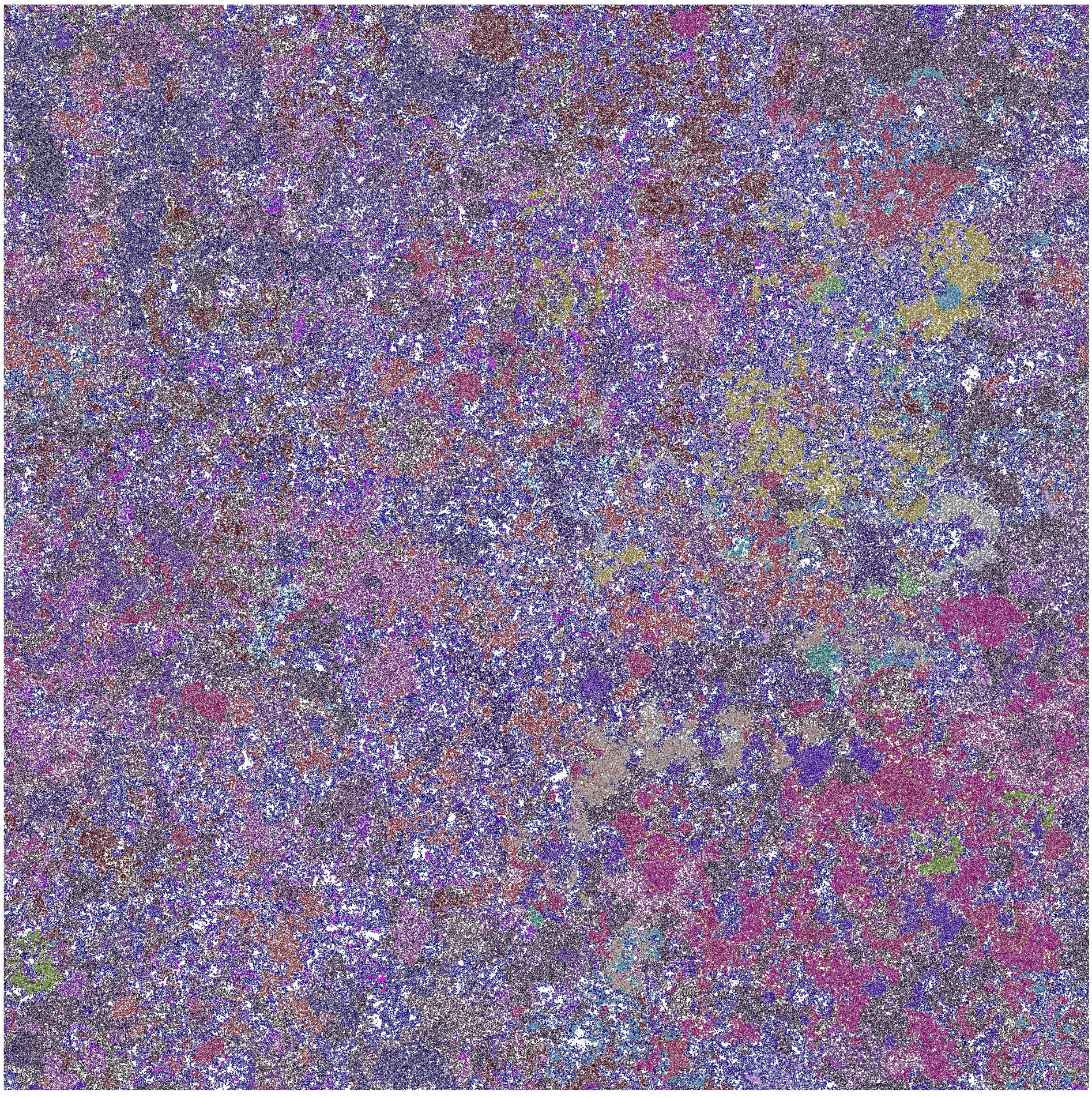
Nu-life

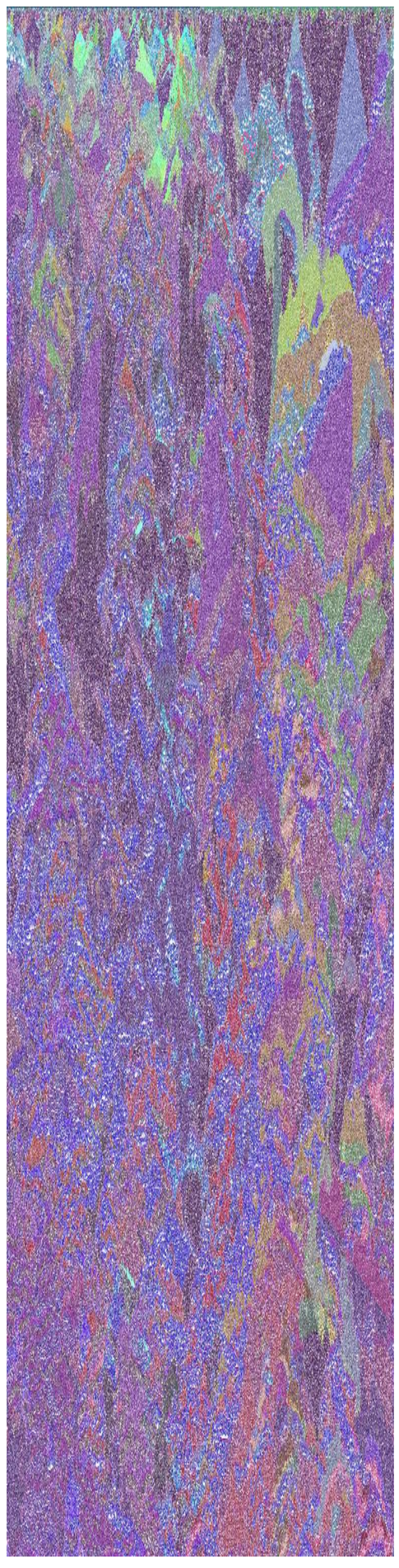
Spontaneous Dynamic Hierarchical Organization in a Non-uniform "Life-like" Cellular Automaton

Ben Cole (ben-c@moving-picture.com)
Michael Muthukrishna (michael@psych.ubc.ca)

We present a novel 2D cellular automaton with rules that are a non-uniform generalization of a Moore-neighbourhood, outer-totalistic, two-state ("life-like") cellular automaton. The system is purely deterministic and exhibits interesting multi-scale emergent behaviour, including the spontaneous formation of mobile particles and other self-organizing structures. In particular, smaller-scale structures can be shown to combine with other structures to form inhomogeneous higher-order constructions, and to do so at multiple orders of magnitude. The system has features in common with reaction-diffusion models. We propose that this system has properties that make it useful as a model of an artificial chemistry with the potential for supporting open-ended evolutionary growth. We call it Nu-life. (**nulifeautomata.org)**



The state of the system after 10,000 steps on a 3072x3072 grid starting from random initial conditions. Different colours represent different rules, with the red component indicating high-density rules (6 – 8 active neighbours), blue medium density (3 – 5) and green low density (0 – 2). White cells are active at all densities. Patterns of interwoven rules can be seen at multiple scales.



1-D time-slice of the same 3072x3072 simulation, again indicating recursive nature of pattern formation.

```
#Initialize grid
grid_and_rules = random_bool((grid_x,grid_y,rule_layers + 1))

for each timestep:
    #Clear rules for inactive cells
    for x in rule_layers:
        grid_and_rules[:,:,x] &= grid_and_rules[:,:,state_layer]

    #Sum up densities for both cell states and rules
    accum = zeros( (grid_x,grid_y,rule_layers + 1) )

    for dx,dy in neighbourhood:
        left = roll(grid,dx,axis = x_axis)
        up = roll(left,dy,axis = y_axis)
        accum += up

    cell_density = accum[:,:,state_layer]
    cell_state = grid_and_rules[:,:,state_layer]

    #Determine new rules for each cell:
    #An active cell picks only rules shared by its active neighbours and itself.
    #An inactive cell picks any rules used by its active neighbours.
    for x in rule_layers:
        ac = accum[:,:,x]
        grid_and_rules[:,:,x] = ((ac > 0) & (cell_density > 0)) &
                                ((ac == cell_density) | (cell_state == 0))

    on_rules = grid_and_rules[:,:,on_layers]
    off_rules = grid_and_rules[:,:,off_layers]

    output_states = zeros( (grid_x,grid_y))

    #Perform standard "life-like" outer-totalistic rules
    for x in densities:
        on_r = on_rules[:,:,x]
        off_r = off_rules[:,:,x]

        output_states |= (cell_state == 1) & (cell_density == (on_r * (x + 1)))
        output_states |= (cell_state == 0) & ((cell_density + 1) == (off_r * (x + 1)))

    grid_and_rules[:,:,state_layer] = output_states
```


Graphs showing proportion of cells with each rule. (A), (C) and (D) exclude counts for empty cells (approx. 0.33) and the rule with all densities active (approx 0.42). (A) shows distribution on frame 10000. (B) shows correlation on that frame of rules in neighbouring cells, with y-axis as square-root of frequency of occurrence relative to chance. (C) shows rule evolution over time recorded every 1000 frames. (D) shows 50000 frames on a 1536x1536 grid, in 1000 frame increments. In (A), (C) and (D), rules are converted to binary (in range 0 – 2^{18}), ordered so that the bits for survival are most significant, with bits for higher densities more significant than lower densities. The vast majority of active cells (when they don't have all bits active) have no birth rules. This observation is used in (B) where only survival rules are used to calculate index (range 0 - 2^9).