ProtestGP: Coevolutionary Genetic Programming For Simulating Collective Action

James Petullo*
Brandeis University
jamespetullo@brandeis.edu

Jordan Pollack
Brandeis University
pollack@brandeis.edu

ABSTRACT

To investigate the dynamics of multi-actor collective action, this paper proposes a novel agent environment grounded in genetic programming. The context of the experiment is a protest involving four actors: police, protestors, counterprotestors, and the public. Each agent in its actor population is assigned a randomized trait array and boolean circuit for decision making while a set of payoff matrices governs the interaction outcomes. In particular, this paper presents a novel circuit architecture and mutation scheme which we have found decreases the overall volatility of mutations on the circuit and also possesses extremely high resistance to bloat. Our results show that the average agent circuit complexity did indeed increase over generations, while the agents themselves tended towards choosing their dominant strategies.

1 INTRODUCTION

Within the last two decades, there has been a renewed interest in analyzing and modeling the underlying dynamics of protests between two or more actors, usually the police, protestors, and the public. To date, much of the research has focused on building probabilistic models for predicting the likelihood of protestors mobilizing the public, toppling a regime, or alternatively being defeated and repressed by pro-government forces [2, 4], while others have used game theory to examine outcomes given a fixed set of choices [1, 9]. With some exceptions [2, 3, 5], these studies have rarely deployed agent-based modeling, and are thus limited in their ability to include individual traits and behaviors of participating agents in the framework. As such, we believe that genetic programming allows for both the incorporation of actor traits and agent decision-making capabilities. In this way, our game is an adaptation of Moran and Pollack [8] and an extension of the work of Wechsler and Bascompte [10]. From the former we have derived our game setup as a four-species coevolutionary model, while from the latter we have utilized a boolean circuit as each agent's decision-making device and an array of boolean values to represent an agent's traits. In our game, interacting agents produce decisions by passing their rival's traits to their own circuit, with the output serving as the decision.

2 THEORY OF PROTEST

The game consists of four actors: protestors, police, counterprotestors, and the public. Their coevolutionary interactions are defined in Figure 1.

*Corresponding author

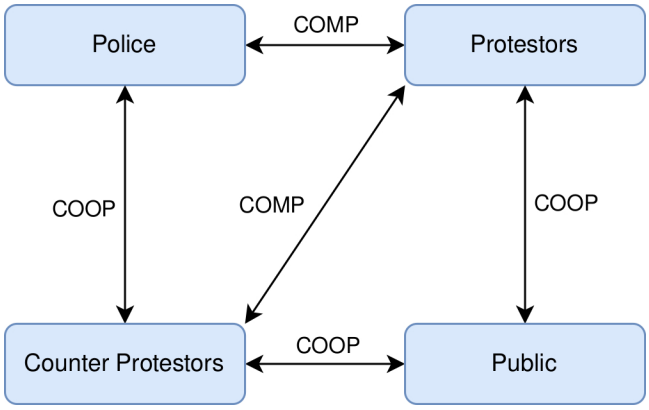

Figure 1: Coevolutionary interactions between four actors in a protest. Each bidirectional edge shows the match pairings between species, while the edge label describes the interaction type.

The protestors have an adversarial, competitive relationship with the police and counterprotestors, while both the protestors and the counterprotestors have a cooperative relationship with the public. Furthermore, the counterprotestors and the police have a cooperative relationship. The nature of these interactions are derived from the methodology described in Ginkel and Smith [4]: the primary goal of the protesters is to mobilize the members of the public, without whom there is little chance of success in topping the ruling regime. Meanwhile, police forces must counter the protesters is such a way so as to decrease the will to continue engaging in the demonstration, while avoiding the stoking of any rebellious sentiment further. At the same time, our game also includes the novel introduction of counterprotestors, who are utilized by the government as controlled opposition forces. The goal of the counterprotestors is to coordinate and cooperate with the police in subverting the messaging of the protestors that is aimed at galvanizing the masses.

3 MODEL

3.1 Agent Model

Each agent is initialized in its parent population with a random array of boolean values that denote its traits. Additionally, the agent is equipped with a randomly-generated boolean circuit that takes in the traits of an agent from a species it is assigned to interact with and produces a boolean value that represents a decision. Both the trait array and circuit are mutated with a certain probability after each interaction during gameplay.

3.2 Traits

An agent trait is an array of boolean values 1 and 0, denoted as T_1, T_2, T_3, and T_4. For every species, the trait booleans are generated with a certain probability P, as listed in Table 1. In this way, each species includes agents that share the same general characteristics while allowing for the presence of unique individual agent traits.

Table 1: Agent Trait Generation Probabilities

Species	T_1	T_2	T_3	T_4
Protestors	$P(1) = 0.8$	$P(1) = 0.4$	$P(1) = 0.4$	$P(1) = 0.4$
Police	$P(1) = 0.4$	$P(1) = 0.8$	$P(1) = 0.4$	$P(1) = 0.4$
Counterprotestors	$P(1) = 0.4$	$P(1) = 0.4$	$P(1) = 0.8$	$P(1) = 0.4$
Public	$P(1) = 0.4$	$P(1) = 0.4$	$P(1) = 0.4$	$P(1) = 0.8$

3.3 Trait Mutations

When an agent mutation is to occur, a random index $I \in \{0, 1, 2, 3\}$ is chosen, and the boolean value at that index is negated. This is in keeping with standard genetic programming mutational practice for boolean arrays.

3.4 Decision Circuit

Each agent is initialized with a random boolean circuit that acts as a simple decision-making device. Here, our circuit is utilized in a manner described in Wechsler and Bascompte [10], while borrowing its topology from standard Cartesian Genetic Programming (CGP) as detailed in Miller [6]. The circuit takes the form of a directed acyclic graph with four input nodes, four internal (hidden) layers composed of four nodes each, and an output layer of four nodes. Each output node is connected to a single node from the last internal layer. Internal nodes are randomly connected to other nodes from layers of at least l levels-back from itself. In our game, $l = 3$. Thus, in keeping with standard CGP notation, the random circuit template RT can be expressed as $RT = C(n_c, n_r, n_o, l)$, where n_c is the number of columns, n_r the number of rows, n_o the number of output nodes, and l the number of levels back. Thus, in our game, $RT = C(4, 4, 4, 3)$. Each internal node is randomly assigned a boolean logic gate from the set $\Omega = \{AND, NAND, OR, NOR\}$. Thus, an internal node must be wired to at least two nodes that satisfy the levels-back criteria. Table 2 lists the definitions of each gate in Ω, while Figure 2 displays a randomly-generated circuit.

Decision circuits are employed by setting each input node N_i to a corresponding input trait value T_i. These values are then fed forward through the circuit, where internal nodes compute a boolean via its gate from the nodes it is connected to and subsequently passes the result to its child nodes. Once all input values have been fully fed through the circuit, the value stored in the parent connection of each node in the output layer is saved in the output array. The circuit's final output is the boolean value that occurs most frequently in the output array. This process is referred to in subsequent sections as *frequency activation*.

The reasons for an output layer, as opposed to the existence of a single output node as utilized in Weschsler and Bascompte [10], are twofold. First, choosing a boolean value from the output array on the basis of its frequency allows for a more gradual accumulation of mutations to take place before the overall output of the circuit for the same given input trait changes. Second, the existence of an output layer means that more input and internal nodes contribute to the output, reducing the deleterious effects some mutations can have on the structure of the circuit. Thus, the output layer aids in decreasing the natural volatility a mutation has on both the circuit's topology and final output.

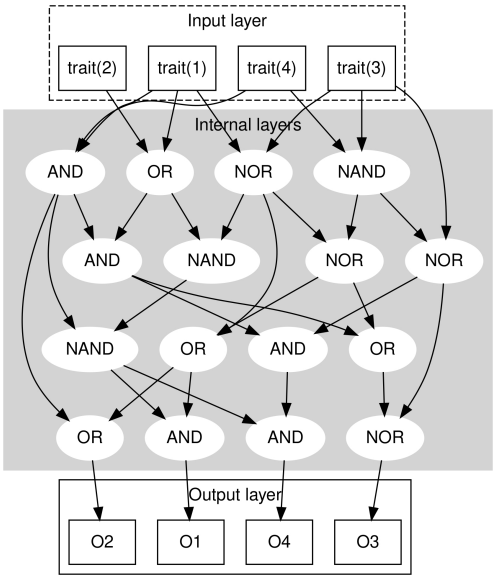

Figure 2: A random agent decision circuit of the form $C(4, 4, 4, 3)$

Our analysis shows that this architecture produces very well balanced random circuits. After generating 10,000 random circuits, each of the 16 possible input trait arrays were passed though every one, and the resulting boolean decision from the frequency activation of the output layer was recorded. We found that a final output of 1 occurred exactly 50.1% of the time. This is important validation that the agents in their randomly generated populations will not be inherently biased towards one particular decision at the beginning.

Table 2: Gate Definitions

Gate	Defintion
AND	returns 1 if all values passed to it are 1
OR	returns 1 if any of the values passed to it are 1
NAND	negates the results of *AND*
NOR	negates the result of *OR*

3.5 Complexity

The complexity of an agent's decision circuit is defined as the number of unique input nodes and internal nodes that contribute to the output. A node n contributes to the output if there exists a path from n to any output node o. Furthermore, we consider

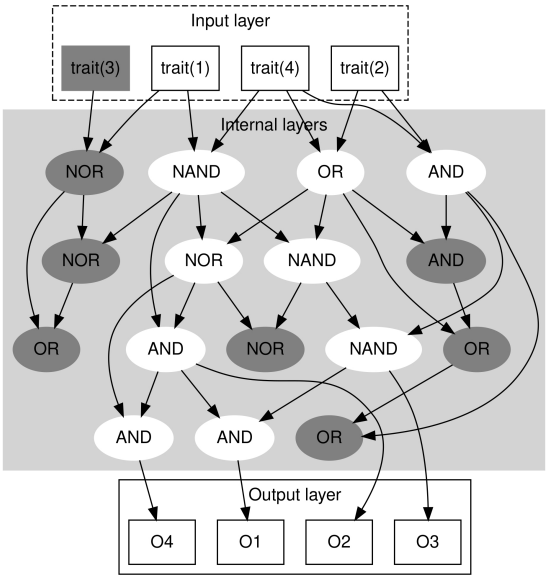

Figure 3: A random circuit $C(4, 4, 4, 3)$ with a set of active nodes (in white) and a set of inactive nodes (in gray). Each white node serves as a link in the connection from an input node to an output node, while the gray nodes do not contribute the computed values from their boolean gates to any output node.

nodes needed to complete the computation of the output as *active*, and nodes that are not needed as *inactive*. For an example of this paradigm, see Figure 3.

4 CIRCUIT MUTATIONS

There are four mutation operators that are applied with uniform probability to an agent's decision circuit: activation of a randomly-chosen inactive node, deactivation of a randomly-chosen active node, node connection rewiring, and node gate updating. Here, our first two mutations deviate from the standard CGP node addition and removal operators. The primary reason for these choices of mutation operators is that pure node addition and removal often have a net deleterious effect on the complexity of the graph. Our analysis shows that node removal decreases the complexity of the circuit disproportionately to node addition, which only provides a smaller net complexity increase in contrast (see Figure 4). This is due largely in part to the random rewiring of the connections to nodes that were formerly linked to the node selected for removal. As a result, the complexity of an agent's circuit decreases precipitously across generations, resulting in a "race to the bottom" and the overwhelming presence of low-complexity, underperforming decision circuits in the populations of each species. Meanwhile, node activation and deactivation provide a much more balanced net increase and decrease in complexity change across mutations (see Figure 5).

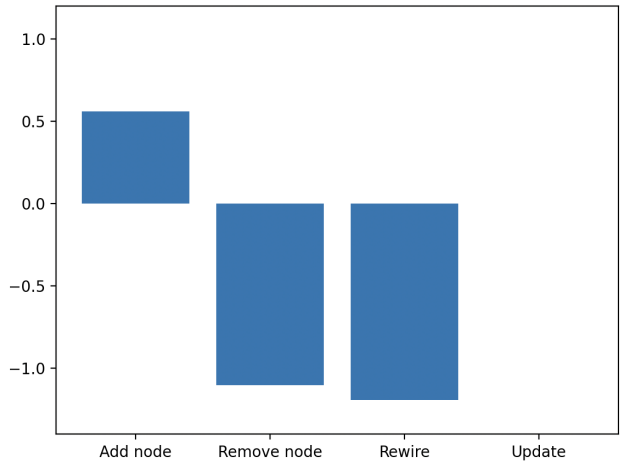

Figure 4: Net complexity changes recorded when applying standard $\{add\_node, remove\_node, rewire, update\}$ mutations to 1000 randomly-generated $C(4, 4, 1, 3)$ circuits. With only a single node in the output layer and the application of node addition and removal, complexity is only marginally added by node addition, while node removal and rewiring significantly decrease complexity, on average.

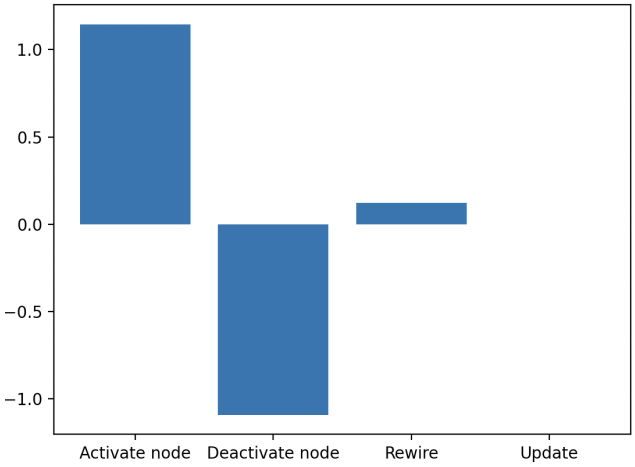

Figure 5: Net complexity changes recorded when applying new $\{activate\_node, deactivate\_node, rewire, update\}$ mutations to 1000 randomly-generated $C(4, 4, 4, 3)$ circuits, each with an output layer of 4 nodes. Now, greater pairity is achieved between the net complexity increase and loss that occurs during node activation and deacivation, respectively.

4.1 Node Activation

Node activation is performed by randomly selecting an inactive node and connecting it to a random active node within l levels after it (see Figure 6). If every node in the circuit is active, a new node with a random gate from Ω is created and wired into the circuit.

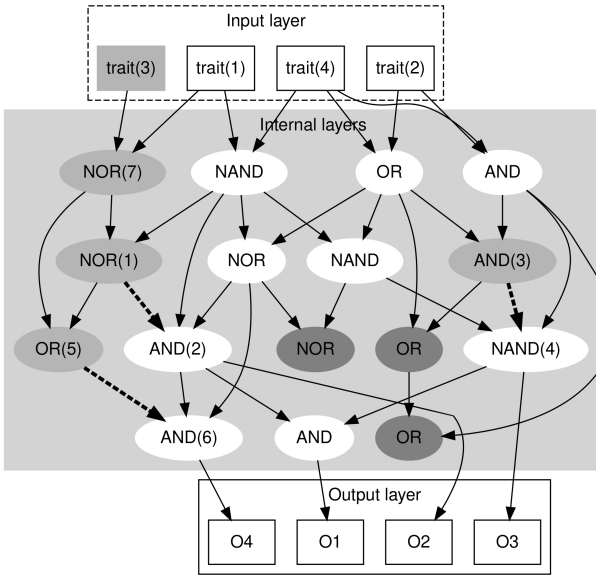

Figure 6: The nodes $\{NOR(1), NOR(7), OR(5), trait(3)\}$ are all activated if either $OR(5)$ or $NOR(1)$ is connected to $AND(6)$ or $AND(2)$, respectively, while $AND(3)$ is activated when connected to $NAND(4)$. This process is reversed during node deactivation.

A circuit with such a node activation schema is devoid of bloat, as detailed in Miller and Smith [7], due to its bounded nature with a fixed number of nodes. However, in our game, it is possible a circuit could be fully connected and thus require the creation of a new node. To investigate this, we ran 1000 mutations on 100 randomly generated circuits and found that new node creation only occurred 1.6% of the time, for an average of 16 out of 1000 mutations. As such, we believe that our schema does not suffer from undue bloat.

4.2 Node Deactivation

Node deactivation consists of randomly choosing an active node and removing all connections it has to other active nodes. The latter are rewired to other active nodes within l levels behind it.

4.3 Node Rewiring

A rewiring mutation chooses a node at random and changes one or more of its connection origins.

4.4 Node Gate Update

Here, a random node has its gate updated to a function chosen uniformly at random from Ω.

5 PAYOFF MATRICES

The payoff matrices have been designed to reflect the dynamics of cooperative and competitive relationships in protests. In the context of the game, a decision of 1 corresponds to a cooperative choice (non-violent, conciliatory, etc.), while 0 is a competitive response (violent, hostile). Figure 7 displays the payoff matrices that govern the interactions between each pair of interacting species.

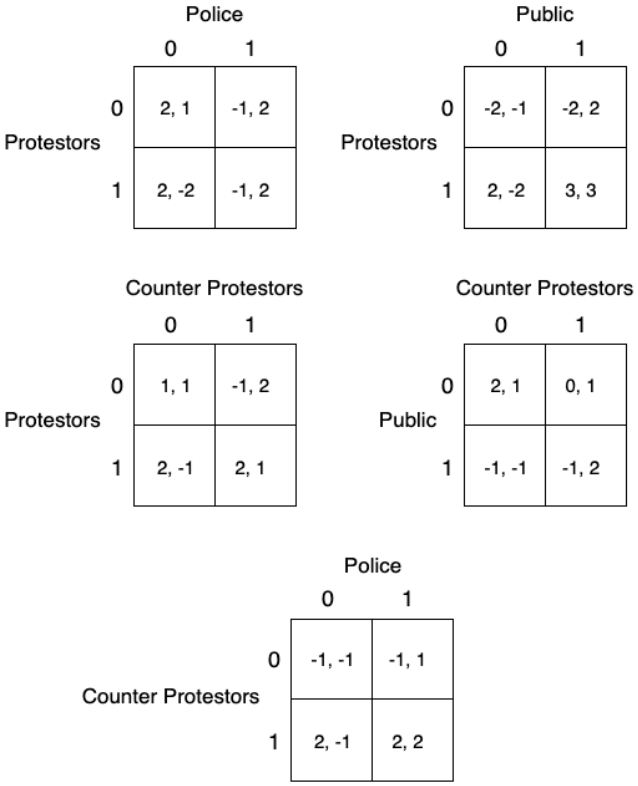

Figure 7: Game payoff matrices

5.1 Protestors vs Police

Protestors are incentivized to act aggressively, while the police's dominant strategy is non-violence.

5.2 Protestors vs Public

Both public and protestors are incentivized to cooperate, as the primary goal of the protestors is to mobilize the masses, while the public are seen to benefit the most from the advocacy of the protestors.

5.3 Protestors vs Counterprotestors

Both actors are competing with each other for the attention of the public, while mutual cooperation nullifies the threat that the counterprotestors present to the protestors.

5.4 Public vs Counterprotestors

This is a weaker cooperative relationship, as it is deemed in the best interests of the public to cooperate with the protestors as opposed to the counterprotestors, hence the small penalty to the public when responding with a 1 when matched against the counterprotestors.

5.5 Counterprotestors vs Police

As the goal of the counterprotestors and the police is to work in tandem to dilute the efficacy of the protestors, both the counterprotestors and police are incentivized to cooperate.

6 GAMEPLAY

Each generation, for every pair of interacting species, a set of all-vs-all games are played. Each agent from its parent species population is matched against every agent of the opposing species. When two agents from different species are matched, each have the trait array of the other passed to its own decision circuit. The activated output from the circuits are the output decisions, which are subsequently used to award payouts to each agent for that encounter from the payoff matrix that governs the interaction of their species. Once all matchings have been performed, a new population for each species is created. Using fitness-proportionate selection, candidate members of the new population are randomly chosen. With a very small probability ($p = 0.01$), the new agents are mutated, with both their trait array and decision circuit being altered via the processes described previously.

7 EXPERIMENTS

Each species was initialized with a population of 50 random agents. Then, we performed seven experiments over 5000 generations. At the end of each generation, the complexity scores for all the agents' decision circuits were computed. Then, the median of the complexity scores of every agent in each population were recorded, in keeping with the methodology described in Moran and Pollack [8]. After all the experiments were performed, the mean of the medians was found for every population. Furthermore, the process described was repeated again with random selection in place of fitness-proportionate selection for new population creation, establishing a random drift control.

Every generation, two additional agent metrics were computed and averaged across the seven experiments for each population. When awarded a payout, the maximum possible payout the agent could have received was also recorded. Then, once the all-vs-all games had been played, the sum of all the payouts each agent received was divided by the sum of the best payouts it could have received. The fitness score is defined as

$$fitness(species) = \frac{\sum_{agent \in species} agent_{awarded\_payout}}{\sum_{agent \in species} agent_{optimal\_payout}}$$

Thus, the highest performing populations can be observed to have an average agent fitness score that tends towards 1.

Lastly, the frequency of decisions 0 and 1 made by agents from each pair of interacting species was recorded and averaged across experiments. In the end, the ratio of 0 decisions to 1 decisions made by agents when being matched against other species was calculated. These measurements made it possible to observe trends in decision making among agents in specific matchups and compare the decisions with the expected strategies derived from an analysis of the payoff matrices.

8 RESULTS AND DISCUSSION

8.1 Complexity

To establish a quantifiable measure of complexity growth in both the control and fitness-proportionate experiments, three values were computed for each species' complexity scores across generations: final complexity, maximum complexity, and the slope of the complexity curve, found using the least-squares method. Table 3 contains the measurements found using the calculations described while Figure 8 illustrates the overall complexity trends in the control and fitness-proportionate experiments.

Overall, the control displayed greater rates of complexity growth, as evidenced by the slopes and maximum complexities than the fitness-proportionate experiments. This is a surprising result, as it stands in contrast to the findings in Moran and Pollack [8]. We hypothesize that the choice of payoff matrices in this game may be such that the strong coevolutionary pressure needed to drive complexity beyond the baseline of random drift may not be present, although future research is needed to explore this more thoroughly.

8.2 Fitness

Metrics similar to those for complexity growth were also computed for the fitness values after the first 100 generations of each species: final fitness, maximum fitness, slope (found using the least-squares method), and y-intercept. Table 4 contains the measurements found using the calculations described while Figure 9 illustrates the overall fitness trends in the control and fitness-proportionate experiments. The fitness for the control displays a very small increase across all species except the protestors. This is in keeping with a random drift towards more complex agents, which would be expected to correspond to fitter species.

Meanwhile, fitness-proportionate selection yielded species which very quickly, after the first 100 generations, established a set threshold of fitness (identifiable by its fit line's y-intercept) that largely persisted for each across the entire course of the game.

8.3 Matchup strategies

To understand the decision making of individual agents, the frequency of circuit outputs 0 and 1 made by each pair of interacting species was recorded for each generation and averaged across all the fitness-proportionate experiments. The percentage of 1 decisions to 0 decisions was then found and graphed.

8.4 Matchup 1: Protestors vs Police

Both protestors and police favored an output of 1, signifying nonviolence (see Figure 10). Per the payoff matrix, the police have a dominant strategy of 1, which they very quickly converged to. The protestors do not have a dominant strategy, yet also chose 1 most frequently.

8.5 Matchup 2: Protestors vs Public

The protestors have a dominant strategy of 1, which is quickly discovered and maintained by the agents in the species across subsequent generations (see Figure 11). The public also have a dominant strategy of 1, however, they were far less inclined to adhere to it as

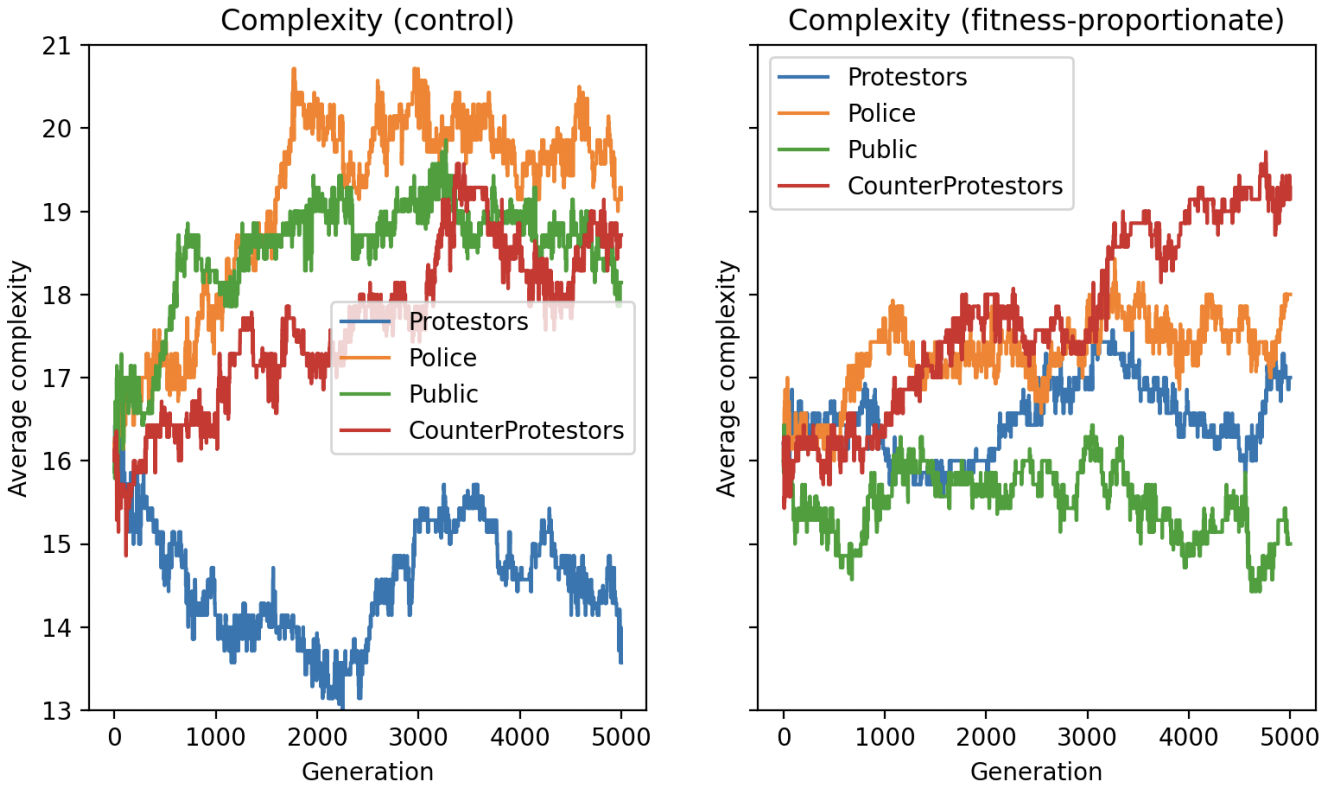

Figure 8: Species Complexities Across Generations

firmly as the protestors. This may be due to competing interests in matchups with other species.

8.6 Matchup 3: Protestors vs Counterprotestors

Both protestors and counterprotestors have a dominant strategy of 1, which is reflected in both species' overall decision to produce 1 across gameplay generations (see Figure 12).

8.7 Matchup 4: Public vs Counterprotestors

Here, the counterprotestors have a dominant strategy of 1, while the public have a dominant strategy of 0. However, neither side decisively converged on their dominant strategy, with considerably more flux in choice present (see Figure 13).

8.8 Matchup 5: Counterprotestors vs Police

The counterprotestors quickly found and maintained their dominant strategy of 1. The police also have a dominant strategy of 1, which they largely produced throughout the game, although not as consistently as the counterprotestors (see Figure 14)

REFERENCES

[1] Konstantin Ash. 2022. A game-theoretic model for protest in the context of post-communism. *Communist and Post-Communist Studies* 44, 1 (May 2022), 15. https://doi.org/doi:10.1016/j.postcomstud.2011.01.007

[2] Salvador Barbera and Matthew O. Jackson. 2019. *A Model of Protests, Revolution, and Information.* Stanford University - Department of Economics; Santa Fe Institute. Retrieved December 14, 2023 from https://papers.ssrn.com/sol3/papers.cfm?abstract_id=2732864

[3] Joshua M. Epstein. 2022. Modeling civil violence: An agent-based computational approach. *Proceedings of the National Academy of Sciences* 99, 3 (May 2022). https://doi.org/10.1073/pnas.092080199

[4] John Ginkel and Alastair Smith. 2022. So You Say You Want a Revolution: A Game Theoretic Explanation of Revolution in Repressive Regimes. *Journal of Conflict Resolution* 23, 3 (May 2022), 25. https://doi.org/10.1177/002200279043003002

[5] Tamsin E. Lee. 2018. The Thin Blue Line Between Protesters and Their Counter-Protesters. *Journal of Artificial Societies and Social Simulation* 21, Article 10 (March 2018). https://doi.org/10.18564/jasss.3676

[6] Julian F. Miller. 2019. Cartesian genetic programming: its status and future. *Genetic Programming and Evolvable Machines* 21 (August 2019), 39. https://doi.org/10.1007/s10710-019-09360-6

[7] Julian F. Miller and Stephen L. Smith. 2006. Redundancy and Computational Efficiency in Cartesian Genetic Programming. *IEEE Transactions on Evolutionary Computation* 10, 2 (April 2006), 7. https://doi.org/10.1109/TEVC.2006.871253

[8] Nick Moran and Jordan Pollack. 2019. Evolving Complexity in Prediction Games. *Artificial Life* 25 (April 2019), 17. https://doi.org/10.1162/artl_a_00281

[9] Daniel Stockemer. 2012. When do People Protest? – Using a Game Theoretic Framework to Shed Light on the Relationship Between Repression and Protest in Hybrid and Autocratic Regimes. *Social Sciences and Cultural Studies - Issues of Language, Public Opinion, Education and Welfare* 25, Article 10 (September 2012), 13 pages. https://doi.org/10.5772/38155

[10] Daniel Wechsler and Jordi Bascompte. 2022. Cheating in Mutualisms Promotes Diversity and Complexity. *The American Naturalist* 199, 3 (March 2022), 12. https://doi.org/10.1086/717865

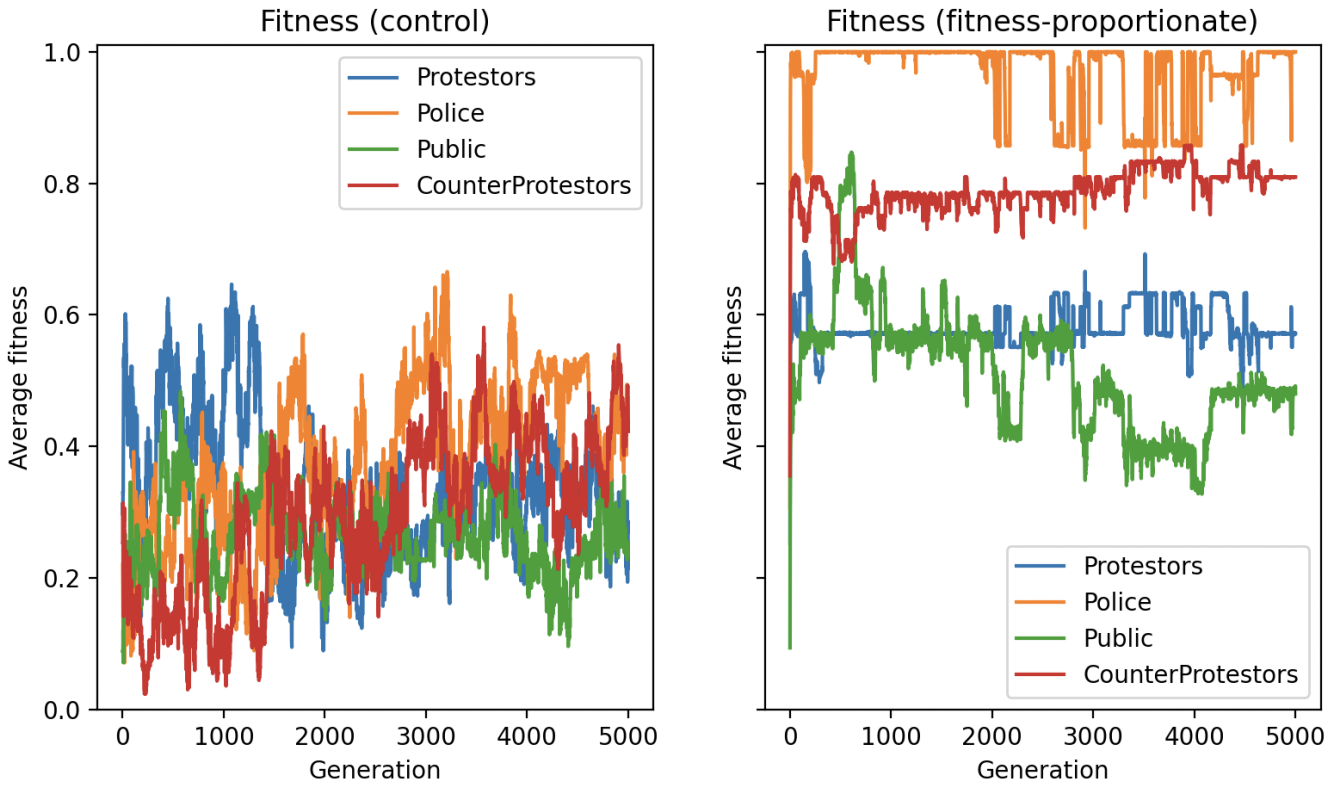

Figure 9: Species Fitnesses Across Generations

Table 3: Complexity Trends

Species	Final complexity	Max. complexity	Complexity trend (slope)
Control *Protestors*	13.57	16.43	3.58e-05
Control *Police*	19.14	20.71	0.0006
Control *Public*	18.14	19.86	0.0003
Control *Counterprotestors*	18.71	19.57	0.0006
Protestors	17.0	17.71	0.0001
Police	18.0	18.43	0.0002
Public	15.0	16.43	-7.84e-05
Counterprotestors	**19.28**	**19.71**	**0.0007**

Table 4: Fitnesss Trends

Species	Final fitness	Max. fitness	Fitness trend (slope)	Y-intercept
Control *Protestors*	0.28	0.65	-2.45e-05	0.39
Control *Police*	0.44	0.66	5.19e-05	0.25
Control *Public*	0.25	0.48	-1e-05	0.29
Control *Counterprotestors*	0.42	0.58	6.51-05	0.13
Protestors	0.57	0.69	3.35e-06	0.57
Police	**1.0**	**1.0**	-9.5e-06	**0.99**
Public	0.48	0.85	-4.55e-05	0.63
Counterprotestors	0.81	0.86	**1.62e-05**	0.75

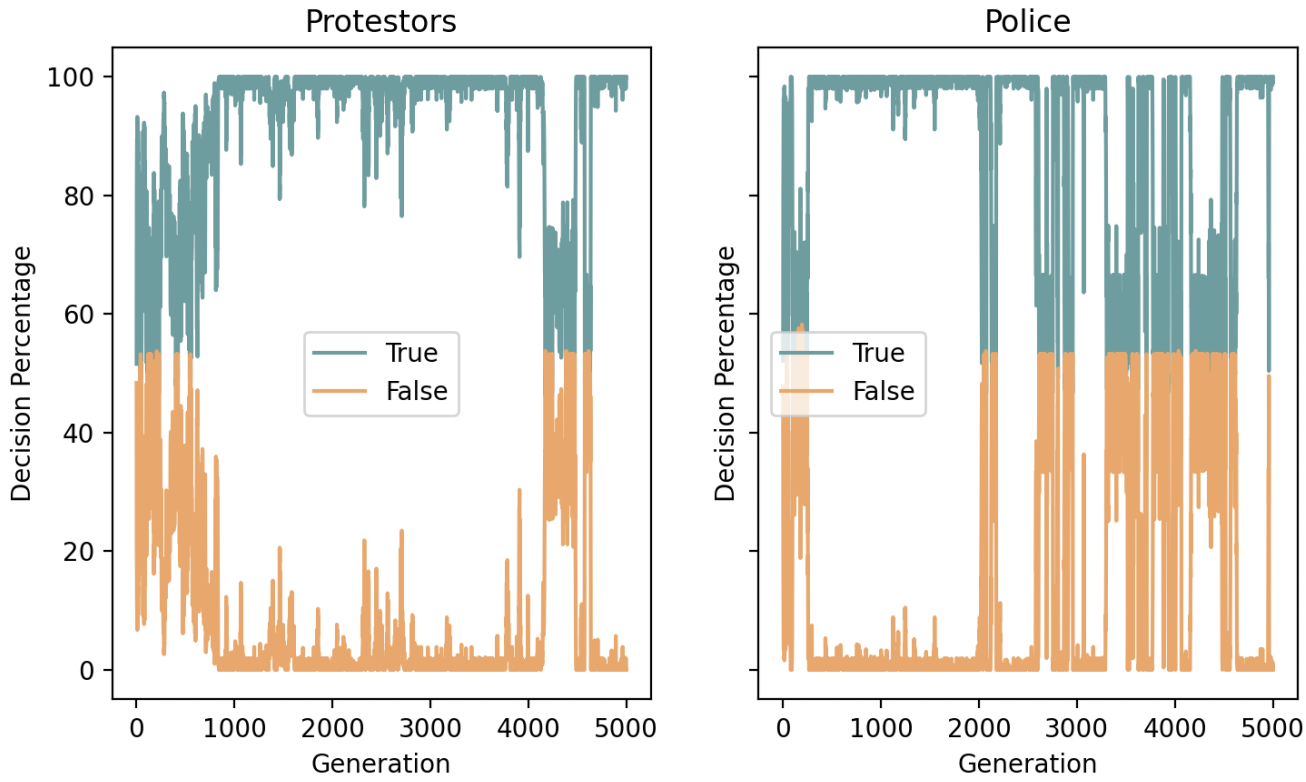

Figure 10: Average Frequency of Decisions during Protestors and Police Matchups

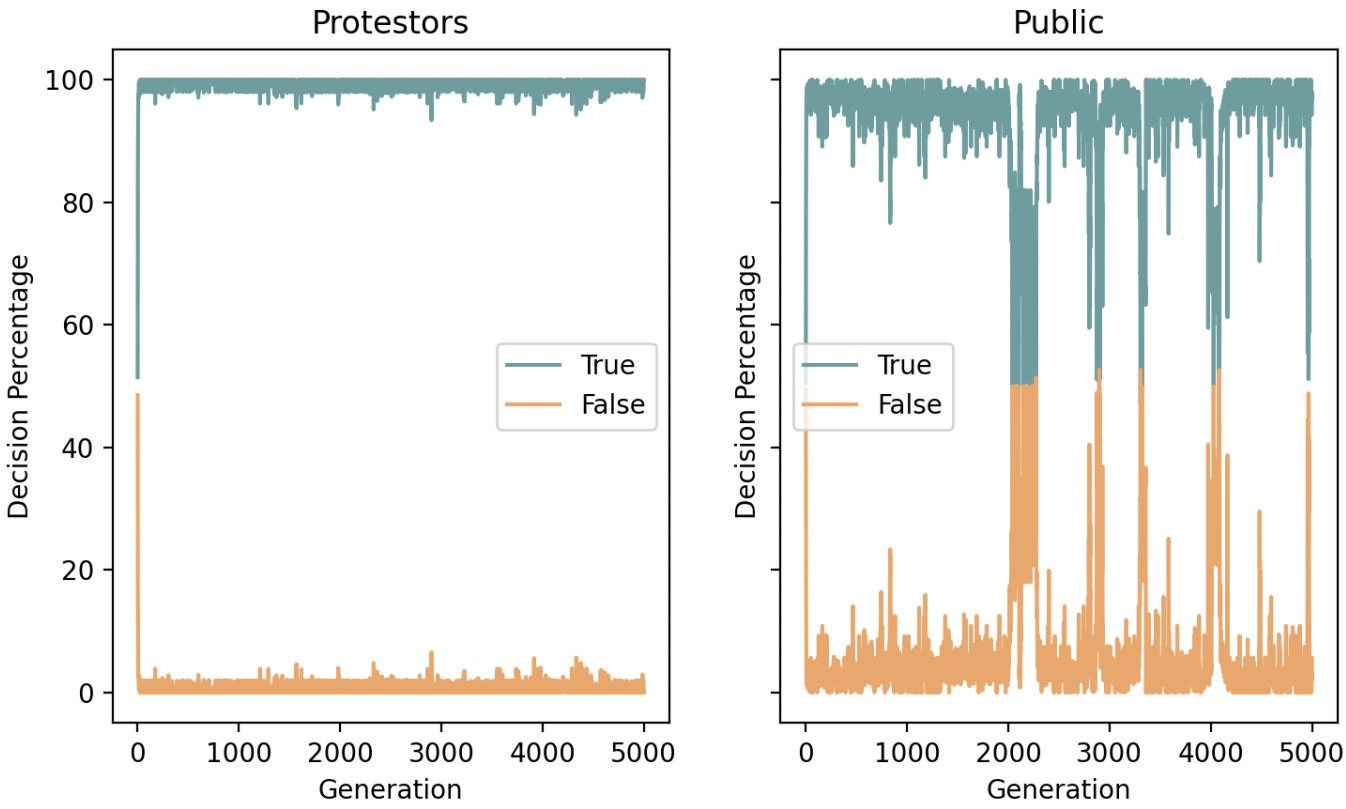

Figure 11: Average Frequency of Decisions during Protestors and Public Matchups

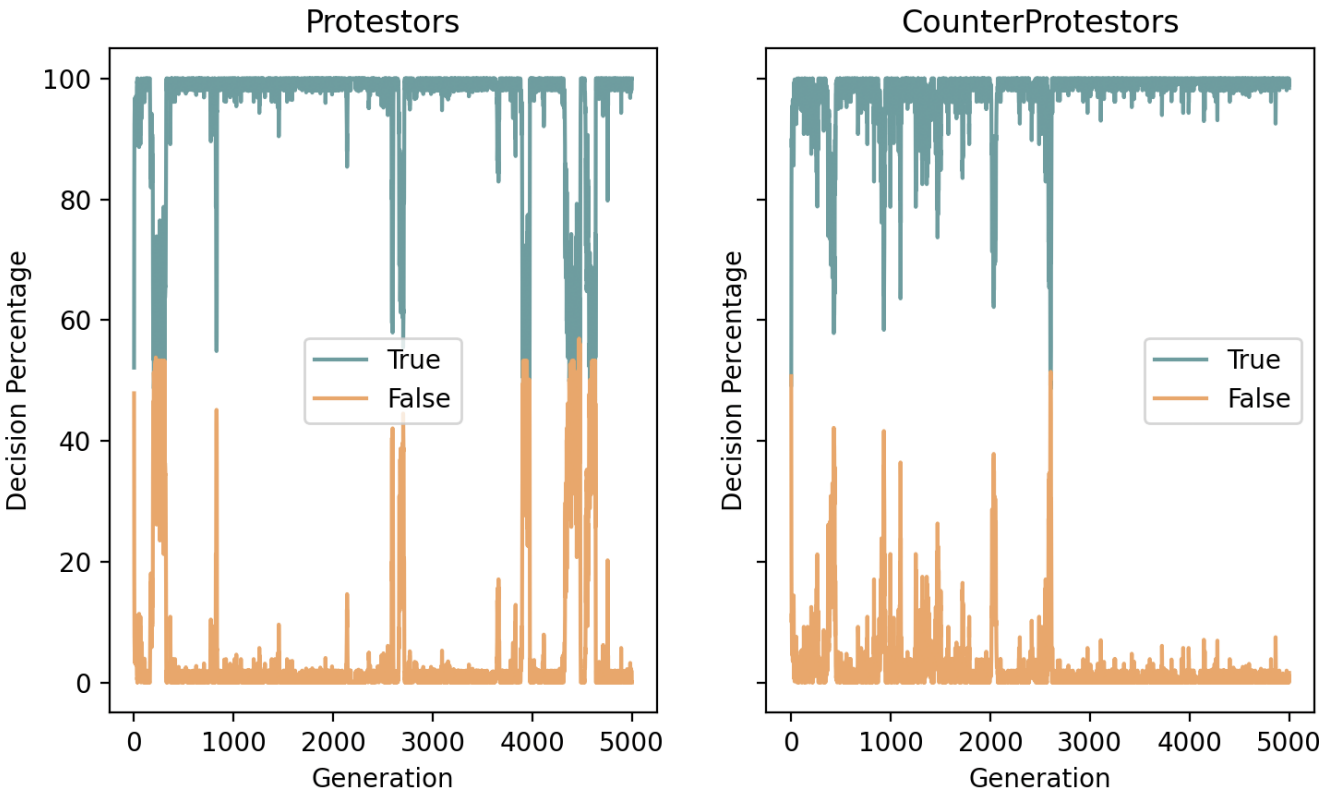

Figure 12: Average Frequency of Decisions during Protestors and Counterprotestors Matchups

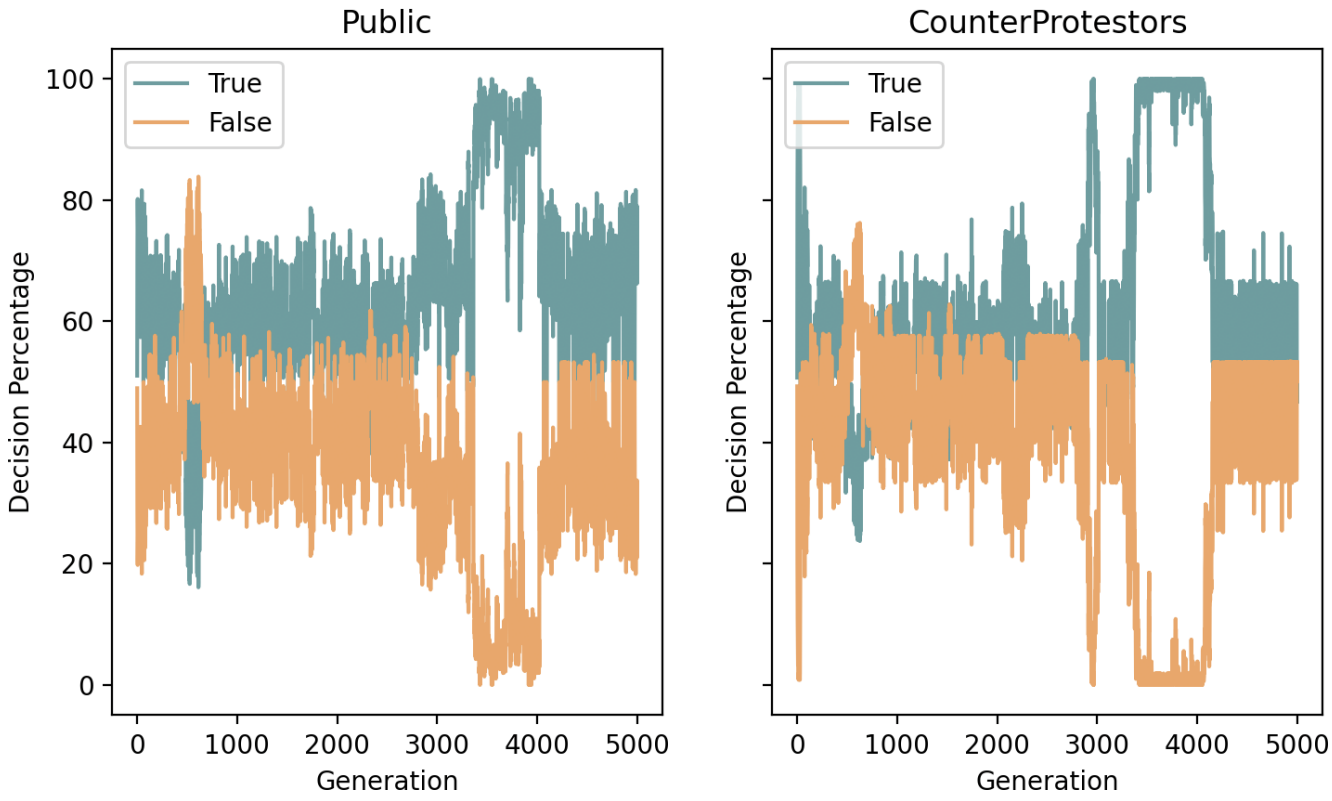

Figure 13: Average Frequency of Decisions during Public and Counterprotestors Matchups

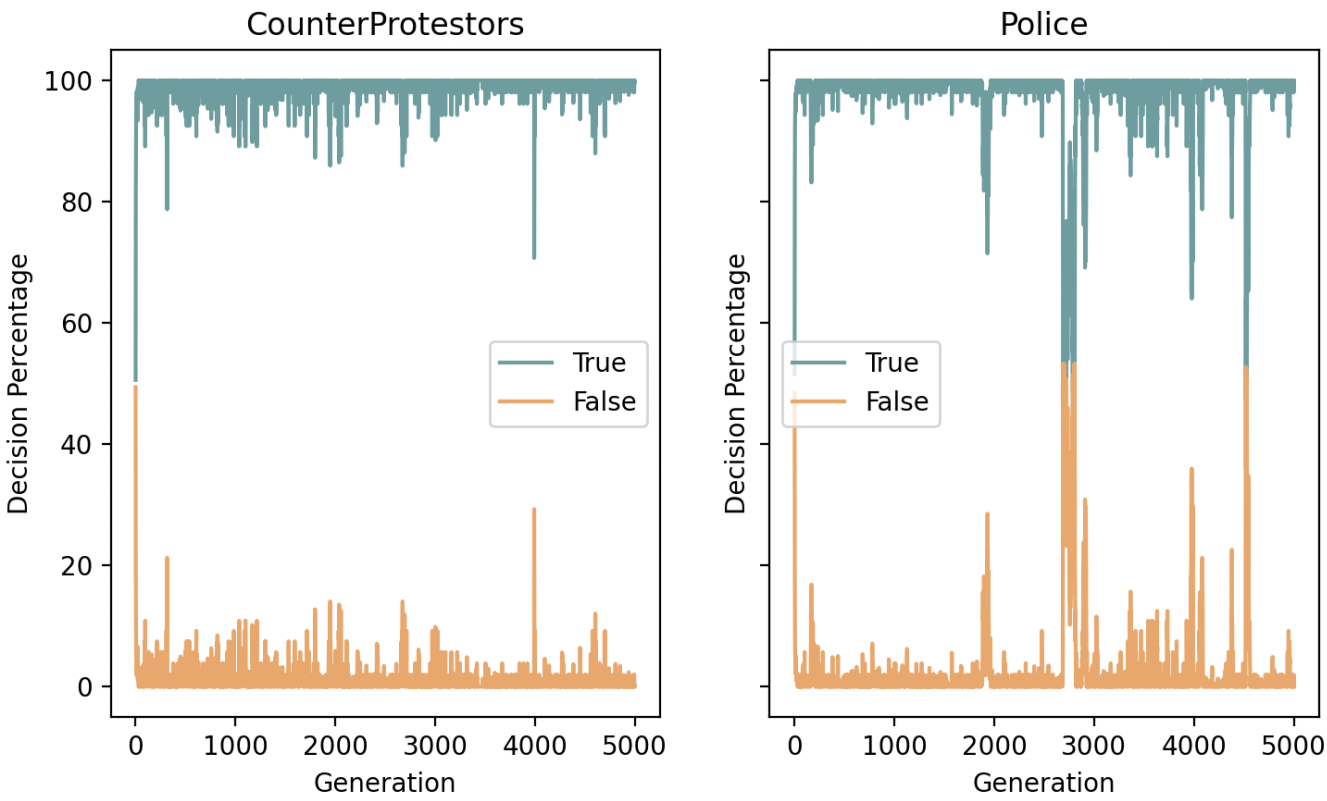

Figure 14: Average Frequency of Decisions during Counterprotestors and Police Matchups