A. Terminology

1. State = value of a single cell, e.g, 1 or 0
2. k = number of states, e.g., 2
3. radius = r
4. state at location i is s^i
5. configuration = total configuration of states across entire string = global state
6. density of 1s in configuration **s** is rho(**s**). = #1s / total # of locations
7. phi = the ‘equations of motion’ = the rule = the mapping from set of locations to new value for a location
8. rule = string or ‘chromesome’ or ‘gene’ that specifies the output values of the phi equations.
9. eta = the neighborhood = 2 * r + 1
10. So the output bit for state is: s = phi(eta); see p. 3 for notation for describing output for a particular location.
11. majority rule = it’s the rule that just says each bit in the neighborhood gets 1 vote, and the majority wins for the output.
12. lambda for a rule = fraction of 1s in the output bits of its rule phi.
    1. Ex, lambda for majority rule is 0.5.
13. rho-zero is shorthand for rho(s_0), i.e., the density of 1s at time step 0 of an IC.
    1. So rho(s_5) would be the density of 1s at time step 5 of an IC.
    2. So fixed points are rho(s_infinity) = 0 and rho(s_infinity) = 1
14. the “rho_c = 1/2” task: for a given IC, if rho_0 < ½, then after 200 iterations, the CA should be at all 0s; if rho_0 > ½, then after 200 iterations the CA should be at all 1s.
15. N = the size of the IC, i.e., the ‘size of the lattice’, i.e., the number of bits in the IC.

B. Mitchell’s fitness function (“performance fitness”)

(i) create pool of random ICs such that half of them have rho_0 < ½ and half have rho_0 > ½.
(ii) for each rule (i.e. CA) in the population, run it on each IC for 200 steps.
(iii) The CA receives +1 fitness IC it classifies correctly, and 0 if it does not.

C. The contents of the paper including figures.

1. Introduction

2. Results
2.1. Comparing r = 2 and r = 3

1. Setup:
    1. Population: 100
    2. Generations: 50
    3. Iterations: 200 (at most)
    4. IC length: 121
    5. Radius: 2 and 3
2. Compare radius 2 and radius 3
    1. Figures:
        1. At least 2 space-time diagrams (waterfall plot) (Mitchell figure 1) for some interesting rules.

2. Two figures: Choose some a well-performing rule from each of r = 2 and 3, and graph performance of rule as a function of rho-0 on ICs, perhaps on ICs of different lengths.
3. Two figures: Take a typical run from each of r = 2 and r = 3, and graph best fitness (y axis) by generation (x axis). We can identify epochs in caption, I think.
4. 1 full page figure with histograms showing frequency of elite rules by lambda values for those rules. E.g., count number of rules with 0/N 1s, 1/N 1s, 2/N 1s, etc.
5. A plot comparing length of transients (i.e., # of bit flips in a column of the waterfall plot) to fitness of rule.

2.2. Results of looking at mutational robustness

Choose 1 genotype (i.e., rule) with a high fitness.
For i = 0 to at least 5
    for some number of samples that makes sense
        apply i mutations
            compute fitness
Figure: Graph fitness (y) by # of mutations (x axis). Two lines should be plotted: mean fitness and maximum fitness.

2.3 Results from attempting to improve

We discussed changing the IC random distribution from uniform at early generations to gaussian at later generations.
We discussed changing r so that it is larger at early generations and smaller at later generations.
Should have 1-3 figures.

3. Discussion

Should be broken down into 3.1, 3.2, 3.3.
Each should have 2 paragraphs.

4. Methods

4.1 Parameters of GA
- 4.1.1. Method for initializing initial population of rules
- 4.1.2. Selection method
  - 4.1.2.1. elitism?
- 4.1.3. describe any techniques to speed things up

4.2 How are the ICs chosen?
- 4.2.1. For part 2.1, it is using uniform distribution.
- 4.2.2. For part 2.3, it begins with uniform and slowly favors ICs with lambda approaching ½

4.3 Describe how data was analyzed for the results section.