Transactions on Computer Science and Intelligent Systems Research
ISSN: 2960-1800, eISSN: 2960-2238 | Volume 9
AIDML 2025

SmarTax: An Intelligent Tax Policy Maker via Multi-agent Reinforcement Learning and Irrationality of Humans

Tianlang Xiong *

Troy High School, Orange, LA, USA

* Corresponding Author Email: michaelxiong2020@gmail.com

Abstract. Effective tax policies are difficult to design since taxpayers' behaviors often deviate from the traditional economic models, which assume rational decision-making. Tax policymakers need tools that predict how individuals will respond to different tax policies, accounting for the irrational behaviors caused by biases, emotions, and imperfect information. This study introduces SmarTax, the first income tax policy-making simulator which integrates economics models, multi-agent reinforcement learning (MARL), and more importantly, different models of human irrationality. The goal is to maximize a society's GDP and social equity. Compared to traditional tax policy simulators, SmarTax helps reduce the gap between simulation and real-world scenarios by incorporating irrationality of the populations instead of assuming universal rationality. For example, by modeling various levels of rationality, SmarTax is able to predict how traditional RL-based tax policy-making process might become invalid due to the fact that low-income households might underutilize tax credits due to complexity or a lack of information. The evaluation of SmarTax was conducted on three reinforcement learning (RL) algorithms: Independent Proximal Policy Optimization (IPPO), Multi-Agent Deep Deterministic Policy Gradient (MADDPG) and Bi-level Mean Field Actor-Critic (BMFAC). The results indicate that, first, the inclusion of irrational factors impedes the effectiveness of all three RL algorithms. Second, compared to IPPO and MADDPG, BMFAC performs better when the irrationality level is low (<=0.1): the algorithm can successfully find a tax policy with irrationality levels up to 0.1. However, it is less robust to higher rationality levels. Its performance significantly drops when the irrationality level is beyond 0.1. Hence, SmarTax helps policymakers design transparent, understandable, and inclusive policies, aiming to achieve fairness and economic sustainability in society.

Keywords: Irrationality Model; Markov Decision Process; State; Action; Reward Function; Transitional Function; Deep Q-Learning.

1. Introduction

No modern economics operate without governmental intervention to help drive economic growth and ensure social equity [1]. The degree of intervention varies widely across different economic systems, from the minimal involvement in free-market economies to the extensive control seen in planned economies, with mixed economies falling somewhere in between. However, pinpointing the ideal level of government intervention is a complex task due to several challenges. First, it is difficult to distill actionable insights from the intricate web of societal dynamics. Second, governments struggle to accurately model a large and diverse population, where each individual has his/her unique preferences and characteristics. Lastly, predicting how individuals will react to various incentives is notoriously unreliable, adding another layer of complexity to policy-making decisions.

In this project, I focused on a crucial key component of government intervention – the tax policy. Tax policy making is a prevalent topic in economics that has been studied via traditional agent-based modeling (ABM) [2,3] and Multi-Agent Reinforcement Learning (MARL) [4,5,6]. In fact, there are many taxes policy simulator crafted for government taxation rates and as well as addressing topics like predicting the Gross domestic product (GDP) for a nation [7,8]. While tax policy simulators are an effective approach in visualizing the dynamic interactions between the government and the households, they are overshadowed by a glaring, unaddressed problem: the formulation of good governmental policies closely depends on the modeling of the society/household responses. A good

taxation policy not only needs to support 12 crucial principles such as equity and fairness, certainty, and transparency and visibility [9], but also should be able to accurately predict the human cognitive biases and uncertainty (also known as Irrationality) [10] when modeling their responses.

This work aims to fill this gap. By introducing the irrationality model [11] in households' decision-making process and integrating it into MARL, I constructed a new MARL-based platform/simulator for Tax Policy making. Within this proposed simulator, the governments can in-real time gain more realistic households' responses to different Tax policies during simulation and adapt accordingly. The integration of the irrationality model can also help check whether existing MARL algorithms can robustly maintain the performance when the households' responses significantly deviate from the assumed condition that all of them are perfectly rational decision-makers. I conducted two sets of experiments where the baseline is running different MARL algorithms with rational households, and in the test, I injected different levels of irrationality for households for the same MARL algorithms. Our results show that most of the current RL models do not converge with the integration of the irrationality algorithm.

2. Preliminaries on Reinforcement Learning

Reinforcement Learning (RL) is a computational approach to learning from interactions with an environment, aimed at achieving long-term goals using agents and environments. In RL, an agent makes decisions by selecting actions in a given state of the environment, receiving feedback in the form of rewards. Through this process, the agent aims to maximize the cumulative reward over time [12]. Unlike other learning concepts, where the correct actions are provided in the form of training data, RL relies on trial-and-error, learning from the consequences of actions [13]. Central to RL are concepts like the Markov Decision Process (MDP), value functions, and policies, which guide how the agent acts and adapts its actions based on past experiences [14]. Common algorithms, such as Q-learning and Policy gradient methods, allow agents to navigate environments, optimizing their performance across a wide range of applications [12,15]. Below I give a brief overview of the key concepts.

2.1. The Markov Decision Process

The Markov Decision Process (MDP) is a solution framework for problems that poses a degree of uncertainty, such problems are known as nondeterministic search problems [16]. Typically, it includes several basic components:

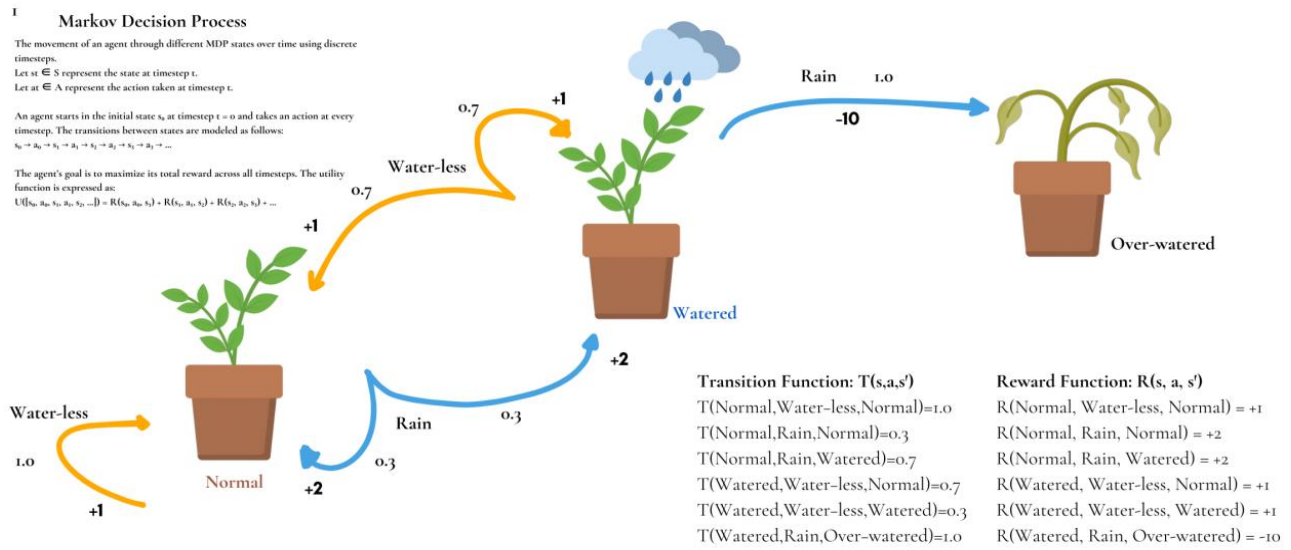

Figure 1. The MDP Process

- A set of states *s*: the collection of all possible scenarios or configurations that the agent can be in during the decision-making process.

- A set of actions a: the collection of all possible decisions that the agent can make from any given state.
- An initial/starting state.
- One or multiple terminal states where the process terminates, and the agent can no longer take further actions.
- Discount Factor γ: this factor determines the significance of future rewards compared to immediate rewards, influencing how rewards are valued over time.
- Transition Function T (s, a, s') indicates the probability of transitioning from state s to state s' after taking action a.
- Reward Function R (s, a, s') provides the immediate reward given from transitioning from states to states due to action a, and is modified to guide the agent's goal to maximize cumulative rewards.

2.2. Deep Q-Learning

Q-learning is a model-free reinforcement learning algorithm, where an agent learns the value of actions in a given state with the aim of maximizing the cumulative rewards over time. The Q-value represents the expectation of future rewards given a certain action taken in a particular state. The core update rule for Q-learning is:

$$Q(s,a) \leftarrow Q(s,a) + \alpha\left(r + \gamma \max_{a'} Q(s',a') - Q(s,a)\right)$$

Where:

- $Q(s,a)$ is the current estimate of the action-value function for state s and action a.
- α is the learning rate, controlling how much new information overrides the old.
- r is the immediate reward received after taking action a in state s.
- γ is the discount factor, determining the importance of future rewards.
- s' is the new state after taking action a.

The Q-learning algorithm works iteratively to update the Q-values based on experiences of the agent and progressively refines its policy in choosing actions with higher expected rewards. This flexibility makes it especially suitable for environments with unknown transition dynamics, as is the case in my proposed tax policy simulations.

2.3. Independent Proximal Policy Optimization

The Independent Proximal Policy Optimization (IPPO) [17] is an extension of its parent optimization algorithm, Proximal Policy Optimization (PPO) [18], modified to handle multi-agent reinforcement learning scenarios by treating agents as independent. Each agent in the algorithm learns its policy using the PPO approach but operates exclusively without observing other agents' actions.

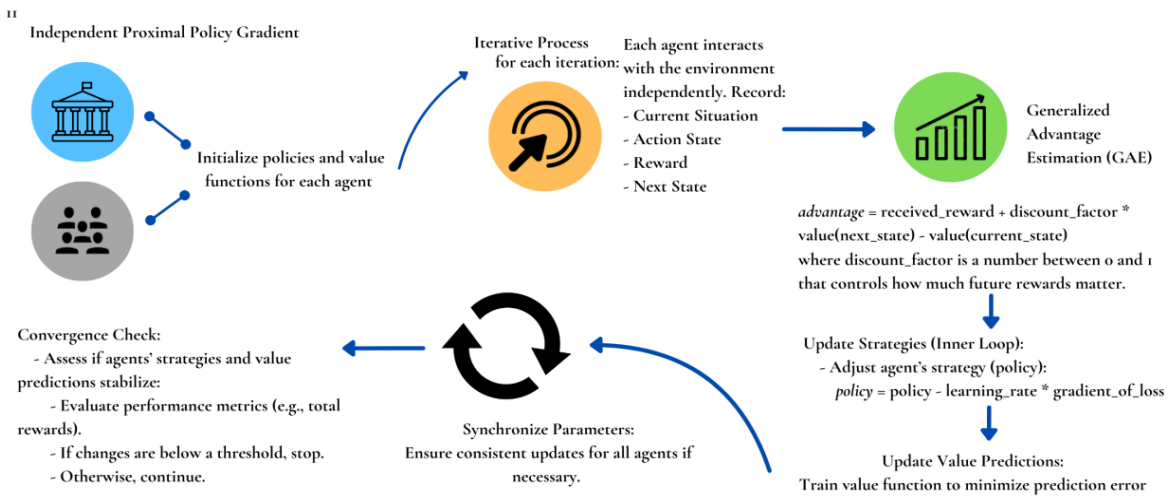

Figure 2. The flowchart for the IPPO algorithm

The algorithm alternates between updating the policies of all agents independently while maintaining stability in their learning process. Specifically, the IPPO algorithm uses clipped objective functions to ensure that updates to each agent's policy are not too large, promoting stable learning, as shown in Figure 2. The key steps include:

1. Agent Policy Update: Each agent updates its policy to maximize expected rewards, but limits how much the policy can change to maintain stability:

$$\theta_i' = \arg\max_{\theta_i} \mathbb{E}_t[\min(r_i(\theta_i)A_i, \text{clip}(r_i(\theta_i), 1-\epsilon, 1+\epsilon)A_i)]$$

Where $r_i(\theta_i)$ is the ratio of the new policy's action probability over the old one, A_i is the advantage, and the clip function limits large changes in the policy.

2. Critic Update: The critic updates its estimates of action values based on the TD error:

$$\theta_i = \theta_i + \alpha_i \delta_i \nabla_{\theta_i} Q_i(s, a_i; \theta_i)$$

Where δ_i is the temporal difference error and Q_i is the value function, estimating how good a state-action pair is.

3. Policy Update for Each Agent: The agent adjusts its policy based on the value estimates from the critic:

$$\phi_i = \phi_i + \beta \nabla_{\phi_i} \left(\log \pi_{\phi_i}(s, a_i) Q_i(s, a_i; \theta_i) \right)$$

Where $\pi_{\phi_i}(s, a_i)$ is the probability of an action under the current policy and Q_i is the critic's estimate of the action value.

2.4. Bi-level Mean Field Actor-Critic

The Bi-Level Mean Field Actor Critic (BMFAC) algorithm [19], improves on the efficiency of bi-level optimization by introducing actor-critic for both the leader (government) and the followers (household), alongside a mean-field approximation for the follower's collective behavior.

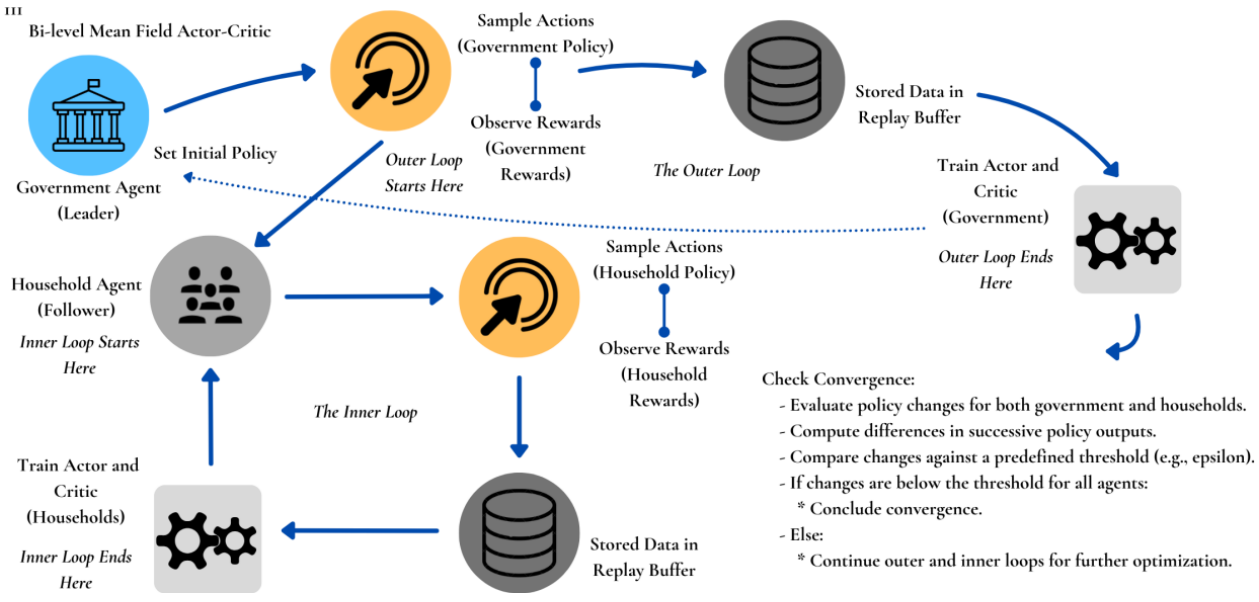

Figure 3. BMFAC flowchart

The algorithm alternates between optimizing the government's policy and optimizing the household's policies using nested loops: The outer loop allows for the government agent to update its policy by anticipating the reactions of the households, adjusting its policies such as tax rates and government spending to maximize its utility. The inner loop allows each household to optimize its policy by considering the government's policies, as demonstrated in Figure 3.

Given a transaction $\langle s, a_g, a_h, s', r_g, r_h \rangle$, the updates are as follows:

Step I: The government action update is given by $a_g{}' = \text{argmax}_{a_g} Q_g(s', a_g, \pi_h(s', a_g; \phi_h); \theta_g)$, where $a_g{}'$ is the updated government action, Q_g represents the government value function, and π_h is the household policy. The household action update is $a_i{}' = \pi_h(s', a_g{}'; \phi_h)$ for all $i \in \{1,2, \dots, N\}$, where $a_i{}'$ is the updated household action, and N is the total number of households.

Step II: The critic updates are as follows: for the government, the critic parameters are updated by $\theta_g = \theta_g + \alpha_g \delta_g \nabla_{\theta_g} Q_g(s, a_g, a_h; \theta_g)$. For the household, the critic parameters are updated by $\theta_h = \theta_h + \alpha_h \delta_h \nabla_{\theta_h} Q_h(s, a_g, a_i; \theta_h)$. The policy update for the household actor is $\phi_h = \phi_h + \beta \nabla_{\phi_h} \log \pi_h(s, a_g; \phi_h) Q_h(s, a_g, a_h; \theta_h)$, where ϕ_h represents the household actor parameters, and β is the learning rate for the household actor.

3. Problem Formulation with the Irrationality model

The concept of irrationality in economic choice is a vital one in successfully modeling household behavior. Traditional approaches often assume that households act rationally, making the most favorable decisions at every point of decision [7]. However, this perspective lacks the depth to represent the complexity that underlies real-world behavior.

3.1. Basic Irrationality Model

In this project, I propose a hybrid model in which households will show behavior with mixed rationality levels. More specifically, I merge two types of decision-making rules:

- RL Algorithm: Under this approach, households are assumed to always act rationally, selecting the optimal actions based on their learned policies. The BMFAC and IPPO algorithms falls under this category.
- Random Selection: Alternatively, households may select actions randomly, which allows for exploration of various strategies and simulates a lack of rationality.

To approximate irrational behavior, I introduce a probabilistic mechanism characterized by an ϵ value. The household will have a probability ϵ of choosing a random action. Conversely, with a probability of $1 - \epsilon$, the household will select actions based on the rational decisions provided by the RL framework.

Probability distribution for action selection:

$$P(a) = \begin{cases} \frac{\epsilon}{|A|}, & a = a_{\text{random}} \\ 1 - \epsilon, & a = a_{\text{RL}} \end{cases}$$

Programmatically, this translates to the following process for action selection:

```
function selection_action(epsilon):
    random_number = draw_random_number(0, 1)
    if random_number < epsilon:
        return select_random_action()
    else:
        return select_RL_action()
```

It is crucial to experiment with different values of ϵ to observe its effect on the results. The expected outcomes are as follows:

- As ϵ approaches 1, the household behavior should resemble randomness, yielding results similar to a purely random policy.
- Conversely, as ϵ approaches 0, the household behavior should align more closely with the rational decisions made by the RL algorithm.

To validate my model, I will conduct experiments using different algorithms, including IPPO, to reproduce results from existing literature. Additionally, I aim to gather data for multiple scenarios, including MADDPG, BMFAC, and random policies.

3.2. Advanced Irrationality Model

To produce even more precise results, the basic model has to be improved upon in order to produce accurate data that can be later used for comparison. In order to do this, I modified the action selection module of the household agent in order to acquire results. The modified selection function is different in its strategy to in exploration, action selection, and stochastically.

select_actions generally returns the mean action processed through tanh and softmax, while select_actions_house samples 100 actions and selects one based on probability weighting.

Below are the equations for producing the optimal action:

Probability of action (a)

$$P(a) = \frac{\exp\left(-\epsilon Q(s,a)\right)}{\sum_{a'} \exp\left(-\epsilon Q(s,a')\right)}$$

Action-value function $Q(s,a)$

$$Q(s,a) = \sum_{s'} [r(s,a) + \gamma V(s')]$$

Optimal action

$$a^* = \text{argmax} P(a)$$

4. The Proposed Simulation Framework

4.1. The Environment for Tax Policy Making

The Tax policy simualation framework is mainly based on the TaxAI simulator [4]. This simulator enables macroeconomic systems simulation, with a specific aim on the interaction between heterogeneous agents (households, firms, and government) and fiscal policy optimization. TaxAI framework has already integrates reinforcement learning, agent-based modeling (ABM), and general equilibrium theory to simulate the behavior and interactions among agents, all of which assumes rational decision makers. The key formulation of the agents (households, Firms, etc) and their interactions are summarized below.

4.1.1. Modeling of the Households.

The simulator considers a surplus of distinct characteristics that heterogeneous agents have, variables such as income, wealth, preferences, and labor supply are all taken in account. Agents are modeled to face idiosyncratic risks like income shocks, and can access imperfect financial markets to simulate the real world economy.

Each household is modeled to maximize utility, which is a function of consumption and labor supply. Household have budgets and borrowing constraints and have limited ability to insure against income fluctuations. The agents adapt their savings and labor decisions in response to the government agent (e.g. taxation).

$$\log c_{a,t} = -\varphi_t + 1 + \frac{\gamma}{\gamma + \theta}\alpha_t + M_{a,t}$$

$$\log h_{a,t} = -\varphi_t + 1 - \frac{\theta}{\gamma + \theta}\alpha_t + \frac{1}{\gamma}(\kappa_t + \theta_t) - \frac{\theta}{\gamma}M_{a,t}$$

4.1.2. Modeling of the Firms.

The Firm represented in the TaxAI framework is representative of all firms and industries; it converts capital and labor into goods and services. Essentially, these firms interact with households by

providing employment and influencing wages. Using market clearing on labor and goods simplifies the environment, which establishes an equilibrium between supply and demand.

$$Y_t = K_t^{\alpha} L_t^{1-\alpha}$$

Where K_t and L_t are capital and labor used for production, α is capital elasticity, and I normalize the output price to 1. The firm rents capital at a rental rate R_t and hires labor at a wage rate W_t. The produced output is used for all households' gross consumption C_t, government spending G_t, and physical capital investment $X_t = K_{t+1} - (1-\delta)K_t$, with the depreciation rate δ, so the aggregate resource constraint is

$$Y_t = C_t + X_t + G_t$$

Suppose the firm takes the marginal income from labor as households' wage rate W_t and the marginal income from capital as the rental rate R_t:

$$W_t = \frac{\partial Y_t}{\partial L_t} = (1-\alpha)\left(\frac{K_t}{L_t}\right)^{\alpha}, \quad R_t = \frac{\partial Y_t}{\partial K_t} = \alpha\left(\frac{K_t}{L_t}\right)^{\alpha-1}$$

Market clearing on labor and goods is an important assumption for simplification, which means there is an equilibrium between supply and demand. The goods market clears by Walras' Law, and the labor market clearing condition is

$$L_t = \sum_{i}^{N} e_t^i \, h_t^i.$$

4.1.3. Modeling of the Government.

The government can be defined to have either a single or multiple goals, such as promoting economic growth, maintaining social fairness and stability, and maximizing social welfare. It evaluate the effects of these policies on overall welfare and economic outcomes. For the purposes of this study, I set the government to have a single goal which is to maximize the Gross Domestic Product for the agents.

Therefore, the main interaction happens between the households and the government agents [4]:

1) The government agent is responsible for setting tax rates with the aim of optimizing long-term GDP growth. The agent is not affected by the irrationality model.

2) The household agents represent individuals or families within the economy. They respond to government policies by making decisions about consumption, savings, and investments in the environment. This agent is modified with the irrationality model to account for human errors.

The environment is defined using the Markov Decision Process (MDP) framework. The environment state contains major economic indicators such as GDP, the distribution of household income, the unemployment rate, and tax revenues. For the government, agent actions involve changes in tax rates, including income tax and corporate tax. For households, agent actions involve decisions related to spending and savings. Government rewards are designed to balance economic growth with equity. Household rewards are tied to utility maximization, considering factors such as income, savings, and consumption.

4.2. Integrating Irrationality Models into the Framework

People, or households in the original framework [4] are not considered to be irrational in terms of optimizing their goals. To integrate irrationality, the original environment is extended to allow households to occasionally act irrationally, deviating from the default pure rational decision-making process, as shown in Figure 4. This allows the simulator to emulate and reflect cognitive biases and decision-making imperfections, which can significantly alter responses to policies.

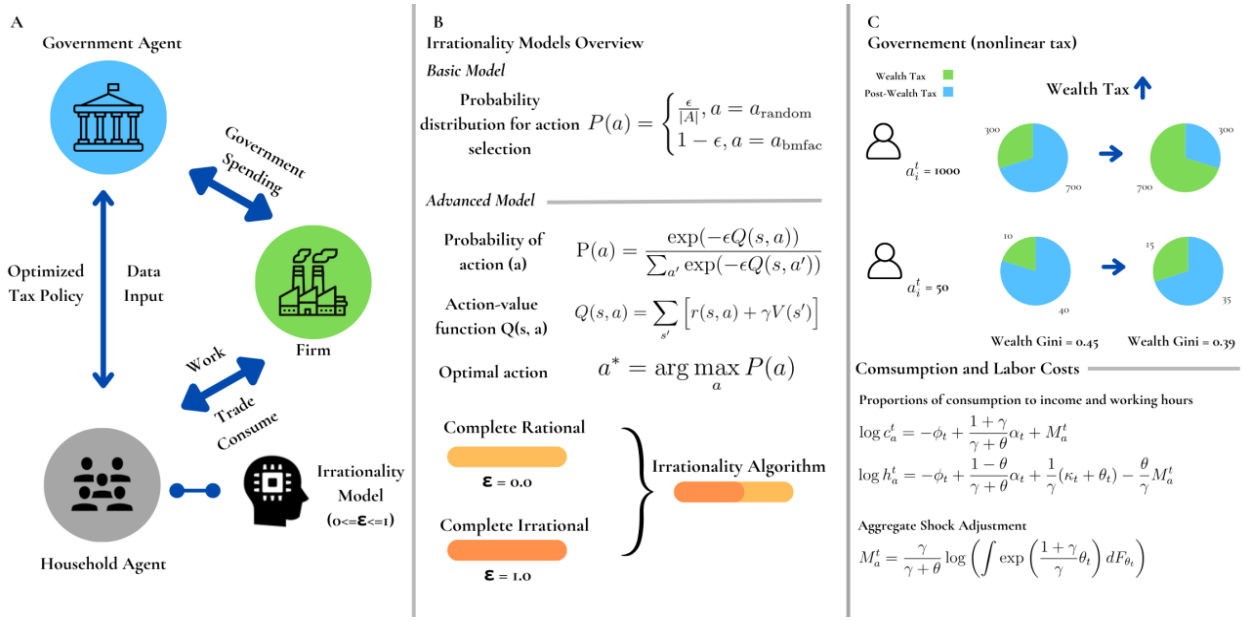

Figure 4. Model Dynamics in the Irrationality Model. A: Economic activities among the government, the firm, the financial intermediary, and households. B: The core equations for the decision-making process. In the basic model, actions are chosen probabilistically, with ϵ controlling the balance between random and rational choices. The advanced model incorporates an action-value function Q(s, a) to evaluate immediate rewards and future benefits. C: The government taxation model show how progressive taxes reduce wealth inequality. Higher taxes on wealthier households redistribute resources. This policy lower the Gini coefficient from 0.45 to 0.39, which reflects greater equity. Households adjust consumption and labor in response to post-tax income, balancing immediate needs and long-term goals.

To support the integration, two major changes are introduced to the simulator framework.

First, I modify the action selection algorithm for the households. In the original framework, household agents consistently select actions (spending/savings) that maximize their utility. To introduce the element of randomness, I allow households to sometimes choose actions randomly. This randomness is determined by a parameter ϵ , which controls the probability of irrational behavior. Specifically, a household selects a random action with probability ϵ, and follows the optimal action (as defined by the specific economic model) with probability 1-ϵ.

Using the above approach allows the model to simulate a continuum of rational and irrational behaviors, depending on the value of ϵ. When ϵ is large, households display more randomness, and when ϵ is small, households behave more rationally, aligning with the chosen economic model. This probabilistic approach to decision-making acknowledges that human decisions are often influenced by cognitive biases, bounded rationality and imperfect information. [11]

Second, I also modify the *log_prob* function by introducing a small constant $(1e-6)$ in the calculation of *pre_tanh_value* to improve stability when computing the inverse tanh. This mitigates issues when the result is very close to ± 1. The original framework [4] omits this term, which might lead to numerical errors.

5. Experiment and Results

This section presents the experimental setup, evaluation metrics, and experimental results. The goal is to evaluate the robustness of several reinforcement learning algorithms for getting effective tax policies in the presence of human irrationality.

I selected three different RL algorithms: 1) Independent Proximal Policy Optimization (IPPO), 2) Multi-Agent Deep Deterministic Policy Gradient (MADDPG) and 3) Bi-level Mean Field Actor-

Critic (BMFAC). I define our baseline to be the default RL algorithms where the household agents are strictly rational agents, i.e., $\epsilon = 0$.

5.1. Experiment setups with irrationality model integrated

I include four different levels of irrationality levels:

1. Setting 1: Household agents are with irrationality level $\epsilon = 0.1$.
2. Setting 2: Household agents are with irrationality level $\epsilon = 0.5$.
3. Setting 3: Household agents are with irrationality level $\epsilon = 0.9$.
4. Setting 4: Household agents are with irrationality level ($\epsilon = 1.0$), i.e., the decision-making process is completely random.

5.2. Experimental Results

Table 1 summarizes the average cumulative rewards (GDPs) with the three different RL algorithms with different levels of irrationality. I can see that, when the households are acted as perfectly rational agents ($\epsilon = 0.0$), MADDPG and IPPO performed similarly, but not as good as BMFAC. However, IPPO and MADDPG performed more robust to higher levels of irrationality of the households, particularly the MADDPG algorithm: the performance didn't degrade too much with $\epsilon <= 0.9$. BMFAC, on the other hand, is more sensitive to the irrationality levels: it performed better when $\epsilon <= 0.1$ but significantly degraded when the irrationality levels are higher.

Table 1. The per capita GDP achieved by different RL Algorithms for ϵ irrationality households under maximizing GDP task using the advanced Irrationality Model.

Algorithm	$\epsilon = 0.0$	$\epsilon = 0.1$	$\epsilon = 0.5$	$\epsilon = 0.9$	$\epsilon = 1.0$
IPPO	6.4e6	6.0e6	6.3e6	6.5e6	1.4e6
MADDPG	9.5e6	6.0e6	6.3e6	6.2e6	6.3e6
BMFAC	29.6e6	29.3e6	3.7e6	2.3e6	-0.03e6

5.2.1. Results with the basic irrationality model.

Figure 5 showed the cumulative rewards (GDP) of the IPPO algorithm. When the household agents acted perfectly rational, the algorithm converged to the highest rewards. When the household agents demonstrated some levels of irrationality, the performance went down ($0.1 <= \epsilon <= 0.9$). When the households are completely random, the algorithm failed to converge.

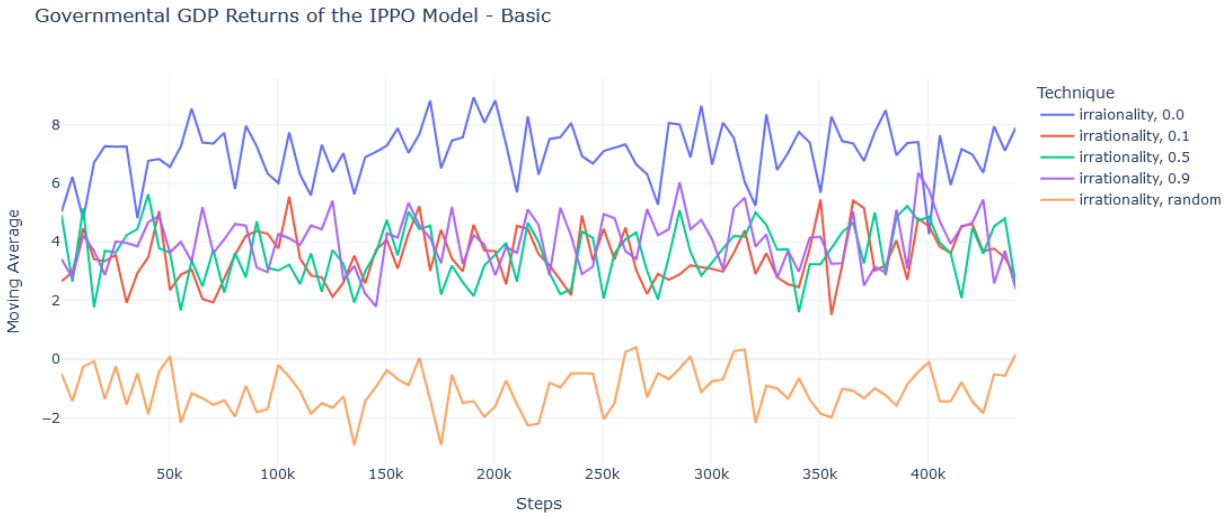

Figure 5. Basic IPPO Algorithm Results of Governmental GDP Return Under Different Household Irrationality Levels

Figure 6 showed the cumulative rewards (GDP) of the MADDPG algorithm. Similar to the IPPO algorithm, when the household agents acted perfectly rational, the algorithm converged to the highest rewards. When the household agents demonstrated some levels of irrationality, the performance went down slightly ($0.1 <= \epsilon <= 1.0$).

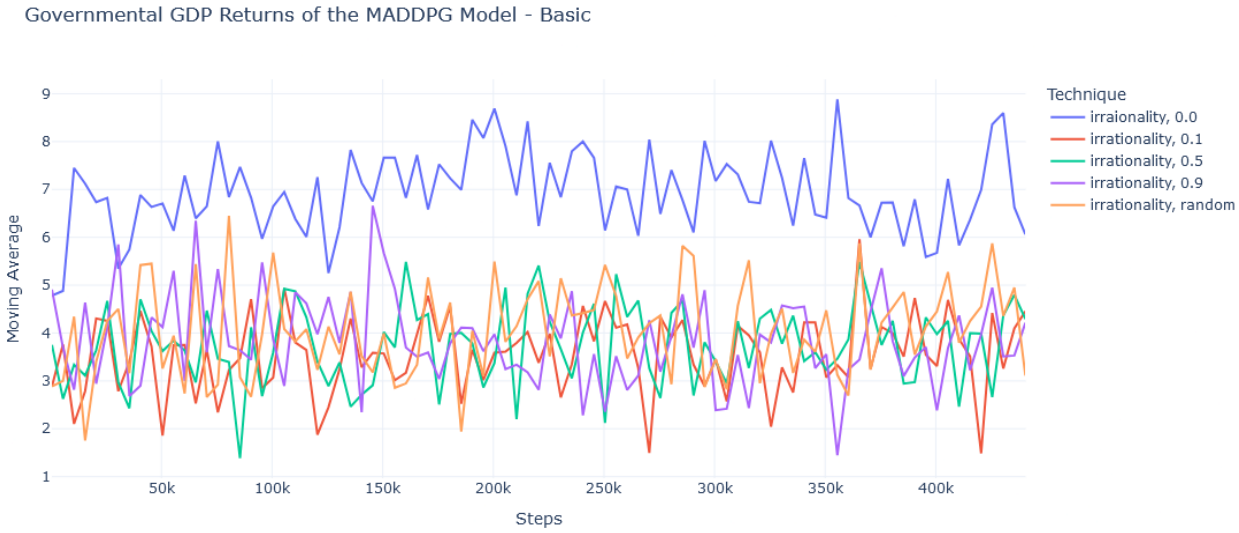

Figure 6. Basic MADDPG Algorithm Results of Governmental GDP Return Under Different Household Irrationality Levels

Figure 7 showed the cumulative rewards (GDP) of the BMFAC algorithm. Compared to the IPPO and MADDPG algorithms, when the household agents acted perfectly rational, the algorithm converged to a much higher reward. The algorithm is also quite robust when the household agents demonstrated a low level of irrationality ($\epsilon <= 0.1$). However, when the household agents demonstrated higher levels of irrationality, the performance went down more significantly ($0.1 < \epsilon <= 1.0$).

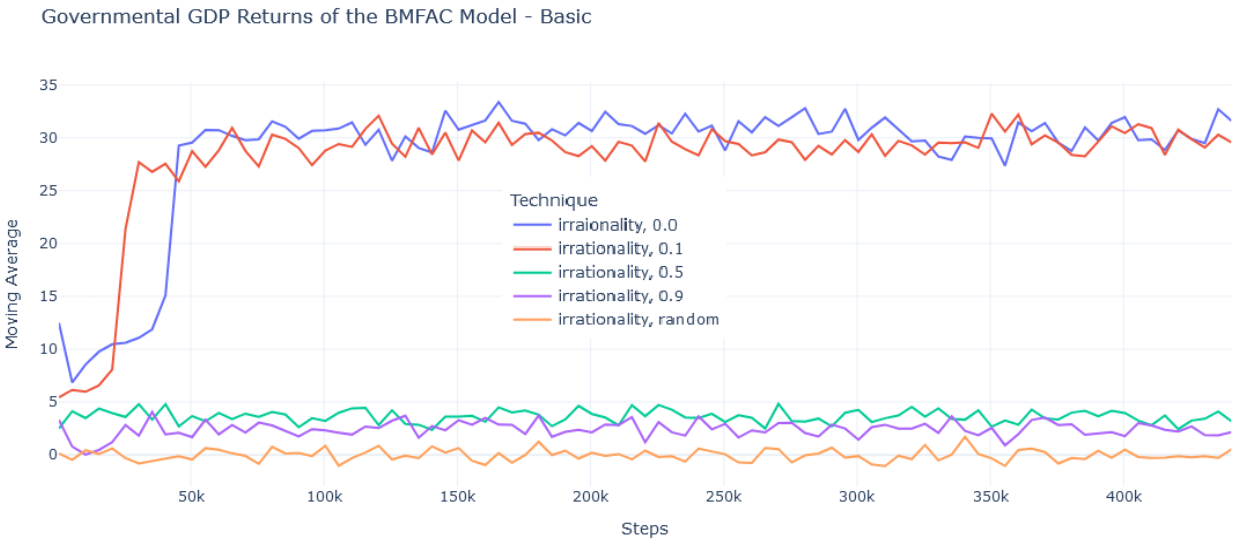

Figure 7. Basic BMFAC Algorithm Results of Governmental GDP Return Under Different Household Irrationality Levels

5.2.2. Results with the advanced irrationality model.

Figure 8 shows a similar observation as in Figure 5 when I switch to the advanced irrationality model. IPPO is not sensitive to irrationality levels unless the agents become completely irrational, i.e., $\epsilon = 1.0$.

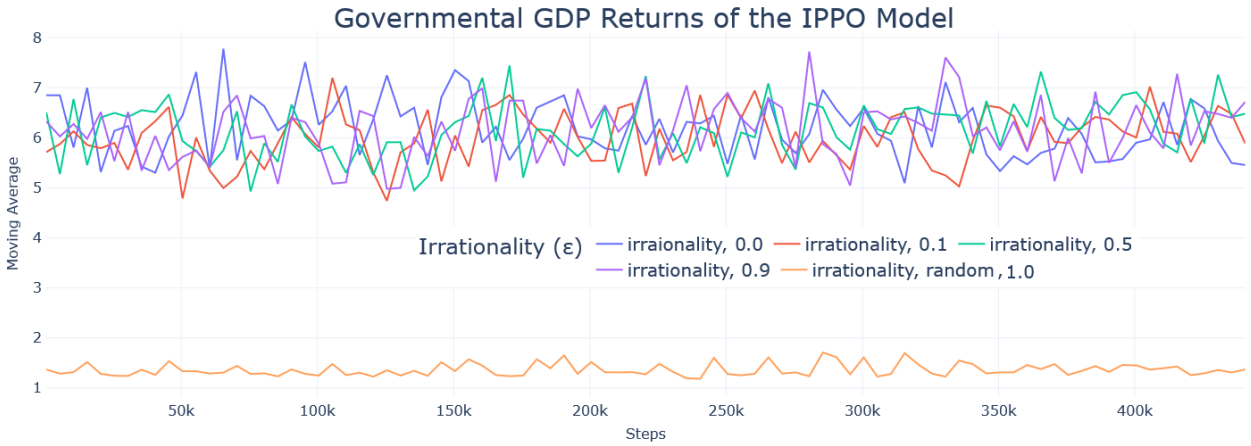

Figure 8. IPPO Algorithm Results of Governmental GDP Return Under Different Household Irrationality Levels

Figure 9 also shared a similar finding as in Figure 6.

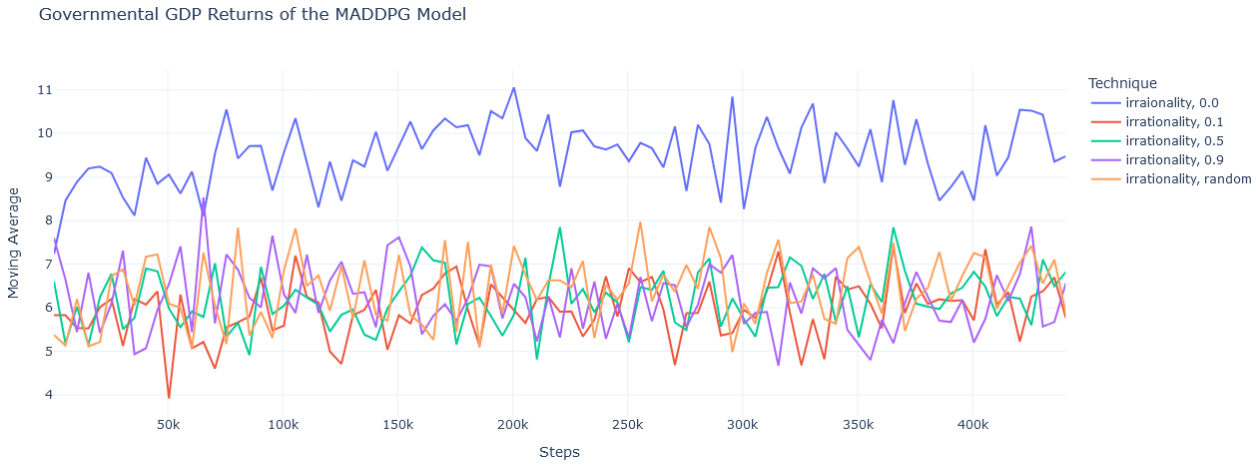

Figure 9. MADDPG Algorithm Results of Governmental GDP Return Under Different Household Irrationality Levels

Figure 10 also shared a similar finding as in Figure 7: the algorithm is robust to a low level of irrationality and performed the best among all three algorithms. However, with higher levels of irrationality levels, the performance went down significantly.

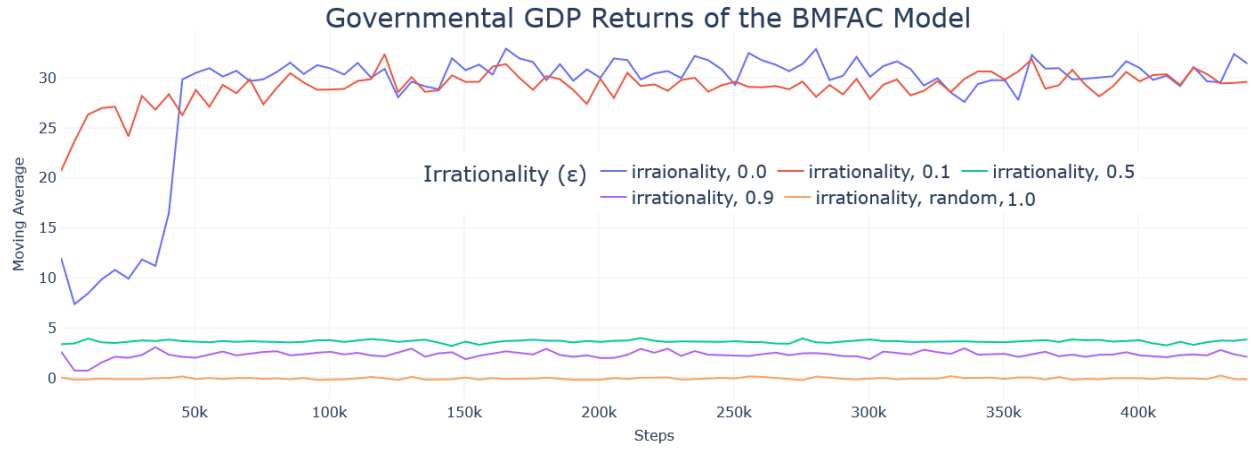

Figure 10. BMFAC Algorithm Results of Governmental GDP Return Under Different Household Irrationality Levels

6. Conclusion

In this work, I focused on verifying the effectiveness of different reinforcement learning algorithms for tax policy making in simulators when the households are not perfectly rational decision-makers as I observe in practice. Towards this goal, I proposed two different irrationality models and integrated them with three different RL algorithms in the TAXAI simulator for experiments. I selected the GDP growth as our reward. The experimental results showed that most of the performant RL algorithms and tax policy-making simulators are not robust enough to higher level of irrationality levels, i.e., their performances degrade significantly when I introduced some randomness in the households decision-making process. Such results suggest that there is a strong need for further RL algorithm and tax simulator development to consider such practical factors for better tax policies.

Acknowledgments

Authors wishing to acknowledge assistance or encouragement from colleagues, special work by technical staff or financial support from organizations should do so in an unnumbered Acknowledgments section immediately following the last numbered section of the paper.

Appendices

Technical detail that it is necessary to include, but that interrupts the flow of the article, may be consigned to an appendix. Any appendices should be included at the end of the main text of the paper, after the acknowledgments section (if any) but before the reference list. If there are two or more appendices they should be called appendix A, appendix B, etc. Numbered equations should be in the form (A.1), (A.2), etc., figures should appear as figure A1, figure B1, etc. and tables as table A1, table B1, etc.

References

[1] Adam Smith. The wealth of nations [1776], volume 11937. na, 1937.

[2] Eric Bonabeau. Agent-based modeling: Methods and techniques for simulating human systems. Proceedings of the national academy of sciences, 99 (suppl_3): 7280 – 7287, 2002.

[3] Roger H. Gordon and Wojciech Kopczuk. Using behavioral economics to inform the optimal taxation of labor income. National Tax Journal, 70 (4): 107 – 126, 2017.

[4] Qirui Mi, Siyu Xia, Yan Song, Haifeng Zhang, Shenghao Zhu, and Jun Wang. Taxai: A dynamic economic simulator and benchmark for multi-agent reinforcement learning. arXiv preprint arXiv: 2309.16307, 2023.

[5] Richard S. Sutton and Andrew G. Barto. Reinforcement Learning: An Introduction. MIT Press, 2 editions, 2018.

[6] Berkeley University of California. Cs188 lecture notes on rl and mdps, 2024.

[7] Stephan Zheng, Alexander Trott, Sunil Srinivasa, David C Parkes, and Richard Socher. The ai economist: Taxation policy design via two-level deep multiagent reinforcement learning. Science advances, 8 (18): eabk2607, 2022.

[8] El Baroudi Nedaa, Fabio Caccioli, and Adrien Vignes. Agent-based modeling in economics and finance: Past, present, and future. Journal of Behavioral and Experimental Finance, 34: 100722, 2022.

[9] AICPA. Guiding principles of good tax policy: A framework for evaluating tax proposals', 2001.

[10] Haiyang Chen, Hyung Jin Chang, and Andrew Howes. Implications of human irrationality for reinforcement learning. arXiv preprint arXiv: 2006.04072, 2020.

[11] Daniel Kahneman and Amos Tversky. Prospect theory: An analysis of decision under risk. Econometrica, 47 (2): 263 – 291, 1979.

[12] R. S. Sutton and A. G. Barto. Reinforcement Learning: An Introduction. MIT Press, 2018.

[13] R. Bellman. Dynamic Programming. Princeton University Press, 1957.

[14] M. L. Littman. Markov games as a framework for multi-agent reinforcement learning. In Proceedings of the 11th International Conference on Machine Learning (ICML – 94), 1994.

[15] David Silver, Guy Lever, Nicolas Heess, Thomas Degris, Daan Wierstra, and Martin Ried- miller. Deterministic policy gradient algorithms. In International Conference on Machine Learning (ICML), pages 387 – 395. PMLR, 2014.

[16] John DeNero and Dan Klein. Cs 188: Introduction to artificial intelligence. URL: https://inst. eecs. berkeley. edu/ ~ cs188, 2018.

[17] Christian Schröder de Witt, Tarun Gupta, Denys Makoviichuk, Viktor Makoviychuk, Philip H. S. Torr, Mingfei Sun, and Shimon Whiteson. Is independent learning all you need in the starcraft multi-agent challenge? CoRR, abs/2011.09533, 2020.

[18] John Schulman, Filip Wolski, Prafulla Dhariwal, Alec Radford, and Oleg Klimov. Proximal policy optimization algorithms, 2017.

[19] Haifeng Zhang, Weizhe Chen, Zeren Huang, Minne Li, Yaodong Yang, Weinan Zhang, and Jun Wang. Bi-level actor-critic for multi-agent coordination, 2020.