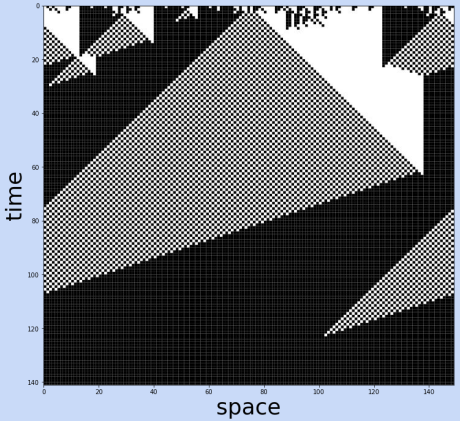
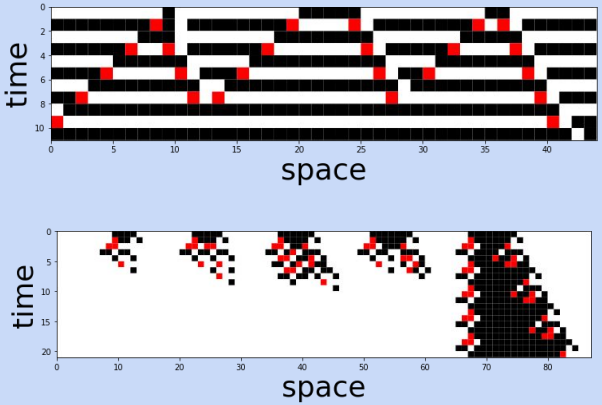
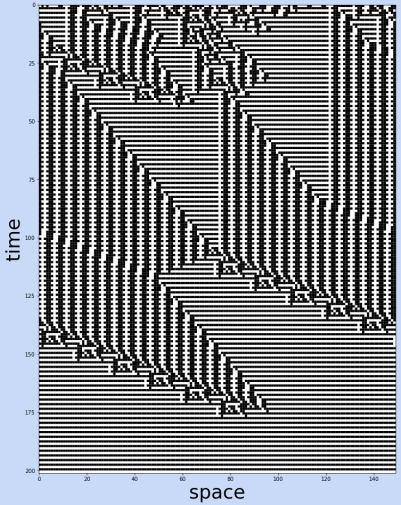
Evolving Cellular Automata







Eric Severson

NCASO Project Presentation

June 8, 2017

Project work based on a series of papers by Jim Crutchfield, Melanie Mitchell, et al. (1993-1998) http://csc.ucdavis.edu/~evca/evca1/papers.htm#EvCA

Additional thanks to Adam Rupe for his CA simulator python code

CA Framework

One dimensional CA operates on a lattice of *N* cells (with periodic boundary conditions) over a *k*-state alphabet

Each cell simultaneously updates its configuration based on the values in its neighborhood, which will be all cells within a fixed radius *r*

The CA rule is defined by the lookup table, which tells the cells how to update in each of the k^{2r+1} neighborhoods

i.e. rule 10010010 with $k = 2$ $r = 1$

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

Genetic Algorithm

Want to find CA that perform certain function

Rule space is too big to search exhaustively ($k^{k^{2r+1}}$ possible rules)

Use a Genetic Algorithm:

- Start with an initial population (size *p)* of rule strings
- Each generation evaluate the CA rules with a fitness function
- Some top scoring proportion *E* of the population is copied over
- Rest of the new generation is created by random crossover between two of these “elite” strings, along with a mutation in each rule site with probability *m*
- After running for *G* generations, hope to find CA rules that can perform the function well

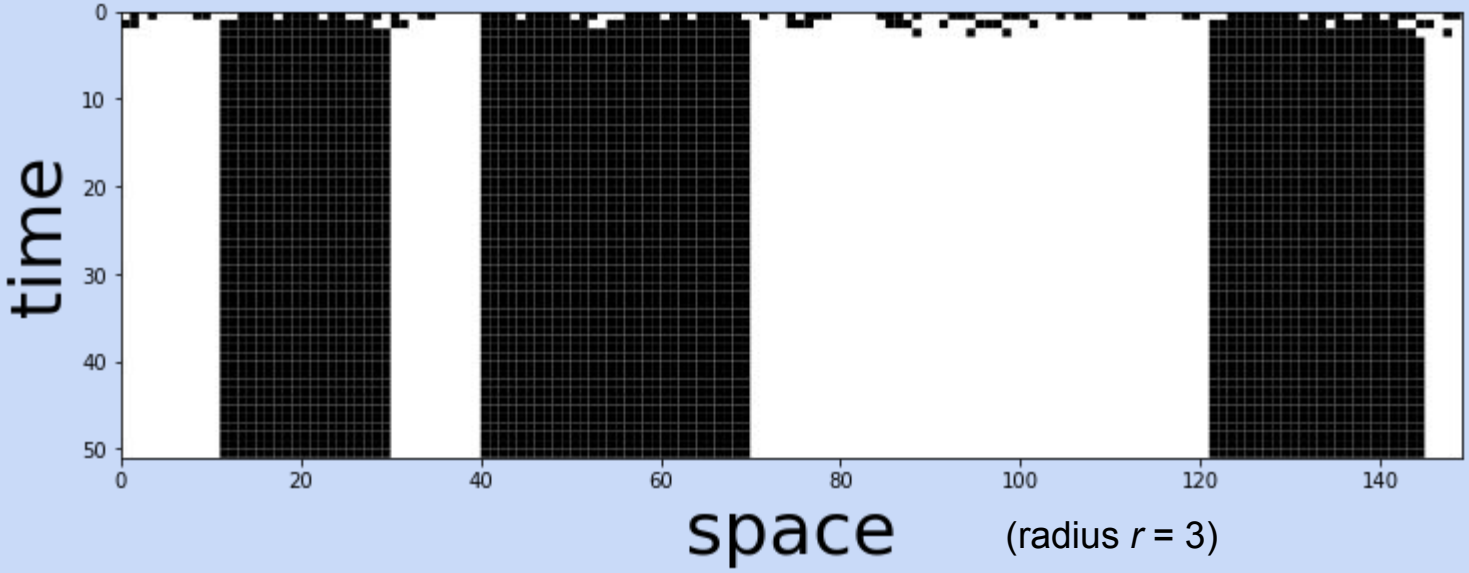
Majority Classification Task

Given an arbitrary initial configuration (*k=2)*, want the CA to recognize if the starting state is majority *0* or *1* by converging to a stable state of all *0* or all *1,* respectively

Non-trivial problem because of need to identify this global property given the relatively small fixed radius

Naive solution attempt: “Local Majority” Rule. Each cell conforms to the state that is the local majority in its neighborhood

Local Majority Rule

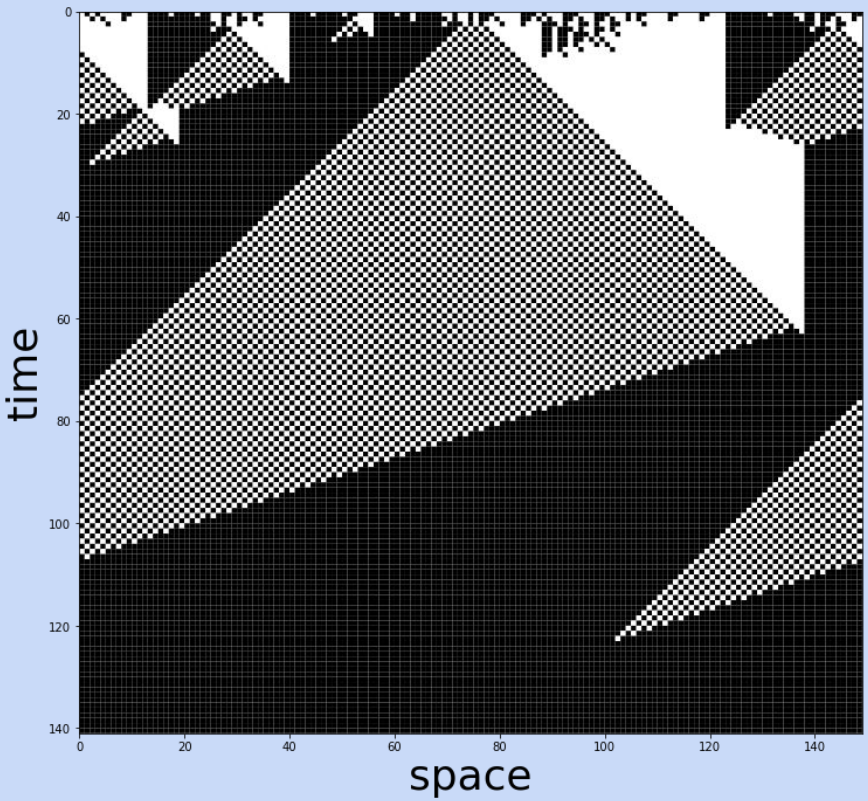


(radius $r = 3$)

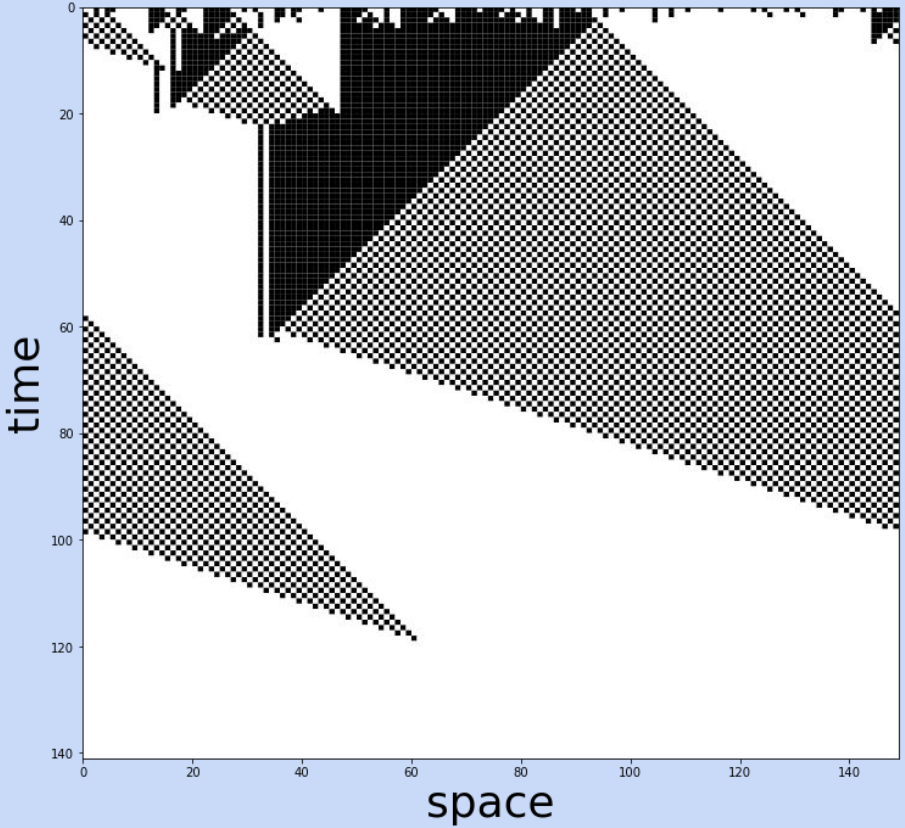
Reaches steady state with bands of all *0* or all *1*

GKL Rule

(Initial Condition A: 55% 1s)



(Initial Condition B: 45% 1s)



GKL Rule

$k = 2$ $r = 3$ CA invented by Gacs, Kurdyumov, and Levin (but not for this purpose)

Each cell updates to the majority between itself and the cell one and three spaces to the right (if the cell is a 1) or left (if the cell is a 0)

Solves majority classification through domain / particle interactions

Correctly classifies ≈ 81% of random length $N = 149$ initial conditions

(random initial conditions are binomially clustered around initial density 0.5, the hardest to classify)

Proven that a CA cannot solve the majority classification perfectly

Majority Classification GA

Fitness function takes i = 100 initial conditions (with uniformly distributed density) of size N = 149 and evolves them for t = *320* time steps using the given rule

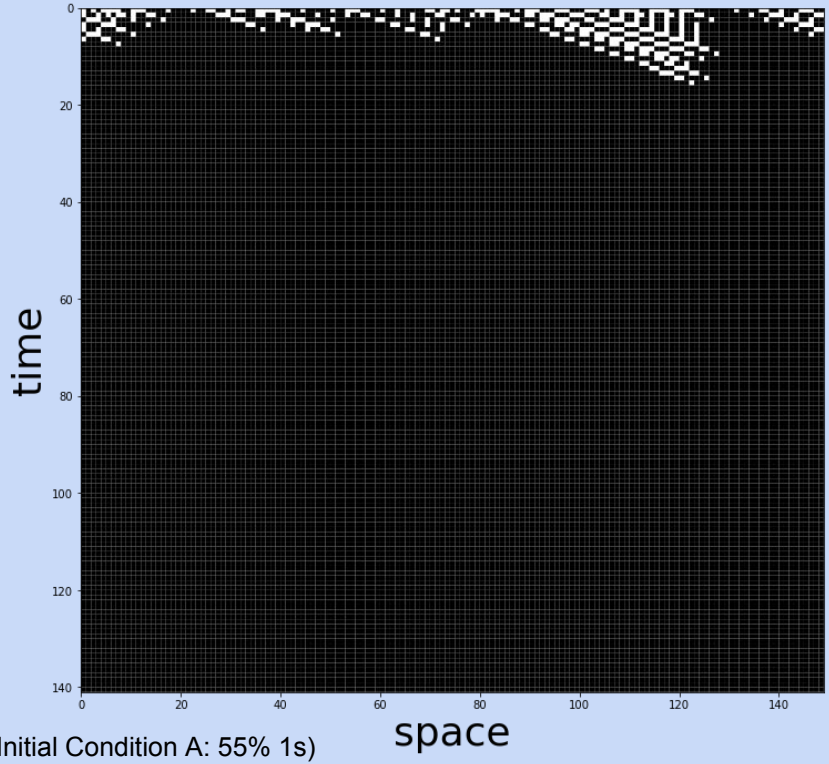
Fitness score is the proportion of these conditions that converge to the majority

Used population size p = 100, E = 20% of the population carried over each generation, mutation rate m = 0.032, and ran for G = 50 generations

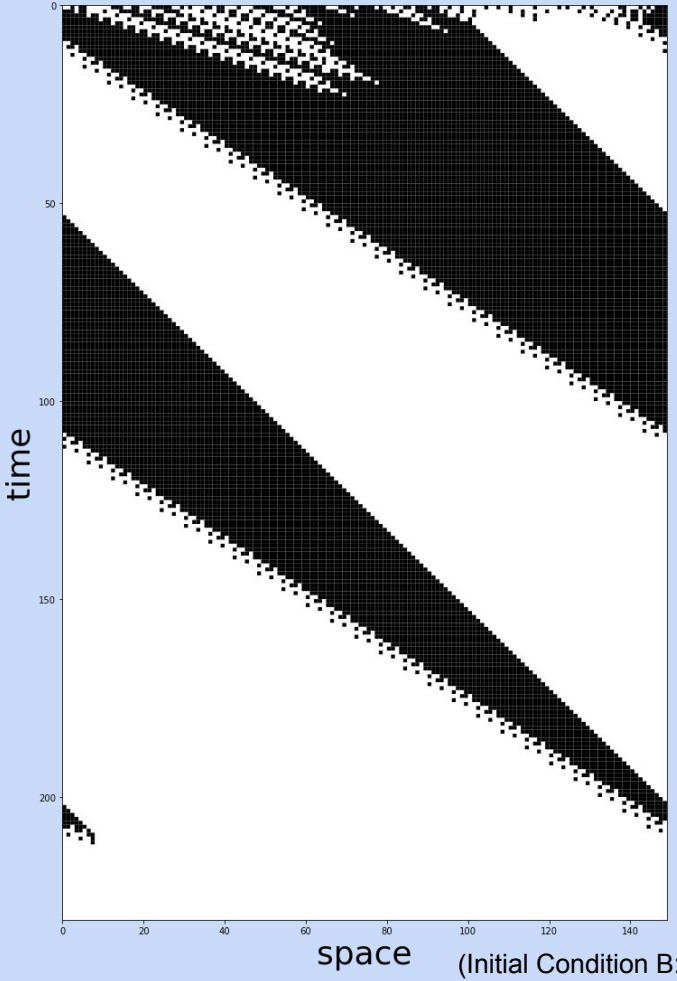
These parameters all similar to those used in the previous work

The successful GA runs all found block expanding strategies (13 of 15 runs, where the other 2 runs were stuck at a fitness of 0.5)

White Block Expanding CA

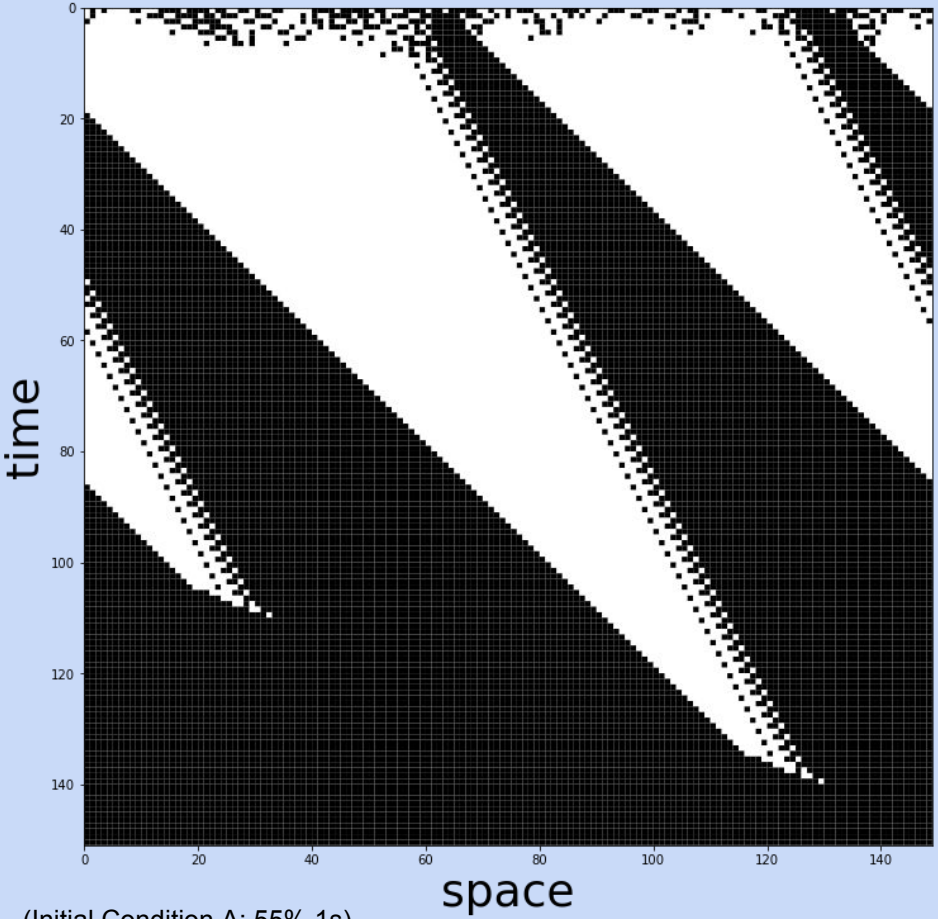


(Initial Condition A: 55% 1s)

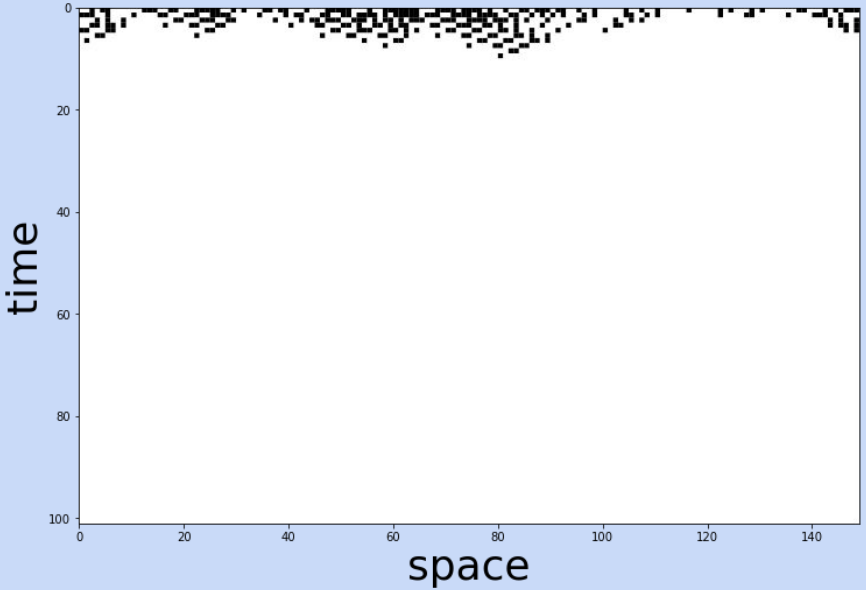


(Initial Condition B: 45% 1s)

Black Block Expanding CA



(Initial Condition A: 55% 1s)



(Initial Condition B: 45% 1s)

Block Expanding CAs

Asymmetric solutions where most neighborhoods converge to one color, but large blocks of the second color expand to eventually dominate the whole space

Relies on correlation between large blocks of a color existing and that color being the majority of the initial configuration

Does well on fitness sets with uniform initial densities, but worse on random initial configurations than GKL (above rules get ≈ 66% of random $N = 149$ initial states)

Not robust: as $N \to \infty$, success rate $\to 50\%$, unlike particle methods

Reference papers had evolved some more advanced particle strategies, but only on 9 of 300 GA runs

Synchronization Task

Given arbitrary initial condition (k=2) want the CA to reach a periodic state of all *0* followed by all *1*

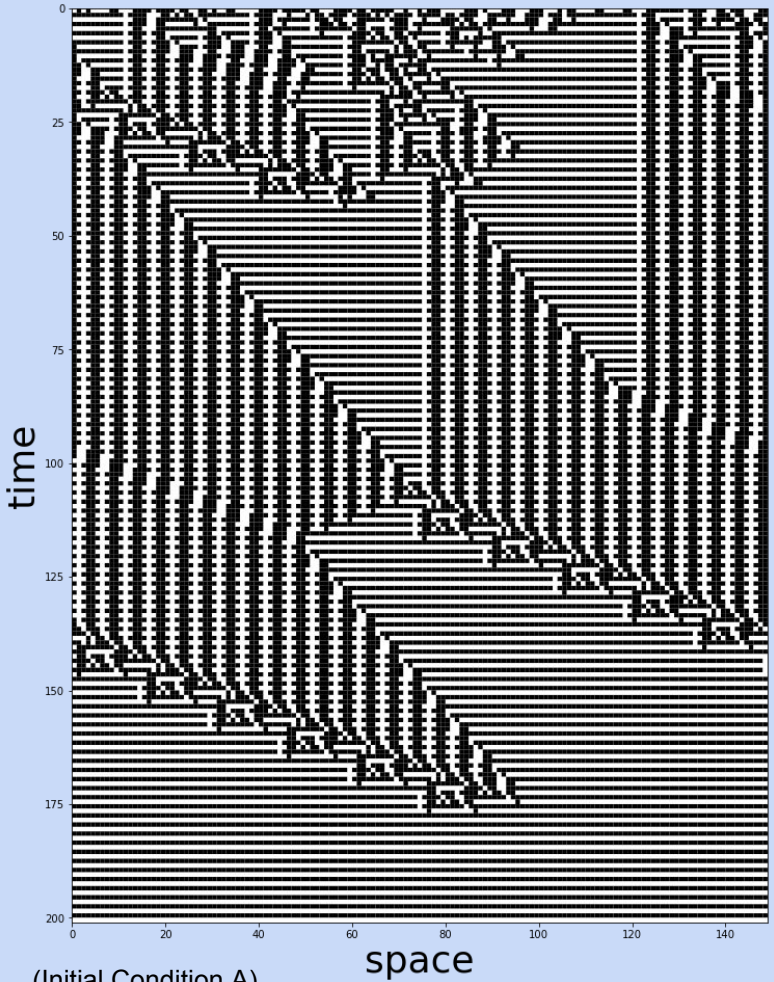
Non-trivial task because have to resolve phase discrepancies

Same parameters used as before, with changed fitness function

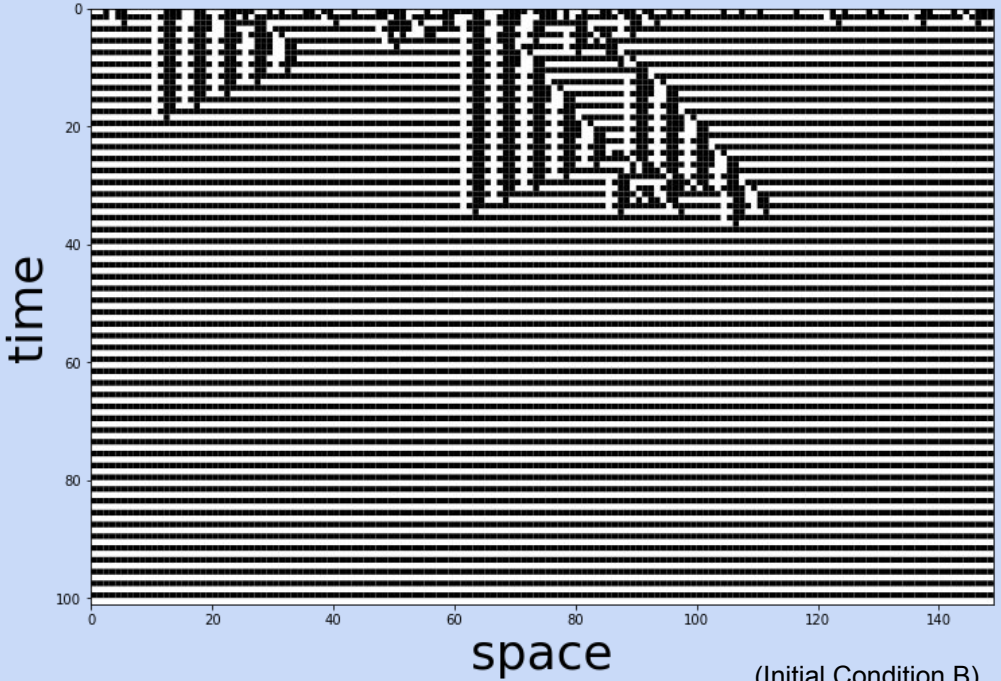
Found multiple solutions that achieved perfect fitness scores and also were able to perform perfectly on ≈ 100% of random N = 149 initial conditions

All successful solutions use domain / particle interactions to resolve these phase discrepancies

Evolved Synchronization CA

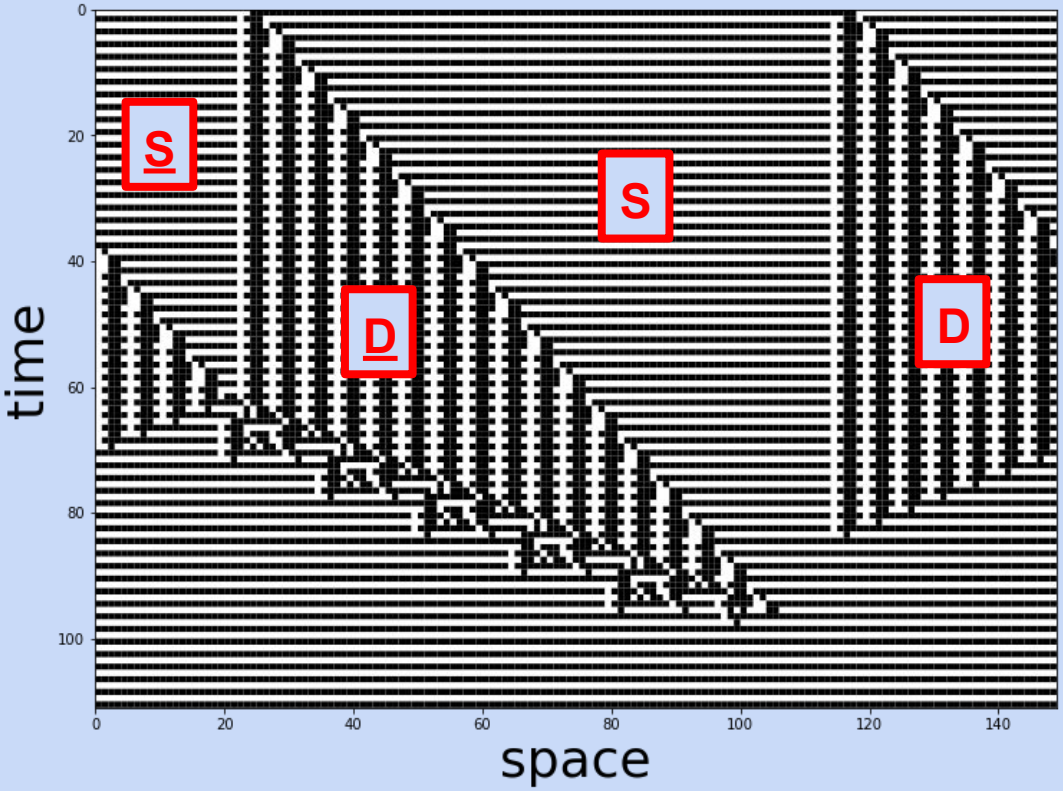


(Initial Condition A)



(Initial Condition B)

Domain Interactions Resolve Phase Discrepancies



SS : create D domain
SS : create D domain
SD or SD : period 2, velocity 0
DS or DS : period 6, velocity 5/6
DS or DS : period 2, velocity -5/2
SD or SD : period 6, velocity 5/2
D / D interactions depend on relative phase

D domain expands when between opposite phases, contracts when between same phases

Extensions to the Previous Work

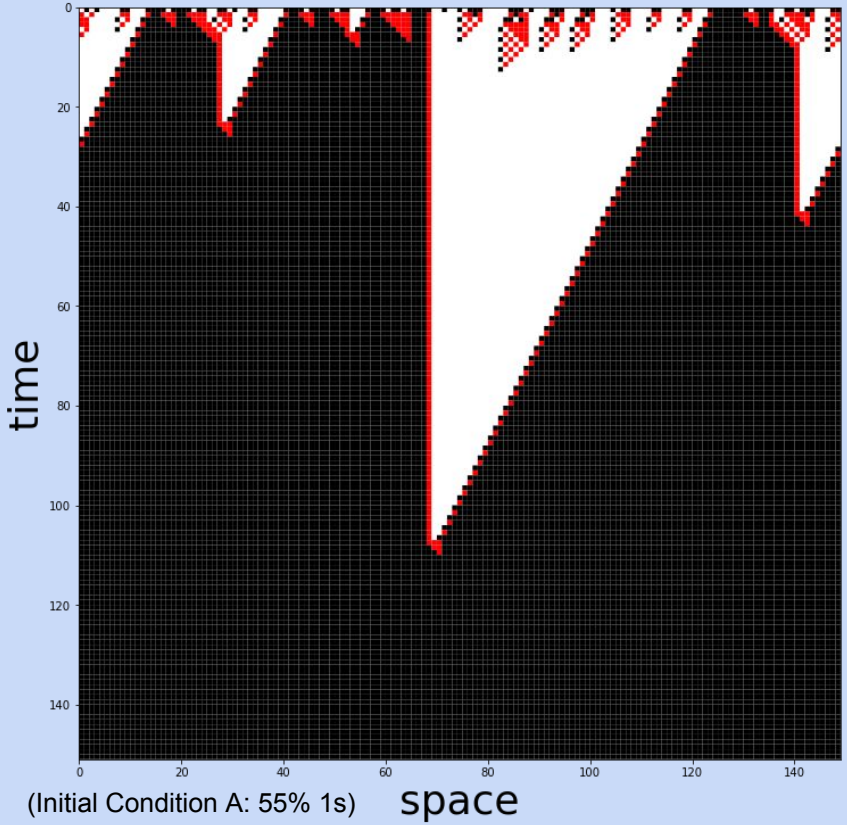
The above results all replicated what was found in the literature

Next I looked at increasing the alphabet size *k* to 3 for the above tasks

The initial conditions are still over the first two letters, and the fitness goals are still the same, but now the rule can use a third letter in its intermediate computation

To balance for the increased complexity of larger alphabet, looked at radius $r = 1$ and $r = 2$

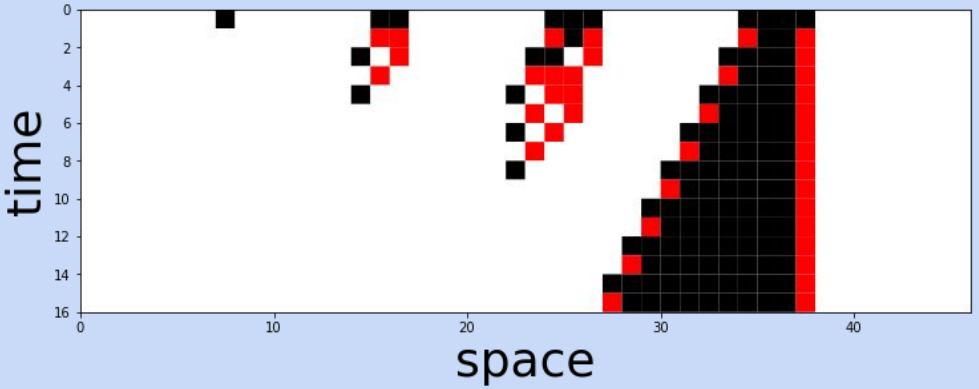
Majority Classification: *k = 3, r = 1*



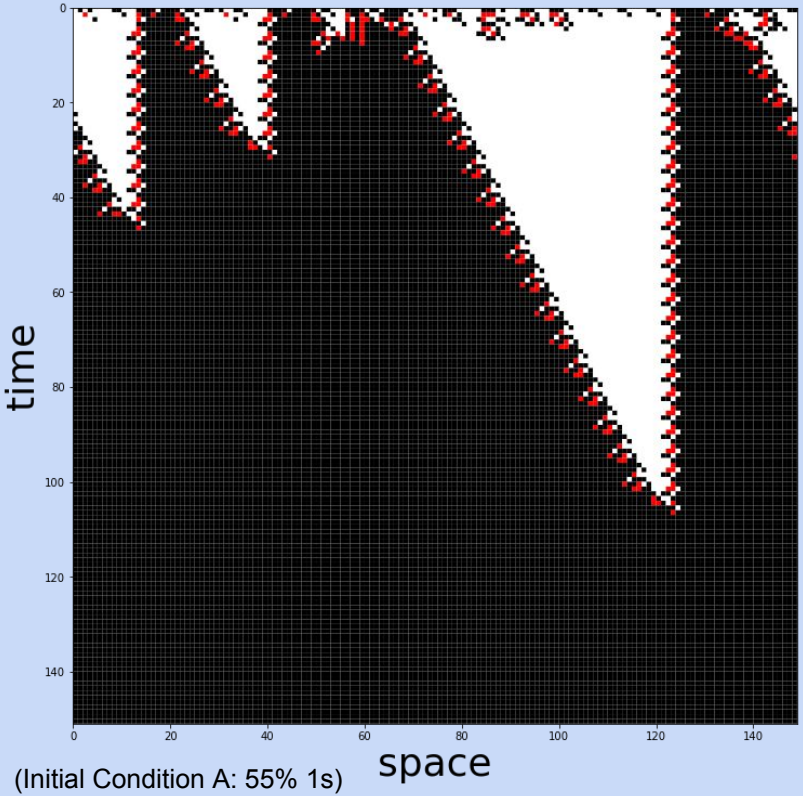
(Initial Condition A: 55% 1s)

Uses 3rd letter for better block expanding, only expands blocks of size 4+

Fitness score ≈ 0.78 (compared to 0.51 for best ECA), but still only gets 50% of random initial conditions



Majority Classification: *k* = 3, *r* = 2

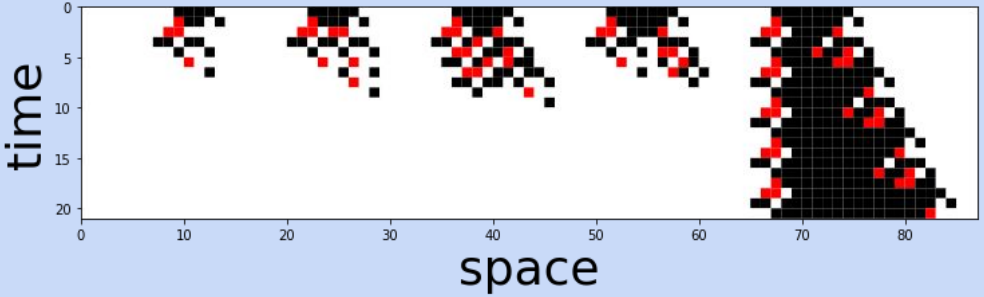


(Initial Condition A: 55% 1s)

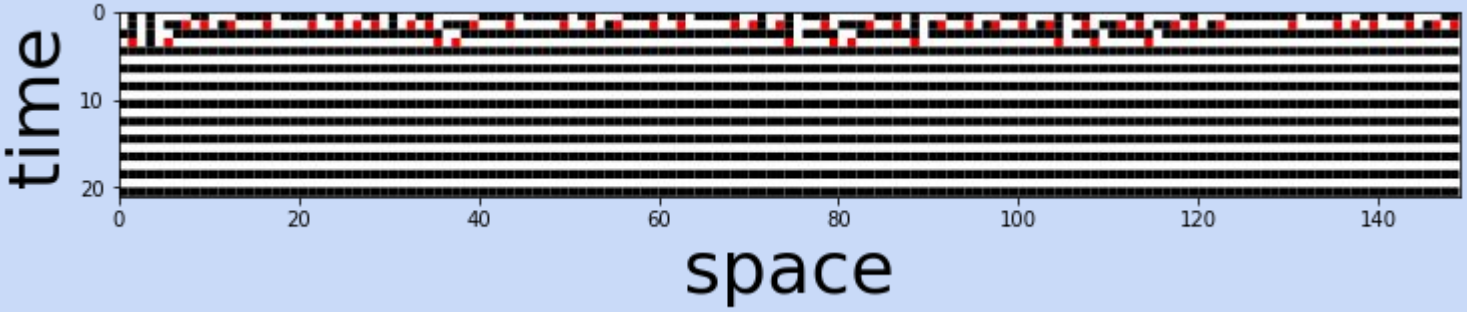
More improved block expanding

Fitness score ≈ 0.93 and classifies ≈ 65% of random states (comparable to my best *k* = 2 *r* = 3 block expanding scores)

k = 2 *r* = 2 trial had best fitness ≈ 0.85 and ≈ 54% of random states

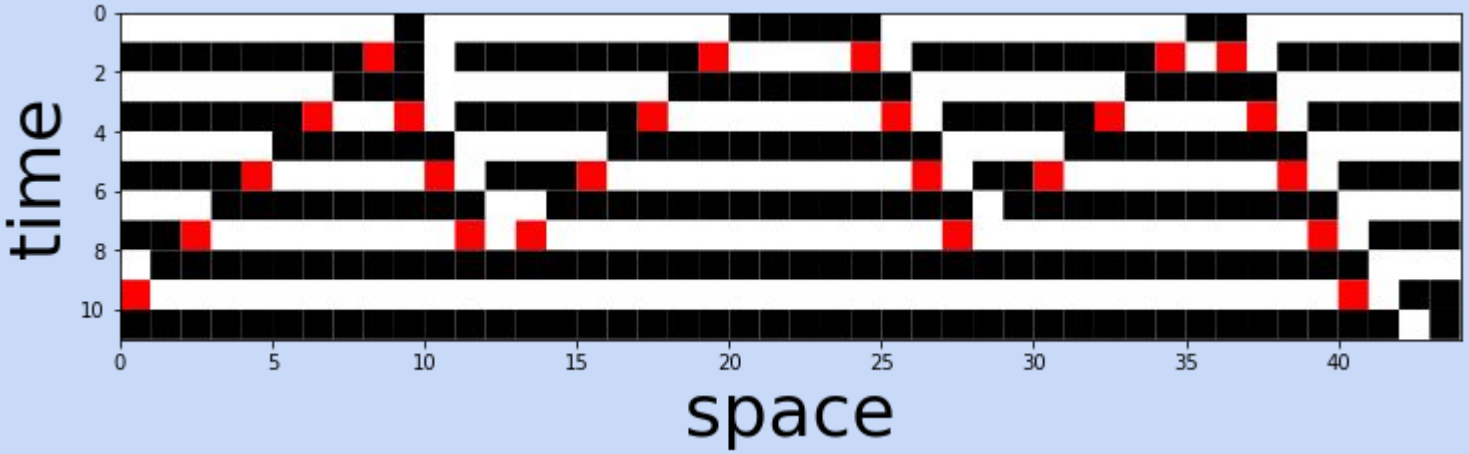


Synchronization: $k = 3$, $r = 1$



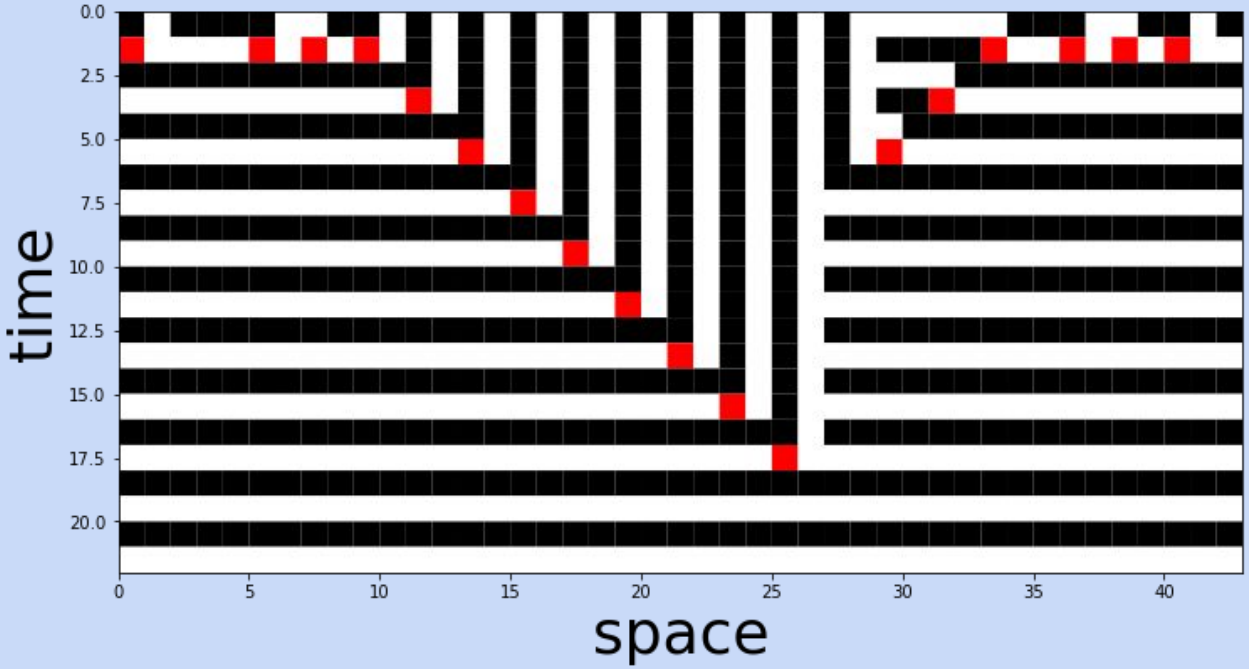
Multiple similar rules evolved that achieved perfect fitness scores and correct classification on ≈ 100% of initial conditions

Synchronizes extremely fast without having to propagate information all the way across the lattice



This rule always syncs the initial row to be the black part of the phase

The third letter is used at the boundaries in such a way that the in phase domain regions always expands out and the out of phase regions. It is only used on rows when the correct phase is white and incorrect phase is black, and acts to shield the incorrect phase from affecting surrounding cell



(10)* is another domain, which is annihilated on the left by the correct phase domain

For the synchronization task, we see that increasing the alphabet creates a qualitatively different type of solution, which is much more efficient

Future Goals

Improve program speed by writing the code in C instead of Python, and set the code up to run in parallel. Results from this project were strongly limited by computer hours, and my GA data sets were smaller than the original papers

Move to 2-dimensional CAs

Explore new fitness tasks. Want to try a new letter of static cells in the initial condition that the CA has to evolve around (i.e. a CA trying to propagate through a maze or connect nodes like in the Travelling Salesman Problem)

Catch up on the 20 years of relevant literature between now and the cited papers

References

Melanie Mitchell, Peter T. Hraber, and James P. Crutchfield, "Revisiting the Edge of Chaos: Evolving Cellular Automata to Perform Computations", Complex Systems 7 (1993) 89-130

Melanie Mitchell, James P. Crutchfield, and Peter T. Hraber, "Dynamics, Computation, and the 'Edge of Chaos': A Re-Examination," In Complexity: Metaphors, Models, and Reality, G. A. Cowan, D. Pines, and D. Meltzer (eds.), Santa Fe Institute Studies in the Sciences of Complexity, Proceedings Volume 19, Addison-Wesley (1994) 497-513.

Melanie Mitchell, James P. Crutchfield, and Peter T. Hraber, "Evolving Cellular Automata to Perform Computations: Mechanisms and Impediments", Physica D 75 (1994) 361-391.

James P. Crutchfield and Melanie Mitchell, "The Evolution of Emergent Computation", Proceedings of the National Academy of Sciences, USA 92:23 (1995) 10742-10746.

Rajarshi Das, Melanie Mitchell, and James P. Crutchfield, "A Genetic Algorithm Discovers Particle-Based Computation in Cellular Automata", In Parallel Problem Solving from Nature-III, Y. Davidor, H.-P. Schwefel, and R. Männer (eds.), Springer-Verlag (1994) 344-353.

Rajarshi Das, James P. Crutchfield, Melanie Mitchell, and James E. Hanson, "Evolving Globally Synchronized Cellular Automata", In Proceedings of the Sixth International Conference on Genetic Algorithms, L. J. Eshelman (ed.), Morgan Kaufmann (1995) 336-343.

Melanie Mitchell, James P. Crutchfield, and Rajarshi Das, "Evolving Cellular Automata with Genetic Algorithms: A Review of Recent Work", In Proceedings of the First International Conference on Evolutionary Computation and Its Applications (EvCA'96), Russian Academy of Sciences (1996).

Wim Hordijk, James P. Crutchfield, and Melanie Mitchell, "Embedded-Particle Computation in Evolved Cellular Automata", In Physics and Computation '96 (Pre-proceedings), T. Toffoli, M Biafore, and J. Leão (eds.), New England Complex Systems Institute, (1996) 153-158.

Wim Hordijk, James P. Crutchfield, and Melanie Mitchell, "Mechanisms of Emergent Computation in Cellular Automata", In Parallel Problem Solving from Nature-V, A. E. Eiben, T. Bäck, M. Schoenauer, and H.-P. Schwefel (eds.), Springer-Verlag (1998) 613-622.

James P. Crutchfield. Melanie Mitchell, and Rajarshi Das, "The Evolutionary Design of Collective Computation in Cellular Automata", Machine Learning Journal, submitted.

Land, Mark; Belew, Richard (1995). "No perfect two-state cellular automata for density classification exists". *Physical Review Letters*. 74 (25): 1548–1550.