Complex Systems and Networks HW 2

Rachael Judy, Connor Klein, Josh Smith

31 March 2024

1 Section 1: Random Policies and Social Recommendation Policy

1.1 Simulation Description

This simulation uses variations on the Schelling model exploring Red and Green agents occupying a percentage of a 2-dimensional LxL grid. The agents prefer to be near their own type and are fully happy if at k or more of the eight neighbors are of the same type. The simulation is designed with hyperparameters of a 40x40 (L=40) square grid with wraparound at the borders. The agents occupy 90% ($\alpha = .9$) of the cells and k=3 neighbors define the requirementn for total happiness of the agent. The number of trials run for each case is set to 20 trials of 20 epochs. Each epoch consists of moving every agent in the automata in a random order if the agent is unhappy. Different relocation policies and their parameters are applied to determine where unhappy agents move to seek greater satisfaction.

1.1.1 Happiness Function

An agent is defined as completely happy if k or more of its neighbors are of the same group as itself. The agent will have partial happiness represented by a linear combination of the count of matching neighbors and empty plots nearby. This function will be $H(A_i) = \frac{\text{count}(N_i) + .125\text{count}(N_e)}{8}$ where H is the happiness of an agent A of type i, N_i is a neighbor of type i, and N_e is an adjacent empty square.

1.1.2 Performance Metric

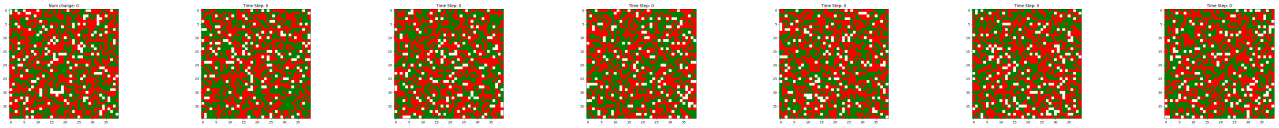
The performance metric was selected to simply be the sum of the happiness of every agent in the cellular automata ie $\sum_{a \in A} H(a)$.

1.2 Policies

The agent will consider moving only if it is unhappy. The policies modeled as described in the homework description are a random move policy and a social network recommendation policy. The random move policy has a single parameter q which limits how many random empty cells to visit looking for a cell where the agent will be happier than it is currently.

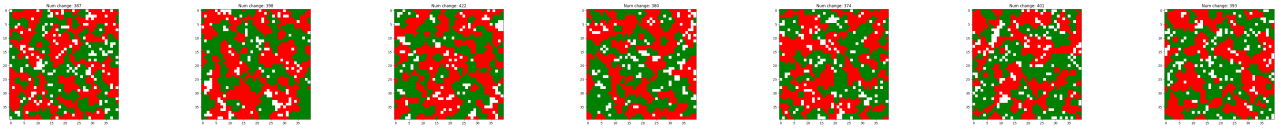
The social network recommendation (SNR) will take a randomly selected set of n friends for each agent who will look in a pxp square around themselves and report suitable squares found. The agent than randomly selects one of the suitable squares and moves there. If none is available, it defaults to the random move policy. This policy is considered over parameter values of p=[3,5] and n=[5, 10, 20].

1.3 Results



(a) Random move (b) SNR n=5, p=3 (c) SNR with n=5, p=5 (d) SNR with n=10, p=3 (e) SNR with n=10, p=5 (f) SNR with n=20, p=3 (g) SNR with n=20, p=5

Figure 1: Initial states for random move and social network recommendation policies



(a) Random move | (b) SNR with n=5, p=3 | (c) SNR with n=5, p=5 | (d) SNR with n=10, p=3 | (e) SNR with n=10, p=5 | (f) SNR with n=20, p=3 | (g) SNR with n=20, p=5

Figure 2: Final states of random move and social network recommendation policies

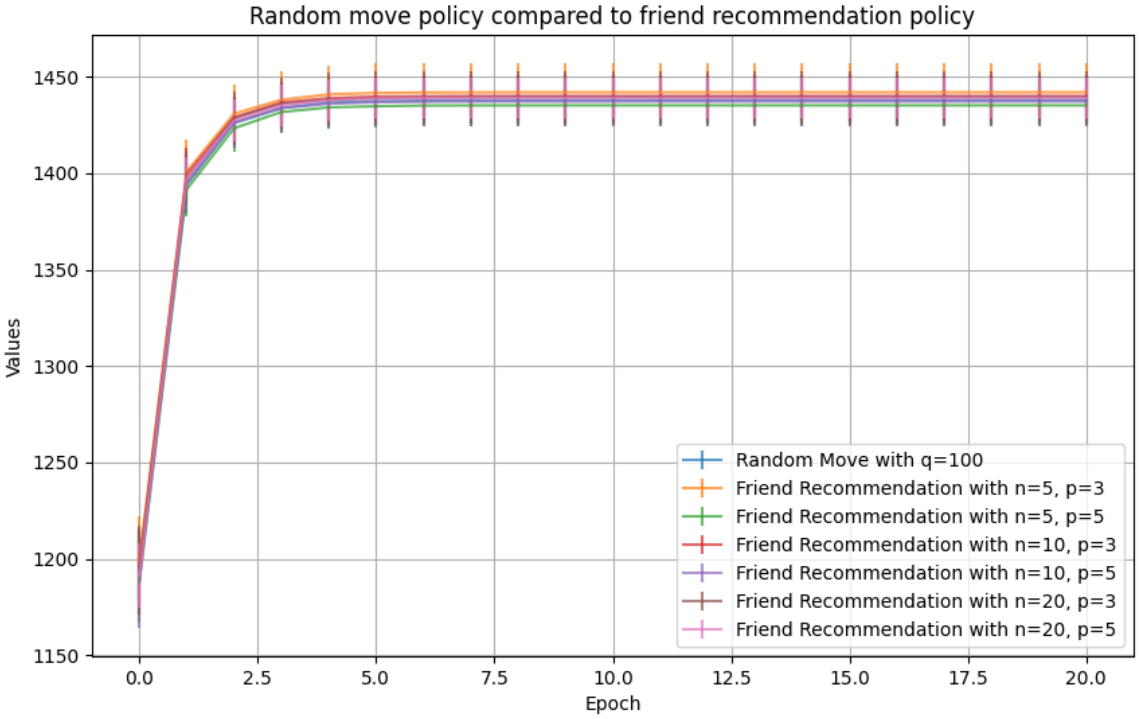


Figure 3: Time series comparing a random move policy with the social network recommendation policy

test test test

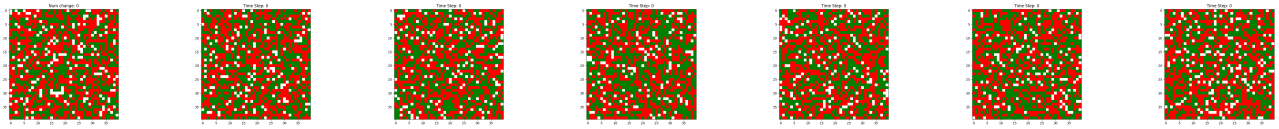
2 Section 2: Specialized Policies

2.1 Rachael

2.1.1 Policy Description

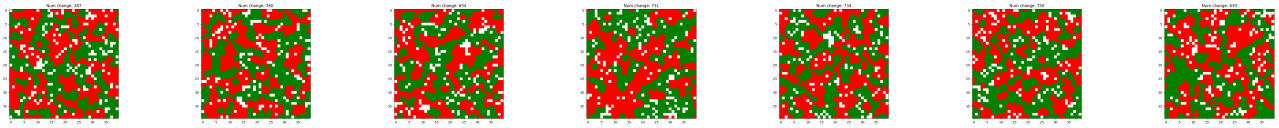
The segregation in housing application can be expanded to include searching within one's own neighborhood (SN) for a better house in the same area. The policy parameter w defines the degree of separation from a neighbor, represented as the number of plots away from one's own plot the agent will travel to ask for recommendations and parameter β defines the probability that a neighbor who is not the same type as the searcher is asked for available spots. This simulates asking neighbors of the same type in the area with a small chance of consulting neighbors of a different type. If none of the neighbors are aware of a good spot, the agent randomly moves. This policy is implemented as a BFS from the agent along matching agents up to the degree of separation limit.

2.1.2 Results



(a) Random move | (b) SN w=5, beta=.1 | (c) SN w=10, beta=.1 | (d) SN w=20, beta=.1 | (e) SN w=5, beta=.2 | (f) SN w=10, beta=.2 | (g) SN w=20, beta=.2

Figure 4: Initial states of neighborhoods for random move and neighborhood search policy



(a) Random move | (b) SN w=5, beta=.1 | (c) SN w=10, beta=.1 | (d) SN w=20, beta=.1 | (e) SN w=5, beta=.2 | (f) SN w=10, beta=.2 | (g) SN w=20, beta=.2

Figure 5: Final states for random move and neighborhood search policies

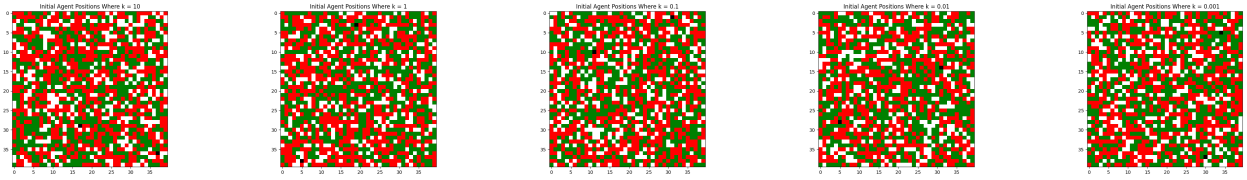
Figure 6: Time series comparison of neighborhood search compared to random move

This policy resulted in fully connected clusters across the map of like agents in Fig 5. The exceptions are small clusters of agents (see Fig 5g), which are fully happy while being small enough to not connect to the main cluster. Due to the search only of neighboring spaces, when a cluster of empty spaces appears, it often remains open because its plots are never reported by neighbors; agents also could get stuck until they talk to a mismatched neighbor. This method results in a slower convergence to optimal happiness as the entire automata is not searched but ends up better than the random search as happy neighborhoods form. The neighborhood search with greatest chance of talking to mismatched neighbors in the largest radius was the most successful as it found the most positions. Smaller radii with more matching neighbors asked performed consistently nearly as well though as perhaps makes the agent remain in its own cluster of like agents instead of hopping between clusters as could occur with a greater w. However, the standard deviations are wide enough that an instance of one parameter configuration with lesser average could perform better than another.

2.2 Connor

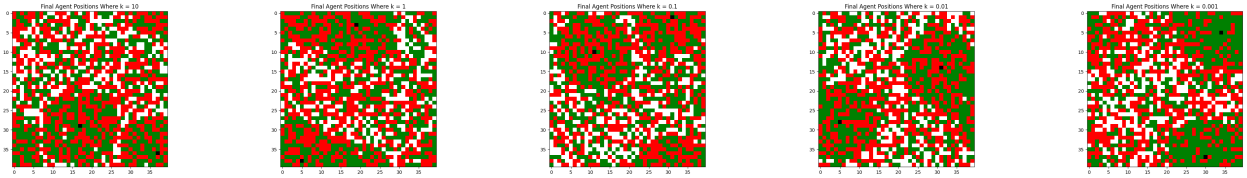
Housing can also be modeled as desireablity to a hotspot; examples being schools, religous buildings, work, etc. To model this, two different hotspots were placed randomly on the grid and red and green agents were set to want to get as close to one of the hotspots as possible. Agents choose from a generated heatmap where the probability of selecting a position is: $e^{\frac{-minimum\ \ euclidean\ \ distance\ \ from\ \ a\ \ hotspot}{radius}}$. This formula creates a probability density higher around the center of the arcs. Agents were set to be happy, and therefor not move if they were in a defined closeness value. However, the green agents were able to replace the red agents with the defined probability multipled by one over k. k was varied from 10 to 0.001 in log scale to determine how quickly the green agents could dominate the hotspots. Red agents were unable to replace green agents from their current position.

2.2.1 Results



(a) k = 10 (b) k = 1 (c) k = 0.1 (d) k = 0.01 (e) k = 0.001

Figure 7: Initial states of neighborhoods for two hotspots with radius = 10 and closeness = 10



(a) k = 10 (b) k = 1 (c) k = 0.1 (d) k = 0.01 (e) k = 0.001

Figure 8: Final states of neighborhoods for two hotspots with radius = 10 and closeness = 10

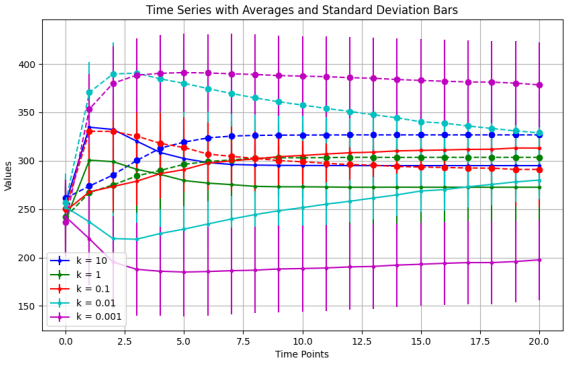


Figure 9: Time series comparing hotspot search with varying k where the full line represents the red agents and the dashed represents the red agents

Initially, Fig 8a both the red and green agents were able to coexist near the hotspots. However, as shown in Fig 8e, the green agents eventually dominate and take up most of the area's around the hotspots. This is also clear in Fig 9 where for k = 10 to 0.01, both the agent groups on average were happier after they were allowed to move. However, as k decreased to 0.001, a large transition happened were the red agents become significantly unhappier than in their starting states. This model can help demonstate the adventages of giving a group the power to evict another group and the eventual domination of a resource of that group. It is interesting that it did require a significant drive, on the scale of 1000 times their inital probability, for the green agents to entirely dominate. Different variations could be performed such as agumenting the probability equaiton so that the most desireable locations are those that are closest to both of the hotspots, varying the radius of the hotspot, or varying the agents happiness with their closeness.

2.3 Josh