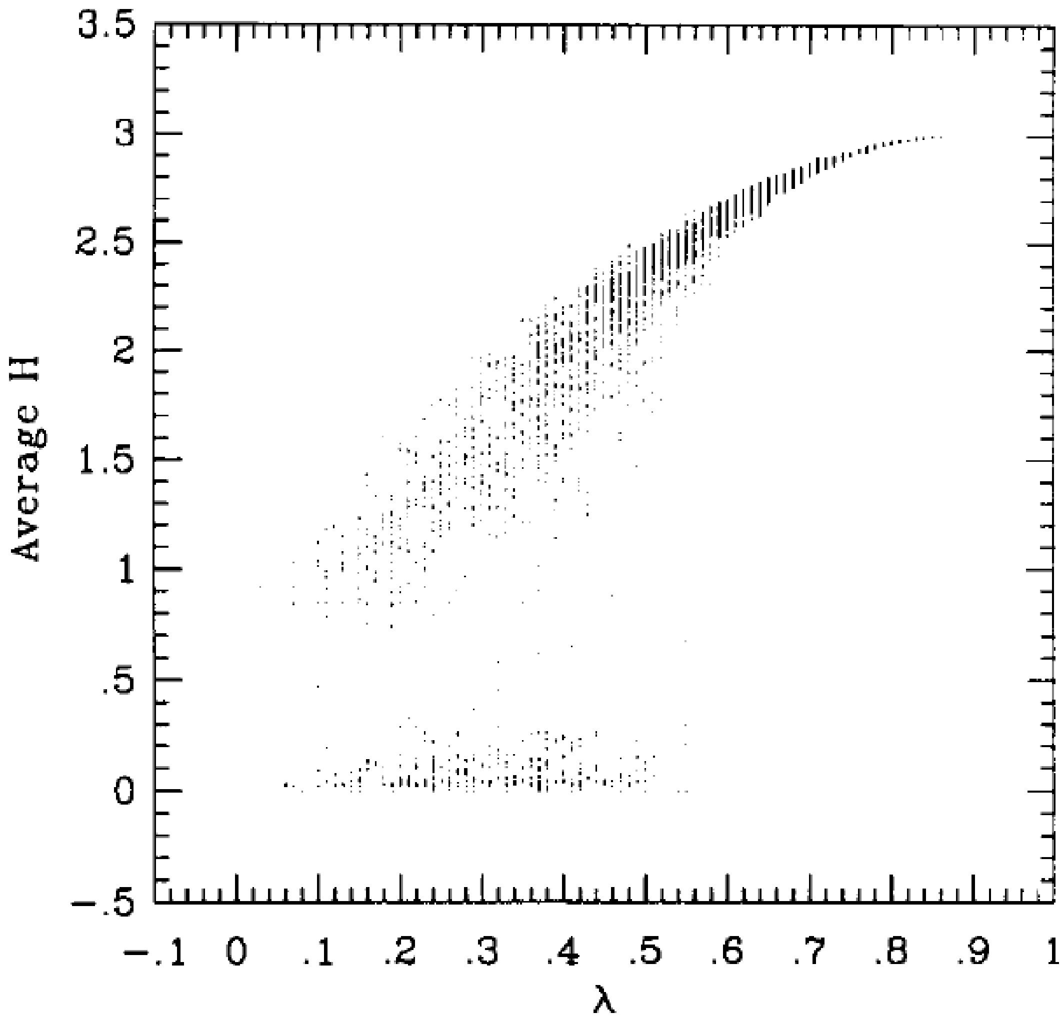


Fig. 6. Average single cell entropy $\overline{H}$ over λ space for approximately 10 000 CA runs. Each point represents a different transition function.

$$H(X) = -\sum_{i=1}^{k} p_i \log p_i$$

Generated by the random-table method

Shannon entropy (Langton 1990)

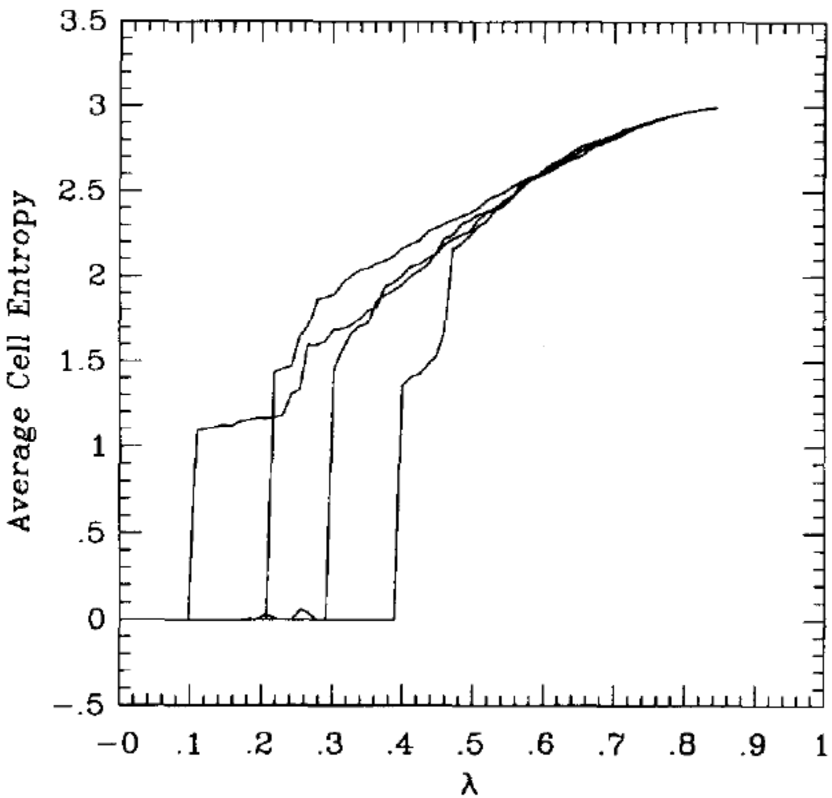


Fig. 7. Superposition of 4 transition events. Note the different λ values at which the transitions take place.

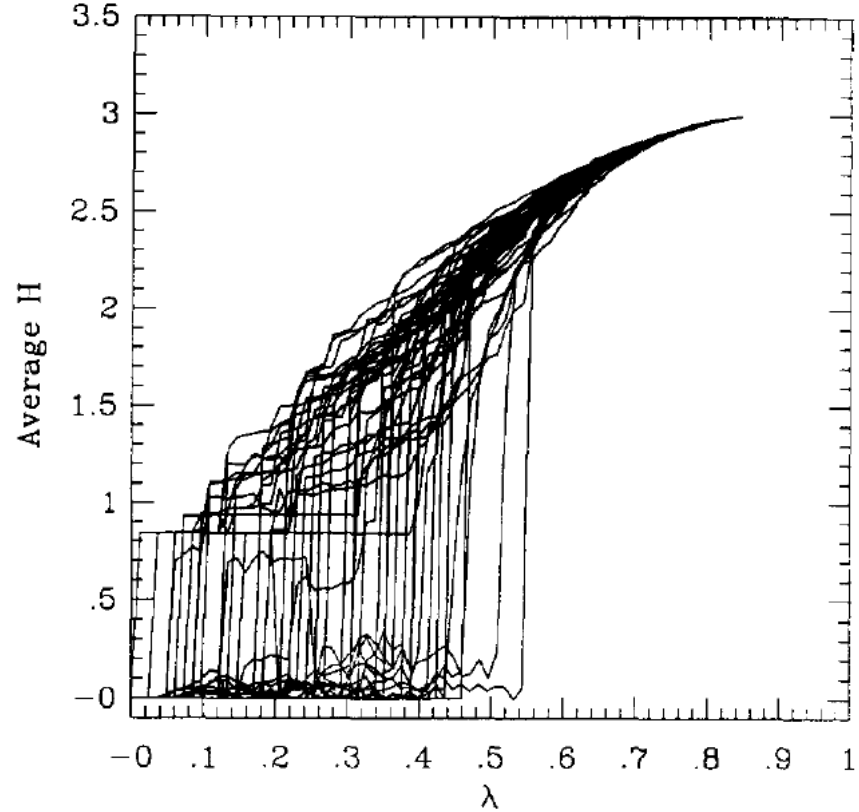


Fig. 8. Superposition of 50 transition events, showing the internal structure of fig. 6.

Generated by the table-walk-through method

NEXT: LECTURE ON GOL