Comparison between Conditional Cash Transfer and Universal Basic Income using Multi-agent Reinforcement Learning

Lokeshwaran Arunachalam

Student ID: 20497614

School of Computer Science, University of Nottingham.

Submitted September 2023, in partial fulfilment of the conditions for the award of the degree of MSc Data Science.

I declare that this dissertation is all my own work, except as indicated in the text.

Abstract

This dissertation explores the impact of two wealth redistribution policies, Universal Basic Income (UBI) and Conditional Cash Transfer (CCT), by balancing both productivity and social welfare. Rising inequality is a major concern in society today. Governments face challenges in implementing suitable policies to reduce inequality without compromising economic growth. UBI and CCT have been widely debated as options, but their impacts are not fully understood. This research aims to compare UBI and CCT using multi-agent reinforcement learning, where worker agents try to maximize productivity and a social planner agent seeks to improve equality through taxation and redistributing funds to workers. The agents were trained in the AI-economist simulation environment, which reflects real-world complexity and heterogeneity. The AI-economist simulation offers flexible features called components to add additional features to the environment. UBI and CCT policies were implemented in AI-economist using component functionality. The Proximal Policy Optimisation (PPO) algorithm is used to train the AI agents. The worker agent was trained using the curriculum learning technique, while the planner agent was trained by adaptive learning. Simulation results showed that the UBI policy performed slightly better in terms of productivity, especially in the long term. However, the productivity of both redistribution policies stagnated after a certain threshold period. Both redistribution policies improved social welfare. The research demonstrates how multi-agent reinforcement learning can be used to optimize and analyse complex economic policies by balancing multiple objectives. Further work could incorporate training agents using human behavioural data, considering additional factors which could assess social welfare more accurately and the integration of inter-agent relationships.

Keywords: Universal Basic Income, Conditional Cash Transfer, Multi-Agent Reinforcement learning, Optimizing Economic policy, Comparison between Universal Basic Income and Conditional Cash Transfer, Proxy Policy Optimization, Reinforcement learning, Economic Agents.

Acknowledgements

I am grateful to my supervisor, Tomas Maul, for his expert guidance and feedback throughout this project. His insights significantly strengthened my dissertation.

I also wish to thank the faculty members whose foundational instruction equipped me with the academic skills needed for this research endeavour.

Finally, I profoundly appreciate my family, especially my parents, for their unwavering encouragement and belief in me. Their steadfast support kept me motivated and enabled me to complete this achievement.

I have been fortunate to receive such helpful guidance and backing. My dissertation has benefited greatly from the contributions of those I have acknowledged here.

Table of Contents

List of Figures

List of Table

1 Introduction

1.1 Background and Motivation

In the last four decades, the middle and lower classes have experienced a stagnation in economic growth. Even though there is substantial growth in the economy, increased inequality would imply that wealthy people would have more resources and influence in the government, which is a threat to democracy. The exponential rise in Artificial intelligence and automation was considered as a great threat to workers. In the name of productivity, many companies began to lay off workers and replacing them with automation robots. To alleviate growing inequality among people, governments introduced many wealth redistribution policies. Conditional Cash Transfer (CCT) and Universal Basic Income (UBI) were the two widely debated wealth redistribution policies among governments. However, the distribution of wealth can have an adverse effect on productivity. Therefore, governments need to implement an optimal policy that is able to balance between economic growth and inequality.

There were many studies that were conducted to analyse the ability of the government to fund the UBI and CCT policy and the impact of the policy on economic growth and inequality. (Saez, 2002) analysed and framed optimal taxation based on the behaviour response in labour supply for targeted transfer programs using numerical simulation. (Stoeffler, Mills and del Ninno, 2016) studied the effectiveness of the Cash Transfer programs in Cameroon and found that the Community Based Targeting (CBT) method exhibits suboptimal performance in selecting poor households for the Cash Transfer program when compared to the Proxy Means Testing (PMT) method. An Interesting study by (Zheng *et al.*, 2020) on optimising and testing tax policies in an economic simulation environment in which agents and government learn to optimise their individual objectives of productivity and social welfare. The developed model performance on tax policies showed that 16% improvement in equality and productivity when compared to other tax frameworks.

(Špeciánová, 2018) conducted a study on the feasibility of funding the Basic Income policy in the Czech Republic using the microsimulation study made by other countries such as Belgium and Spain and found that an increase in tax revenue is needed to fund Basic Income. However, the impact on productivity because of additional taxation is not considered. Another study by

(Marinescu, 2018) states that a 10% increase in income as a result of the Unconditional Cash Transfer program resulted in a 1% decrease in labour supply. (Boadway, Cuff and Koebel, no date) used a complex tax-bracket structure to fund the Basic Income (BI) policy for Canada, but the policy led to a decrease in labour supply, which resulted in a budget deficit of $5 billion. The analysis of different tax bracket rates or labour supply elasticities is not studied because the numerical calculation offers limited opportunity to study those complex dynamics (Boadway, Cuff and Koebel, no date). (Neves *et al.*, 2022) conducted research on Cash Transfer program in Brazil and found Cash transfer policy has many positive impacts like improvement in public health services, child mortality rate is reduced and increased access to food. Although the long-term impact of the Cash Transfer program is not known.

The two major limitations that were found in the above research studies were a way to analyse the impact of the UBI and Targeted Transfer policy in the long term and balancing both economic productivity and social welfare of the people.

1.2 Problem Statement

The rising inequality is a significant concern in today's society. The governments are facing challenges in implementing a suitable policy to alleviate inequality without compromising on economic productivity. Universal Basic Income (UBI) and Conditional Cash Transfer (CCT) were the two widely debated options. Therefore, the aim of this research report is to compare two wealth redistribution methods, UBI and conditional cash transfer policies and their impact on productivity and equality by using Multi-agent Reinforcement learning where the worker agents will try to increase their productivity while the planner agent will seek to improve equality by imposing tax on worker agents. The tax income collected by the planner agent will be used to fund the policies.

1.3 Research Question

1) What is the short-term and long-term effect of UBI and CCT policies on equality and productivity?

2) Do redistribution policies contribute to an increase in people's income?

3) which redistribution policy promotes consumption?

1.4 Methodology and Experimental Design

The objective of this research is to compare UBI and CCT policies by optimizing equality and productivity. Therefore, a simulation environment with two types of agents which can reflect people and government was selected. The redistribution policy functionalities were added to the simulation, and the agents were trained based on two objectives. The trained agents were then used to simulate the economic policies. A comprehensive exploration of experimental design was discussed below.

- Simulation Environment: An advanced simulation environment that is able to replicate complex economic dynamics is selected. The entities which will be used within the simulation environment and the conditions of the agent to interact were framed.
- Redistribution policy integration: The functionality of UBI and Targeted Transfer policy were integrated into the simulation environment.
- Multi-agent Wrapper: The Multi-agent wrapper is an additional layer of abstraction and contains the standard functions which is used to assist the Reinforcement algorithm to train multiple agents using different policies.
- Proximal Policy Optimization (PPO) model implementation: The PPO algorithm is used to train the policy of an agent. The agent policy was trained using the Custom Neural Network model. Actor and Critic were the two main components of PPO. Critic is used to estimate the state value function, whereas Actor is used to evaluate the probability of the next action that an agent should take in the simulation environment.
- Curriculum Learning: The curriculum learning is designed in a way that worker agents will try to learn easier tasks first, and then the difficulty of the task will be gradually increased. The experience and the skill gained in previous tasks will help the agent to learn more complex tasks.
- Two-level Learning: The worker and planner agents were trained parallelly. The planner agent action will depend upon the performance of the worker agents. For instance, If the productivity of all worker agents were high then a lower tax rate would be enough to collect a sufficient amount of tax income, which would be used to fund redistribution policies. The UBI and CCT were trained separately.

- Model Evaluation: The metrics and methods used to evaluate the effectiveness and performance of reinforcement learning model were addressed.

1.5 Structure of the report:

The research report is organised into the following sections:

- Introduction: This foundational section explains the motivation for the report, related research works, the limitations in the research were found, the objectives of the research were framed, and the experimental design of the report was discussed.
- Literature review: This chapter reviews the existing research in redistribution policies, simulation environment in economics and Reinforcement learning.
- Methodology: This section presents a comprehensive examination of the research methodologies and procedural framework employed in this study. The reason behind the choice of particular methodologies and their advantages over alternative approaches were discussed briefly.
- Experimental Design: This chapter discusses practical steps and decisions that were followed in experimental design. It encompasses the configuration of the simulation environment, neural network model implementation and training AI agents using Multi-agent Reinforcement learning.
- Results: The results of the trained model and key findings of economic policies were explained and interpreted.
- Discussion: The results of the redistribution policy simulations were interpreted, compared and the benefits and drawbacks of the policy were discussed briefly.
- Conclusion: Summarizes the important findings of the report and further research works in this field were explored.

2. Literature Review

In the realm of economic policy, addressing wealth inequality while sustaining economic growth is pivotal. This literature review explores Universal Basic Income (UBI) and Conditional Cash Transfer (CCT) policies, emphasizing productivity and social welfare. The AI-driven simulations in economics and Multi-agent Reinforcement learning approaches were briefly studied in this literature review.

2.1 Economics

Economics is the study of decisions people make and the impact of those decisions on the economy. Economics also investigates major issues in society such as inequality, sustainability, and unemployment and how governments should make decisions to resolve those issues (Sloman, Guest and Garratt, 2021).

2.2 Redistribution of wealth

Classical Economists believed that improving economic growth is important for making people's life better. However, they later realised that economic growth alone is inadequate to improve social welfare. Hence, governments have initiated many redistribution policies to alleviate rising income inequality (Kakwani and Son, 2022).

According to an experiment conducted by (DeScioli, Shaw and Delton, 2018) participants can make the decision whether to invest or keep the money. The player can choose from four random investments in which there is a 25% chance that a player can earn 180 cents and a 75% chance that the player will lose all the invested amount. The participants were divided into four groups based on their decision. If all participants decide to invest and share the gain, then each player can get a profit of 5 cents. However, if some players decide to keep 40 cents instead of investing, even if players share the gain, there is a possibility of loss. In the game, there were two scenarios. In the first, no information about the participant's investments was given. Whereas in the second case, the participant's information was shared. It is found that 68% of participants choose to invest when information regarding investment is shared, whereas when information is not shared because of fear of loss, only 47% participants decided to invest. This

experiment shows that people will hesitate to invest if there is uncertainty. In real life scenario people will likely make investments only if they get some kind of insurance from the government. The redistribution of wealth from high-income individuals to low-income individuals can protect people from financial loss.

However, complete equality will hinder economic growth. According to the study conducted by (Benhabib, 2003) states that a decrease in inequality could lead to an increase in economic growth but when we reach a point of sustainable cooperation. The effort of the productive people will decline since those who work diligently and those who do not work will earn the same income regardless of the effort exerted. Therefore, it is essential to balance both growth and inequality.

The Conditional Cash Transfer (CCT) and Universal Basic income (UBI) were the two most debated wealth redistribution policies.

2.3 Conditional Cash Transfer:

The study by (Rawlings and Rubio, 2005) regarding cash transfer programs in countries like Mexico, Nicaragua, Turkey, and Jamaica explains that transfer programs played a major role in poverty reduction and it also promoted higher consumption levels. The consumption level in Mexico rose 13% after the implementation of conditional cash transfers. In times of crisis, the conditional fund transfer policy will play an important role in protecting people's consumption of fundamental needs. Many countries have used different scoring formulas to find poor households. Jamaica used annual consumption data, whereas Turkey and Nicaragua used the Proxy means test for the scoring rule.

According to the research by (Das, Do and Özler, 2005) conditional cash transfers may face drawbacks like low participation and fungibility. When the amount of funds is low, it may lead to low participation. For instance, policymakers want every child to attend the school by offering $5 to parents to make sure that their children will attend the class. Participation in the scheme may decline as a result of the insufficient fund available. The fungibility issue arises when people neutralize the impact of the policy by consuming only the substitute offered by the policy. Fungibility refers to a commodity which can be interchangeable with another commodity of the same type. The policy of providing free lunch in schools could make

students to skip breakfast. One approach to solve the fungibility issue is to closely analyse the close substitute consumption when policy is implemented.

(Hanna and Olken, 2018) have studied some of the challenges faced by developing countries when implementing conditional transfer programs. Most of the developed countries keep a record of the actual income of people. Hence, targeting low-income individuals is easy. However, in developing countries. most of the population would work in the informal sector therefore, obtaining true income from the people is an arduous task. Hence, governments use imperfect proxy means tests for calculating income, which leads to both inclusion and exclusion errors. The governments started the debate of implementing Universal Basic Income instead of targeted transfer because funds will be transferred to everyone, and there will be no conditions in the policy.

2.4 Universal Basic Income

The Universal Basic Income (UBI) is a regular payment given to everyone without considering income level. To implement this policy, governments need substantial amounts of revenues. The governments in Alaska and Iran were able to fund this policy using revenues from abundant natural resources (Hanna and Olken, 2018).

The paper (Hall *et al.*, 2019) discusses the strategies that can be used to fund the UBI policy by excluding other welfare programs, income from capital ownership, inclusive capitalism, and taxing high-income individuals. (Hall *et al.*, 2019) argue that inclusive capitalism is a sustainable approach because the citizens would receive dividend payments from the capital investments made by the government. If the governments make investments in sustainable goods and services, the money will circulate among corporations and promote growth and profit from corporations will be shared with people in the form of dividend payments. However, inclusive capitalism always poses a risk because the stock market is volatile, and the chances of a downturn are high.

The reliable option for developing countries to finance the Universal Basic Income policy is domestic taxation. However, the UBI policy may not raise take-home income for everyone because to fund the policy the marginal tax rate would be increased for some individuals (Hanna and Olken, 2018).

(White, no date) argue that advances in artificial intelligence and automation would eliminate many jobs in the digital economy. Starting from the early 1980's labour productivity began to surpass the growth rate, and the difference in these two factors rose above 10% in the late 1990s. This difference in productivity and growth rate puts more pressure on labour intensive jobs like healthcare and education, which are regarded as unproductive. The impact of the exponential rise in automation is substantial in the industrial process; many workers were laid off in the name of productivity. Therefore, Universal Basic Income would provide financial support and give opportunity for people to resign and move to a worthwhile occupation.

An interesting study by (Marinescu, 2018) on Alaska's permanent fund states that people who receive the fund spend most of their money on non-durable goods. The consumption rate is increased whenever the people received the funds. This clearly states that redistribution policies will promote consumption.

2.5 Simulation Environment

In this dynamic and competitive, world simulation is used as an instrument to aid decision making across various domains such as manufacturing, healthcare, and public services (León-Castro *et al.*, 2019). After the invention of method Monte Carlo in 1947, numerous simulation technique has emerged which is used to find the result of an experiment which are hard to implement in the real world (Mourtzis, Doukas and Bernidaki, 2014).

The lack of data poses a great challenge when the government must make decisions regarding policies. A study made by (Dyson and Chang, 2005) shows that dynamic simulation modelling can be used to predict solid waste management in rapidly growing urban areas. The software used in the study covers important basic building blocks such as stocks and converters these basic blocks were used to create dynamic simulation, which can cover uncertainties and other possible outcomes. The traditional statistical methods were unable to handle these cases.

2.6 Agent-Based Computational Economics

Economic studies must be able to handle difficult complexities in the real world, such as strategic interaction and collaborative learning. Agent-Based Computational Economics (ACE) is used to study economic processes with the help of interacting agents in a dynamic simulation environment (Tesfatsion, 2006).

According to the research by (Colin F. Camerer, 2011) on agent-based behavioural game theory, agents can observe and learn the environment and are able to adjust their behaviour based on previous experience. The agent's actions were similar to actual human behaviour in the real world.

The advantage of using Agent-Based Computational Economics when compared to other modelling approaches is the autonomy of the agents, where each agent can make their own decision (Tesfatsion, 2006).

"An autonomous agent is a system situated within and part of an environment that senses that environment and acts on it, over time, in pursuit of its own agenda and so as to effect what it senses in the future." (FRANKLIN, 1997)

Some of the drawbacks of ACE modelling is that constructing a complete economic model is difficult because the model needs detailed specification, which reflects the complexity of the real world, and it is not known whether the ACE model will be able to scale up to large-scale systems (Tesfatsion, 2006).

A study by (Holland and Miller, 1991) states that Artificial Adaptive Agents (AAA) were better than mathematical models. The AAA models are flexible, and the models can be interpreted at any stage. This important feature helps the economist to tune the model based on their behaviour.

2.7 Reinforcement Learning

Reinforcement learning is learning how to select actions based on observation to maximize reward. The agent does not know which steps to take initially, so it should test some actions and learn which action result in the largest reward.(Sutton and Barto, 2018).

Despite notable progress in Reinforcement learning, it still faces difficulty in finding a balance between exploration and exploitation. Reinforcement learning cannot use its knowledge until sufficient experiences are collected through exploration. The paper (Zhu *et al.*, 2023) suggests Transfer learning or shared knowledge methods to solve this issue.

(Zhu *et al.*, 2023) used Half cheetah simulation as a standard benchmark for Deep Reinforcement Learning. Half cheetah is a two-leg agent, which is a self-balancing robot

trained to run fast. Reward Shaping approach mentioned by (Zhu *et al.*, 2023) was used in Half Cheetah in which an external reward based on expert knowledge was added in addition to the environment reward. Therefore, the agent will be trained using a newly modified reward function.

However, another study by (Narvekar, 2017) states that curriculum learning will be better than existing Transfer Learning methods. In curriculum learning, the agent will be given an easy task first, and then difficulty level is steadily increased. Therefore, the agent will try to solve complex problems based on the knowledge gained from the previous basic tasks.

One of the main challenges of reinforcement learning is handling large amounts of observational data and the time taken to train the algorithm. To handle this problem (Espeholt *et al.*, 2018) developed a new technique called Importance Weighted Actor-Learner Architecture (IMPALA), where Reinforcement algorithm can be trained in many distributed machines by decoupling action and learning. This method is called V-trace.

2.7.1 Deep Reinforcement Learning

The paper (Mnih *et al.*, 2013) released by the DeepMind technologies implemented the first Reinforcement learning using Deep Neural Network (DNN). The DNN application needs many labelled data, but reinforcement learning has only sparse and noisy data, and it will take many sequential iterations to create valuable data to overcome these challenges (Mnih *et al.*, 2013) have used a Convolutional Neural Network with Q-learning and stochastic gradient descent. The model was trained using Deep Reinforcement learning without specifying any information about the game and the hyperparameters were kept constant throughout the game. The trained model outperformed every reinforcement learning algorithm invented at that time.

A similar approach was tried by (Riedmiller, 2005) using the Resilient Backpropagation (RPROP) algorithm to update the weights of Q-learning network parameters, but the cost of each iteration was very high because they used batch update instead of stochastic gradient descent.

An interesting study by (Hodge, Hawkins and Alexander, 2021) on Drone navigation states that using the Long-Short Term Memory model in reinforcement learning helps the agent to learn about obstacles that were faced in the past. The drone agent was first trained using

incremental curriculum learning. After learning basic aerodynamics from the simulation environment, the agent was trained in the physical world. The technique of training environment in simulation and then introducing it to the real world saves resources and time.

2.7.2 Proximal Policy Optimisation Algorithm

(Schulman *et al.*, 2017) have introduced a new algorithm called Proximal Policy Optimisation (PPO), which is a variant of advantage actor critic (A2C). The Trust Region Policy Optimisation (TRPO) algorithm will constrain the policy update by defining a trust region which restricts the policy to make large update from the old policy. The PPO algorithm uses a clipped probability ratio, which restricts the probability ratio to move away from 1. (Schulman *et al.*, 2017) compared the performance of the PPO algorithm on continuous problems with other tuned algorithms like TRPO, cross-entropy method (CEM) and vanilla policy gradient. The PPO algorithm gave better performance using the same clip hyperparameter value 0.2.

Another interesting experiment was conducted by (OpenAI *et al.*, 2019) on training an agent to defeat the Dota 2 game world champions. The agent is trained in complex environment which have high dimensional continuous states and action spaces. One of the key issues is scaling the ability of training algorithm in complex game environment. Therefore, distributed training systems with thousands of GPUs were used, and they created tools to store the learned policy, which is very crucial because they don't have to restart the training. They can resume the training from the saved policy. (OpenAI *et al.*, 2019) used the Long Short-Term Memory (LSTM) model, a variant of recurrent neural network to capture complex historical patterns which helps agents to make better decisions. The PPO algorithm was used to train the agent and the trained agent defeated the Dota 2 game world champions.

2.7.3 Multi-agent Deep Reinforcement Learning:

Multiple autonomous agents engage with one other in a shared environment in Multi-agent Reinforcement Learning (MARL). The MARL is considered an alternative to centralised systems. The agents can exchange their knowledge so that the learner agent can emulate competent agents, and if some agents fail, their duty can be taken over by other agents, which is one of the benefits of Multi-agent Reinforcement learning (Buşoniu, Babuška and De Schutter, 2008).

(Su *et al.*, 2022) undertook an experiment to design a Preventive Maintenance (PM) policy using Multi-agent Reinforcement Learning. In a complex environment like manufacturing systems, single-agent reinforcement learning faces many complexities like non-linearity and stochasticity. Another drawback in single-agent Reinforcement Learning is that action space will increase when we add additional manufacturing systems, which lead to detrimental impact on performance. The centralized Reinforcement learning framework in which every manufacturing system is controlled by a central agent. The central agent's aim was to improve the overall performance of the manufacturing system rather than the performance of an individual manufacturing system. Because of the above-mentioned limitations, MARL is implemented where each agent makes individual decisions and optimises their own policy. Hence, (Su *et al.*, 2022) solved the Preventive Maintenance (PM) problem using a decentralised partially observable Markov Decision Process (Dec - POMDP) in which they adopted centralised training and decentralised execution (CTDE).

A similar study by (Trott *et al.*, 2021) in which MARL agents were trained to solve complex socioeconomic issues like striking a balance between productivity and wellness of the people. The objective of the problem is to develop a model to balance public health and economic policy amidst the COVID-19 crisis in the United States. The US states will try to impose control measures to improve public health, whereas the federal government manages the distribution of direct payment subsidies to the US states. The US state agents and Federal government agents were trained in two separate policies. The Federal government agent's actions will affect the social welfare of the US state agents and vice versa. Therefore, it is a two-level learning problem, which is a great challenge to optimise because if the policy of one actor is changed, the reward function of other actors will change significantly. Hence, policy design configuration is different from normal Deep RL problems. The trained policy using Multi-agent RL leads to substantial improvement in social well-being compared to previous outcomes from government policies.

A similar study by (Zheng *et al.*, 2020) on designing and analysing US Federal and Saez tax policies using Multi-agent Reinforcement learning. The tax policies optimised using reinforcement learning agents improved the social welfare of the agents. The AI agent used optimal tax rates which are different from the tax policies followed by the government. The AI agent was trained without any basic knowledge about economic policy. The agent learned the policy only by trial and error.

The MARL also have some drawbacks such as an exponential increase in state-action spaces for algorithms like Q-learning because the value function for Q-learning is based on state-action pairs. Therefore, computational complexity will also increase exponentially. Another drawback is that complications may arise in exploration strategies because every agents will explore the environment, which leads to excess exploration which can also disrupt the learning task of MARL agents (Buşoniu, Babuška and De Schutter, 2008).

2.8 Summary

The research presented above demonstrates the methods and impact of wealth redistribution policies in different countries.

The governments implement many redistribution policies to improve social welfare and reduce inequality. According to the research by (DeScioli, Shaw and Delton, 2018) and (White, no date), funds received by redistribution policies would give financial support and encourage people to make investment.

The implementation of the Conditional Cash Transfer (CCT) policy in countries like Mexico improved the consumption of goods among people (Rawlings and Rubio, 2005). However, (Hanna and Olken, 2018) imply that in many developing countries, income data is not available to the government. Therefore, the condition used to find people who are eligible for the fund is not straightforward, which leads to inclusion and exclusion errors.

Some governments consider Universal Basic Income as an alternative because of the drawbacks mentioned in the Conditional Cash Transfer (CCT) policy. The exponential rise in automation and artificial intelligence started to replace existing jobs. Therefore, Universal Basic Income would give much-needed time and support to the people (White, no date). However, the government need a large amount of fund for UBI policy. The reliable approach to collecting funds is domestic taxation. Hence, the marginal tax rates will be raised for some individuals (Hanna and Olken, 2018).

Reinforcement learning was used in many fields like health care, robotics, automation, and economic policy design. The reinforcement learning agent can learn complex problems using curriculum learning (Narvekar, 2017). The PPO algorithm outperforms other algorithms based

on research by (Schulman *et al.*, 2017) and (OpenAI *et al.*, 2019). Multi-agent Reinforcement learning supports agents to learn individual policies and their own objectives (Su *et al.*, 2022).

The income distribution, natural resources, productivity, and inequality vary for different countries. Therefore, the impact of a wealth redistribution policy in one country will not be the same in another country. Therefore, governments need a solution to test and optimize economic policy in a virtual world.

The research on Agent-Based Computational economics and multi-agent reinforcement states the possibility of implementing complex economic policies in the virtual world using Reinforcement Learning and simulation environment.

Considering this perspective, the following research question is framed and will be explored more in-depth within this paper.

Can Multi-agent Reinforcement learning can be used to test, analyse, and optimise Conditional Cash Transfer and Universal Basic Income policy?

3 Methodology

3.1 Introduction

The research study is about optimizing and analyzing two redistribution policies by balancing productivity and social welfare and comparing their results. The methodology plays a significant role in structuring the research design. This section provides a systematic approach to train a reinforcement learning model to optimize economic policies. The below flowchart (Figure 1) represents the systematic methodologies employed in this research.

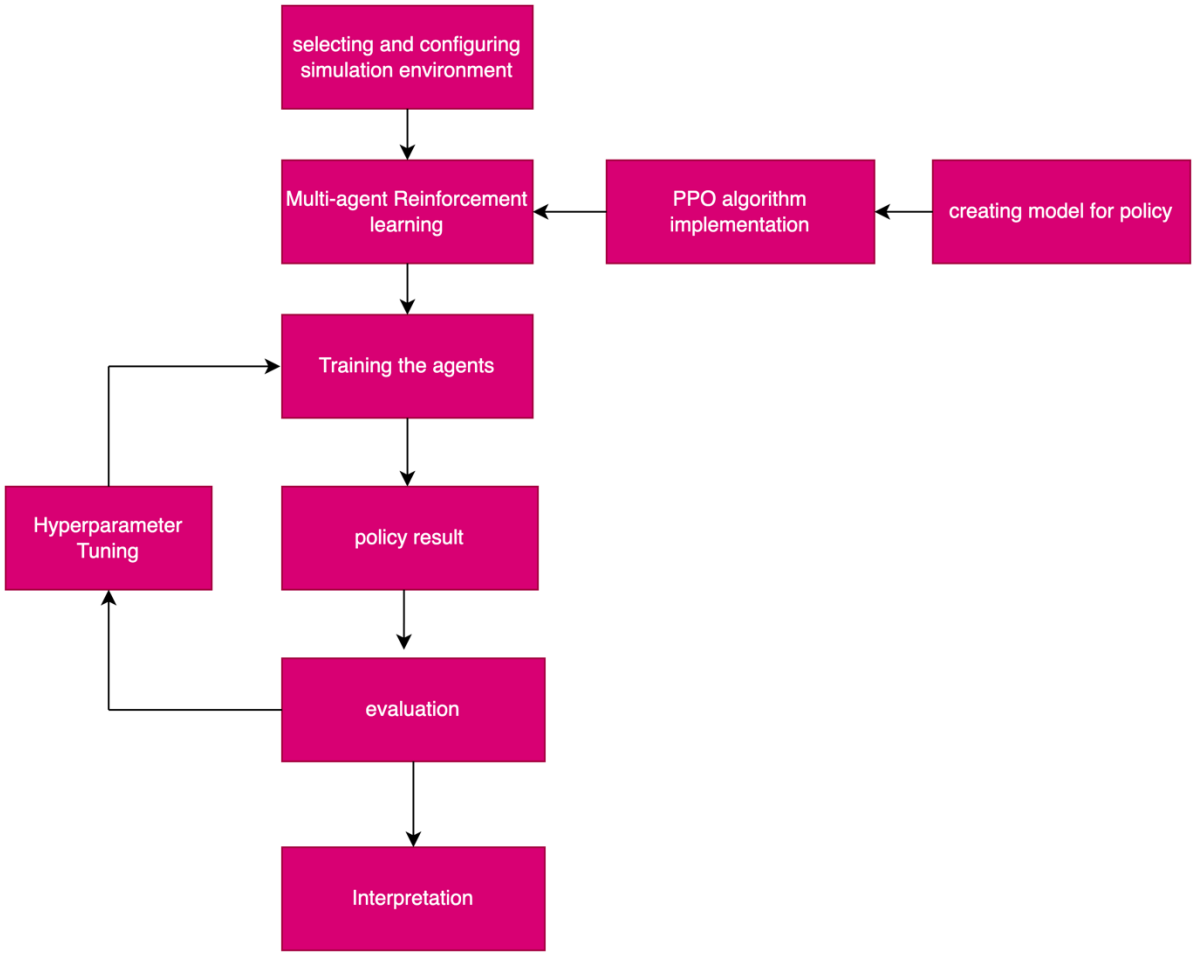

Figure 1: methodology flowchart

3.2 Economic Simulation

The AI-Economist simulation environment is specially designed for optimizing economic policies. The environment consists of worker and planner agents. The social planner agent is

used to optimize economic policies for the welfare of worker agents. The worker agents will try to increase their productivity. The simulation environment has options to configure the spatial layout to reflect the heterogeneity of economic situations in the real world. The simulation environment has a flexible functionality called a component. The new functionality of the policies can be added as a component. Therefore, the UBI and CCT policy functionalities were added to the simulation environment with the help of component functionality.

3.3 Multi-agent Reinforcement learning

The Agent-Based Modelling (ABM) approach is used to analyse the behaviour of economic agents in the simulation environment. The agents can interact with the environment based on the predefined rules. The agent's behaviour was framed based on the human experiments. However, in a complex environment, it is difficult to configure behavioural rules for the agent. Therefore, the Multi-agent Reinforcement Learning approach is used where multiple agents interact with each other and learn their own behaviour by trial and error. The multi-agent wrapper is implemented for the simulation environment to facilitate multi-agent reinforcement learning.

3.4 PPO Algorithm implementation

The reinforcement learning algorithm, like the Deep-Q learning algorithm, will give better accuracy. However, the observation from the simulation environment contains high-dimensional data. The curse of dimensionality increases the complexity of Q-learning to learn. The Trust Region Policy Optimisation (TRPO) algorithm works with multi-dimensional observational spaces. However, tuning hyperparameters in the TRPO algorithm is a time-consuming process. The Proximal Policy Optimisation (PPO) algorithm is used in this research because it is more stable than the TRPO algorithm and it requires fewer samples to learn efficiently compared to other reinforcement algorithms.

3.5 Creating model for policy

The Neural Network model is used to train policy for the agents. The actor and critic were the essential components to train an agent policy using the PPO algorithm. The two separate neural network models were created for the actor and critic components. The input for both the Actor and critic model is the observation of the agent, and the output for the actor model is action

probabilities. In contrast, the output for the critic model is state value. Each policy contains a separate actor and critic components.

3.6 Training the agents

Training agents in complex tasks without any guidance would lead to suboptimal performance. Therefore, curriculum learning is employed for worker agents, which leads to fast and stable learning. The planner agents were trained using an adaptive learning technique in which planner agents learn to optimize policy by understanding the behaviour of the worker agents. The planner agent was trained to optimize the economic policies of UBI and CCT.

3.7 Hyperparameter tuning

If the agent's policy were trained using suboptimal hyperparameters, then convergence on learning could be slower, or it could lead to unstable learning. Therefore, the agent's best policy is selected by tuning hyperparameters. The performance of the policy was evaluated using average episode reward.

3.8 Interpretation

The trained models for UBI and CCT redistribution policies were simulated, and the agent's income and inequality among agents in different timesteps were analyzed using charts.

4 Experimental Design

4.1 Simulation Environment

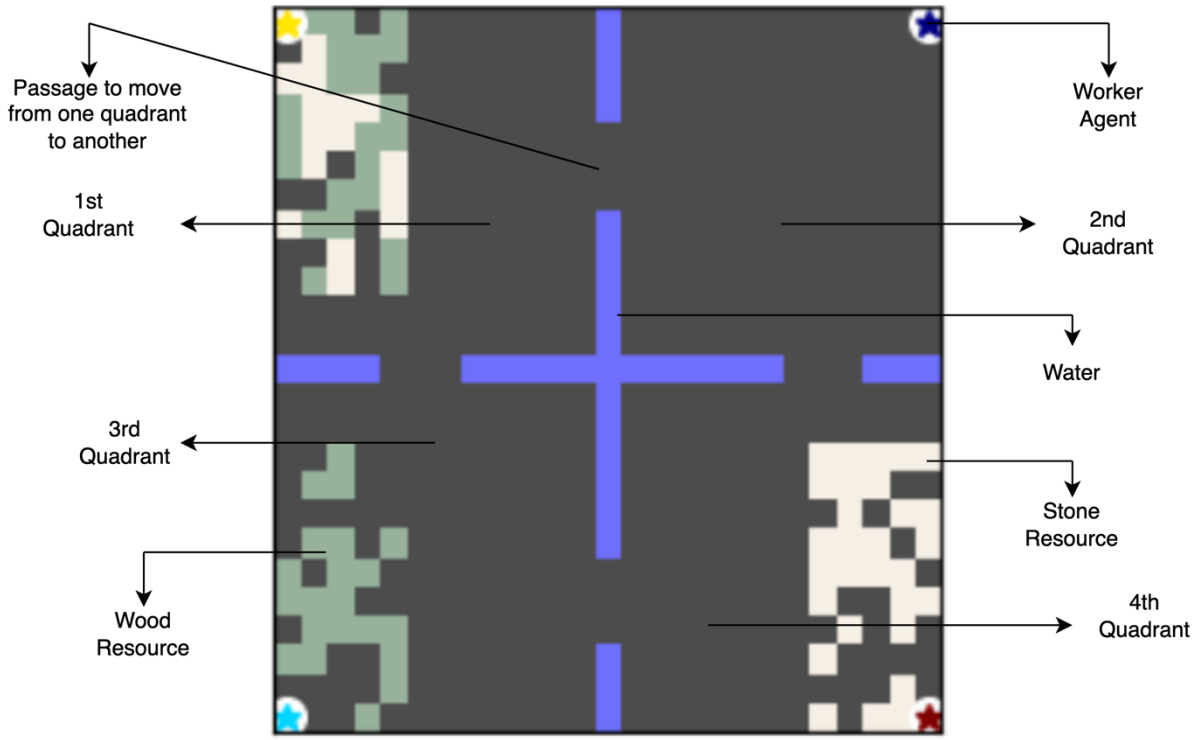

Figure 2: Two-dimensional simulation environment

The above figure depicts the 2D simulation environment used in this research. There are four agents in four corners of the simulation. The green and white boxes represents wood and stone resources.

The AI-Economist is a two-dimensional simulation environment is designed to optimize economic policies (Figure 2). The simulation environment is designed as a Markov Decision Process (Figure 3). At timestep t the agent (i) will receive an observation (s_t). The agent will choose an action based on policy $\pi_i(a_{i,t}|s_{i,t})$. Based on the selected action $(a_{i,t})$ the simulation environment will shift to the next state s_{t+1} and the agent will receive a reward $r_{i,t}$. The policy of the agent will always try to maximize the sum of discounted future reward.

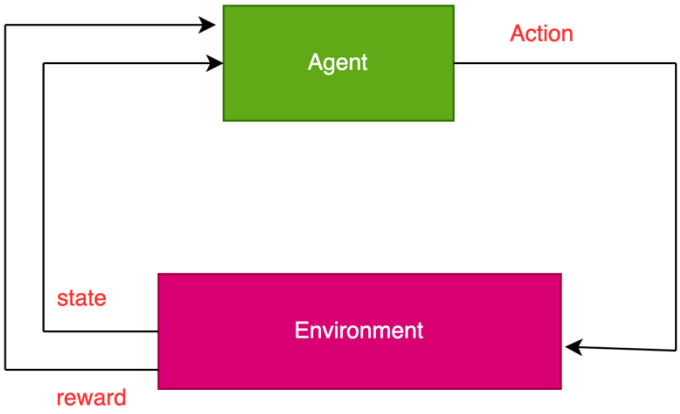

Figure 3: Markov Decision Process (MDP)

The worker and planner were the two types of agents in the simulation environment.

The actions that can be taken by the worker agent were:

- Move: The agent can move in four directions. The agents cannot move through water or houses.
- Collect: The wood and stone were the two resources that an agent could collect.
- Build: The agent can build the house using one unit of wood and stone gathered from the simulation environment.
- Trade: The worker agent can buy resources from other agents by exchanging coins. An agent can request to buy a resource from an agent by paying a certain number of coins. The agent with resources can accept or reject the bid.

Each agent has different skills. The number of coins earned by an agent for building a house is based on the agent's skill. The income for an agent is the number of coins an agent can earn. The agents can earn coins by selling resources or by building houses. The agent's skill and the resources near the agent's location were used to define the heterogeneity between the agents. Therefore, the economic simulation reflects people in the real world with differences and variations.

The wood and stone were the two types of resources that an agent can collect. The resources are scarce and will respawn randomly at specific locations. The planner agent doesn’t occupy any location in the two-dimensional world. The planner agent action was to assign tax rates to worker agents.

The worker agent will receive a labour cost based on the agent's action. The labour cost is accumulated over the period. The labour cost is inversely proportional to the reward, whereas an increase in the income of the agent will also increase the reward of the agent.

4.1.1 Worker agent reward

The reward of the worker agent is based on productivity. The agent's productivity will depend upon how much an agent can earn by doing minimum labour. The reward for the worker agent is calculated based on the concave utility function. The value of the utility function will increase when the agent earns more coins, whereas the value will decrease if there is an increase in labour.

$$utility(coin, labor) = crra(coin) - labour$$

$$crra(coin) = \frac{coin^{(1-\eta)} - 1}{1 - \eta}$$

CRRA is also known as the isoelastic utility function, which is used to add non-linear behaviour to the agents. The variable η is the degree of risk aversion if the value of η is 0 then the agent is not worried about taking risk and the agent wealth is not affected by the risk factor whereas if $\eta > 0$, then the CRRA utility value will be low when compared to coins earned. If the CRRA utility value is low then the agent will hesitate to explore new actions because the reward will be low. In this research, the risk aversion factor is set to 0.23. Therefore, agent is willing to take some risk. However, the agent still values the income earned to some extent. The larger η value leads to higher deviation from normal tendency of the agents.

4.1.2 Planner agent reward

The reward for the planner agent is the product of equality and total productivity. The total productivity is the sum of all worker agent's income.

$$social\ welfare = \ total\ productivity \times Equality$$

$$Equality = 1 - Gini\ coefficient$$

The Gini coefficient is used to measure the inequality of the agent's wealth. The numerical value given by the Gini coefficient is between 0 and 1. If the value of the Gini coefficient is 1

then it represents absolute inequality whereas Gini coefficient value 0 represents the perfect equality.

$$Gini\ coefficient = \frac{\sum_{i=0}^{n-1}\sum_{j=0}^{n-1}\left|coin_i - coin_j\right|}{2n\sum_{i=0}^{n-1} coin_i}$$

$n : number\ of\ agents$

$coin_i : amount\ of\ coin\ earned\ by\ an\ agent\ i$

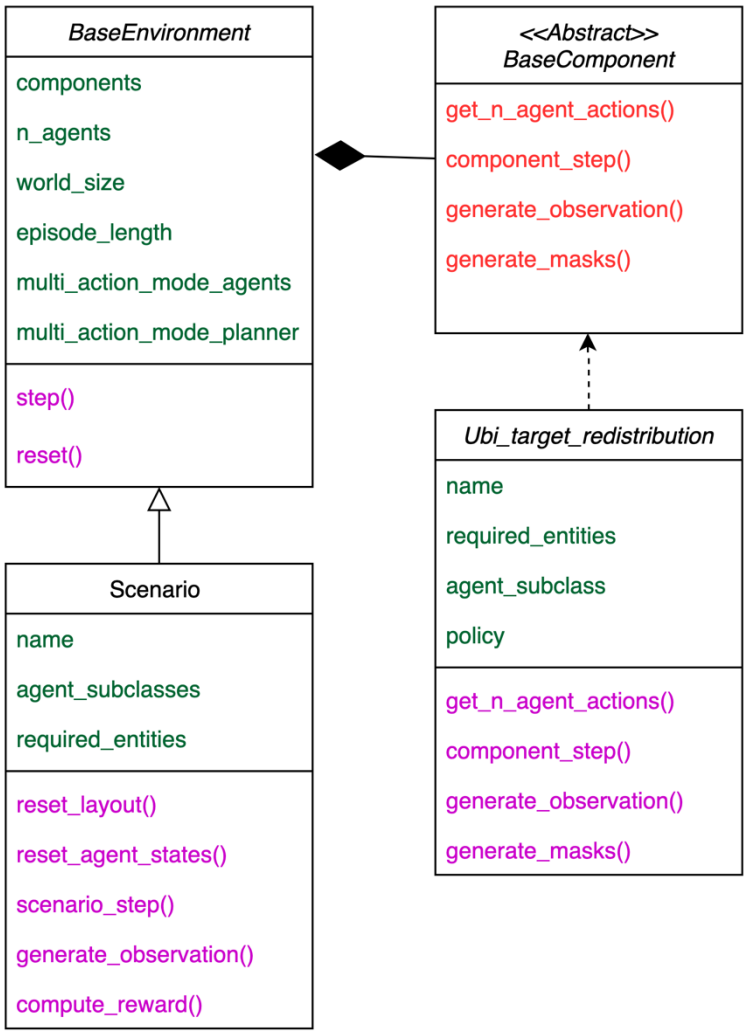

Figure 4: UML Class diagram for Simulation Environment

Figure 4 represents how redistribution policy class *Ubi_target_redistribution* implemented *BaseComponent* abstract class. The *BaseComponent* and *BaseEnvironment* have composition relationship.

The Gym-API is a standard API used to train Reinforcement learning algorithms. The *BaseEnvironment* is the base class for the simulation environment, and it implements Gym-API functions such as.

- *step*: The step function takes action as an input and returns the observations, reward, and other additional information.
- *reset*: This function resets the environment and returns the observation.

The Scenario is one of the important class in the simulation environment, and it inherits the *BaseEnvironment* class (Figure 4). The simulation environment is an instance of Scenario class.

The Simulation class controls the economic simulation coordination. The simulation class is responsible for making transition in the environment based on the agent's action. Some of the essential functions of the Scenario class were.

- *reset_layout*: This function resets the simulation environment to the initial state.
- reset_agent_states: resets agent's initial locations and inventories.
- scenario_step: This function is used to respawn wood and stone resources.
- generate_observation: This function is used to generate basic observations like the spatial layout of the environment and the agent's inventory.
- compute_reward: This function calculates the reward of the worker and planner agent.

4.1.3 Configurations of Simulation Environment

The configuration properties define the component and agent properties of the simulation environment. The configuration properties that were used in this report are mentioned below.

- A 25 X 25 two-dimensional layout was selected for the simulation environment. The water resource was used to divide the layout into four quadrants (Figure 2). A small

space is left between the water resource, which allows the agent to move from one quadrant to another. The first quadrant has both wood and stone resources. The second quadrant has no resources. The third quadrant has only wood resources, whereas the fourth quadrant has only stone resources. The layout is designed in a way to reflect heterogeneity in the availability of resources to the agent.

- Four worker agents were selected for the simulation to find the behaviour of agents in 4 quadrants with different resources.
- Initially, all agents are provided with ten coins.
- The multi-action properties were enabled for planner and worker agents. Hence, the agent can take multiple actions in a single step. For instance, the worker agent can move right and buy a wood resource from other agents in a single step.
- The agent had to do one unit of labour to build a house. The agents would receive at least ten coins as a payment for building a house. The skills of the agents were distributed based on Pareto distribution. The Pareto distribution is commonly regarded as the 80-20 rule (i.e. 80% of the total wealth was with 20% of the people).
    - The cumulative distribution function (CDF) of Pareto distribution is

$$x = x_m\left(\frac{1}{1-u}\right)^{\frac{1}{\alpha}}$$

      The value of $\alpha = 4$, m= 1 and the value of u is randomly assigned so that 80% of agents have little variation in skill. For example, the Pareto distribution for four agents can be 16, 13, 11 and 22. The difference between the first three values is less when compared to the last value 22.

- The agent must spend one unit of labour for moving and gathering resources. The skill for gathering resources is allocated based on Pareto distribution.
- The conditions for trading resources were:
    - The maximum bid an agent can offer for one resource unit is ten coins.
    - The labour cost for asking a bid is 0.1
    - The maximum trade an agent can do in a single step is four.
    - The request for a resource placed by an agent will be revoked after 30 timesteps.

4.2 Redistribution policy Integration

The Component class can be used to add additional functionality to the simulation environment. The *Build*, *Gather*, and *Trade* were the basic components that were used in this simulation. The components can be inserted or removed based on our requirements. Every Component classes have to inherit the *BaseComponent* class. The *BaseComponent* class defines the abstract methods which have to be implemented in the Component class.

The abstract methods of the *BaseComponent* class were:

- *get_n_agent_actions* : The number of actions that an agent can take using the component was given by this function.
- *component_step* : This function is triggered whenever the environment transitions to the next time step.
- *generate_observation* : Observations that were related to the component were returned. For instance, the trade component will generate observations like the number of transactions done, the agents involved in the trade, and the resources exchanged. The observations from each component will be collected by the step function of the *BaseEnvironment* class.
- *generate_masks*: This function will return a binary array consisting of an agent's invalid or valid actions. The value 0 represents an invalid action, and the value 1 represents a valid action that an agent can take.

The Universal basic Income and Conditional Cash Transfer policy functionality were added to the simulation using *ubi_target_redistribution* class. The Figure 4 represents *ubi_target_redistribution* class relationship with *BaseComponent* class.

The basic functionalities of redistribution policies were described below.

- Collection of tax
- Redistributing the funds

Band type	Taxable income (coin) thresholds	Tax rate	Tax rate range
Starting rate	0 - 5	Tax_rate_1	0% to 100%
Basic rate	6 - 20	Tax_rate_2	0% to 100%
Higher rate	21 - 50	Tax_rate_3	0% to 100%
Additional rate	Over 51	Tax_rate_4	0% to 100%

Table 1: Tax income thresholds and tax rate range

4.2.1 Tax Collection

The tax is collected from the worker agents for every 12 timesteps of the simulation period. Each timestep is considered as a month and the tax rates were changed on yearly basis. The income tax collected from the worker agents were transferred to the government treasury. The tax rates were assigned by the planner agent. The Table 1 shows marginal tax rates based on agent's income. For instance, if the agent's income is 44 and $Tax\_rate_{1\ to\ 4}$ is (0%, 20%, 40%, 50%). The calculation below describes the steps to calculate tax income.

0% Bracket: $(5 - 0) \times 0\% = 0$

20% Bracket: $(20 - 6) \times 20\% = 2.8$

40% Bracket: $(44 - 21) \times 40\% = 9.2$

50% Bracket: not applicable

The tax amount collected from an agent is 12 coins (2.8 + 9. 2)

4.2.2 Redistributing the funds

The funds from the treasury were redistributed to agents based on UBI or CCT policy.

4.2.2.1 Universal Basic Income policy

Universal Basic Income (UBI) is an economic policy implemented by some governments like Indonesia and Alaska. According to the policy, the funds will be transferred to all people

without any condition. Therefore, implementing the UBI policy is simple. In this research, according to UBI policy, every agent will receive one coin at every time step.

4.2.2.2 Conditional Cash Transfer policy

Instead of giving fund to all agents the conditional Cash Transfer (CCT) policy will provide fund to low-income agents. The CCT policy use Proxy Means Test (PMT) to determine the wealth of an agent.

Proxy Means Test is implemented in many social welfare programs to identify poor households. The variables that were correlated to the income of the agent were selected as proxy variables. The proxy variables were used to calculate the wealth of an agent. If the agent decides to trade resources, then it will transfer those resources to the Escrow account. An escrow account is managed by a third party in which resources are held on behalf of two agents engaged in trading. The agent has no control to use the resources in the escrow account other than trading. Therefore, the resources in the escrow account and coins earned by the agent were considered the potential wealth of an agent.

$$agents\ wealth\ (PMT) = income + \beta * escrow\ resources$$

If the wealth of an agent is above the threshold value, then the agent will not receive the fund. The amount of funds to an agent will also depend upon the number of low-income agents in the current state. The CCT fund is inversely proportional to the number of low-income agents.

$$CCT\ fund\ per\ agent_t = \frac{total\ fund_t}{no\ of\ low\ income\ agents_t}$$

Where t is the current timestep.

Tax band type	Planner action values
Starting rate	[0%, 10%, 20%, 30%, 40%, 50%, 60%, 70%, 80%, 90%, 100%]
Basic rate	[0%, 10%, 20%, 30%, 40%, 50%, 60%, 70%, 80%, 90%, 100%]
Higher rate	[0%, 10%, 20%, 30%, 40%, 50%, 60%, 70%, 80%, 90%, 100%]
Additional rate	[0%, 10%, 20%, 30%, 40%, 50%, 60%, 70%, 80%, 90%, 100%]

Table 2: Planner agent actions

4.2.3 Planner actions

The planner agent can do four different actions at a single timestep. Each planner action will assign the tax rate for different tax band types. Table 2 represents the available action values that a planner agent can take for each tax band type.

Action mask generation (previous year tax rate is 0.6)	
Action values	[0%, 10%, 20%, 30%, 40%, 50%, 60%, 70%, 80%, 90%, 100%]
Treasury <= 0	
Action mask	[0, 0, 0, 0 0, 0, 0, 1, 1, 1, 1]
Treasury > sufficient fund	
Action mask	[1, 1, 1, 1 1, 1, 0, 0, 0, 0, 0]
0 < Treasury < sufficient fund	
Action mask	[1, 1, 1, 1 1, 1, 1, 1, 1, 1, 1]

Table 3: Planner agent action mask based on treasury value

4.2.3.1 Action mask for tax rate

Action masking is used in reinforcement learning to restrict the agents from taking invalid actions. The action masking for the planner agent is used to guide the agent to learn optimal tax policies. If there is more treasury amount for funding redistribution policies, then the planner agent is restricted to choose a higher tax rate than the previous tax rate because the agent has more funds, and it is not necessary to tax more income from the agent. In contrast, If the government is in debt, then lower tax rate actions are masked so that the planner agent can choose only a higher tax rate than the previous tax rate. The Table 3 shows the action mask values based on the treasury amount and the previous year tax rate.

Observations	
Worker agent	current period, current income, tax rate, tax amount paid, pre-tax income
Planner agent	current period, tax period, treasury amount, worker agents income, worker agents tax amount, worker agents tax rate, pre-tax income of worker agents.

Table 4: Redistribution component observations for worker and planner agent

The Table 4 shows the observation information that were returned by the *ubi_target_redistribution* component class for every environment transition. The redistribution policy functionalities were implemented in *ubi_target_redistribution* class. According to the *policy* parameter (Figure 4) the *ubi_target_redistribution* component will behave either as a UBI or CCT policy.

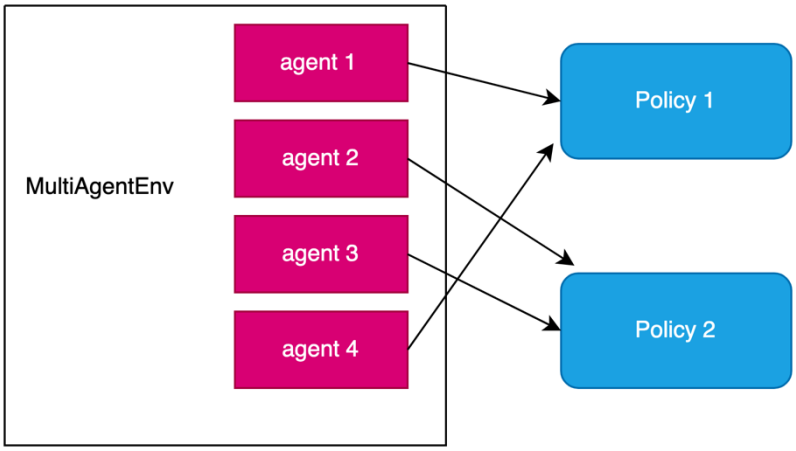

Figure 5: Multi-agent functionality

The above figure depicts the *MultiAgentEnv* wrapper which offers functionality to train multiple agents based on different policy.

4.3 Multi-agent Wrapper

In a Multi-agent environment, multiple agents will act simultaneously by sharing the same environment. In this research, worker and planner agents must be trained on different objectives. Rllib is a Reinforcement Learning library that offers more functionality to train a Reinforcement learning algorithm in a distributed framework. The Rllib framework offers a *MultiAgentEnv* abstract class, which facilitates training multiple agents using different policies. The *EconomicWrapper* class inherits the *MultiAgentEnv* class, and the abstract methods of *MultiAgentEnv* were implemented in the *EconomicWrapper* class (Figure 6).

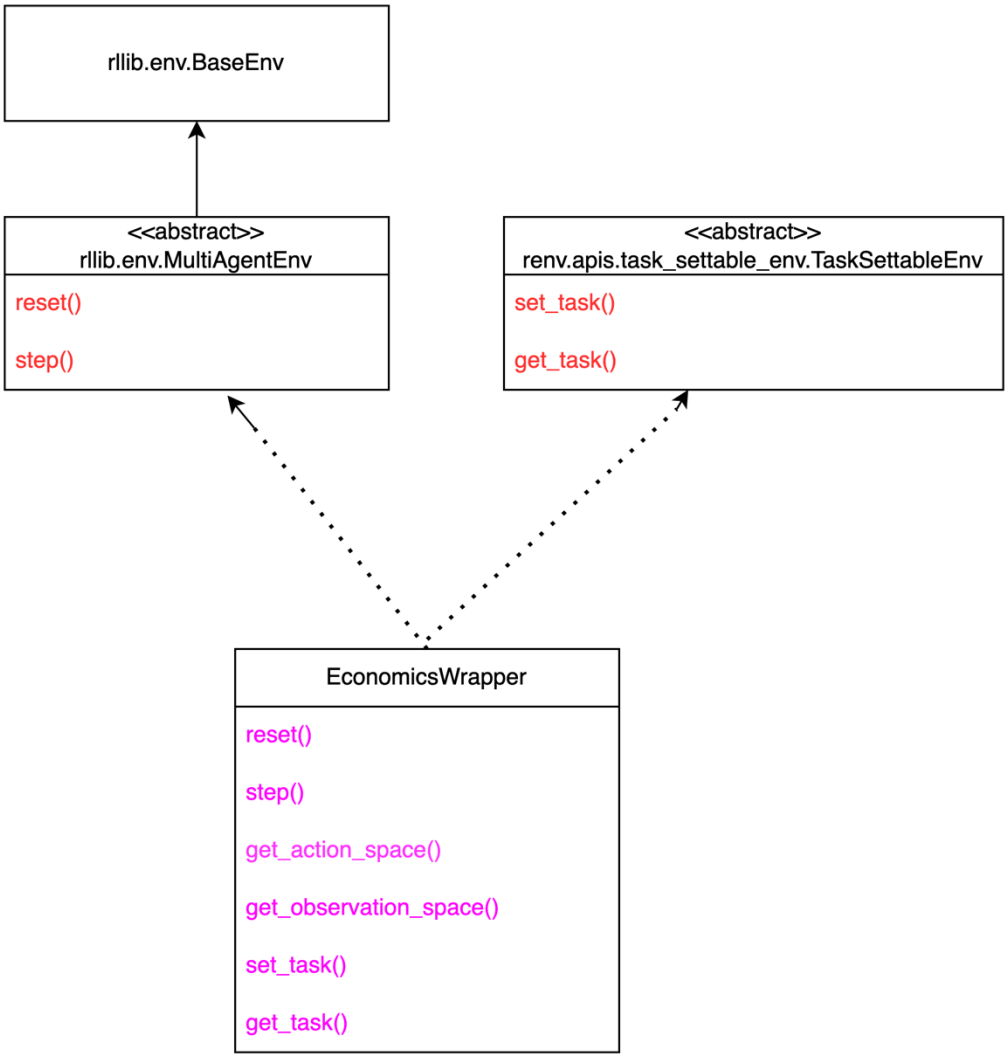

Figure 6: Multiagent wrapper

Figure 6 illustrates that EconomicsWrapper class inherits MultiAgentEnv and TaskSettableEnv class.

The abstract methods that were implemented in the *EconomicWrapper* class were:

- *reset*: This function will reset the environment and returns the observation and other information about the environment. The observation should be a dictionary object. The keys of the dictionary should be the agent IDs, and the values should be agent observations.
- *step*: This function will take action of agents as input and returns observations, rewards, agent termination information, and other information about the environment. Similar to the reset function, all the returned objects should be a dictionary with agent IDs as keys.

Other important functions in *EconomicWrapper* other than abstract methods of *MultiAgentEnv* class were:

- *get_action_space*: The function returns the set of valid actions that an agent can take.
- *get_observation_space*: This function returns the dimension of the observation values. The observation space also contains the information of minimum and maximum value of an observation.

4.3.1 Action Space

The two properties of the action space were completeness and validity. Completeness states that an agent can achieve the goal by following a set of defined actions. If the agent has no knowledge about the available actions, it will be difficult for the agent to achieve the task. The validity property ensures that all the actions are valid.

4.3.2 Observation Space

Defining observation space before training the reinforcement algorithm is important because the observation space defines the dimension of the observation, which will be used to define the structure of Neural Network models. The Neural Network models were used to train the policy of reinforcement algorithms.

4.4 Proximal Policy Optimization (PPO) model implementation

The Proximal Policy Optimization (PPO) is a reinforcement algorithm used in this research to train the agents policy. The PPO is an on-policy algorithm which means the algorithm updates its policy while interacting with the simulation environment. The algorithms have less variance, stable and easy to implement. The PPO algorithm able to handle continuous and multi-dimensional observation and action spaces. The PPO uses actor-critic approach and clipped surrogative objective to optimize its policy.

4.4.1 Actor-Critic

There were two main components in the PPO model which were Actor and Critic (Figure 7).

- The Actor will learn a stochastic policy instead of a deterministic policy. This helps the agent to take random action and explore the environment instead of choosing deterministic action every time. The Actor takes the current state (i.e. observation) as an input and uses a neural network model to predict the probability of actions $\pi_\theta(a|s)$ for the given state (s).
- The critic will evaluate whether the action taken by the actor leads to better reward. The actor will use the suggestion from the critic to optimize its policy. The Critic uses neural network model to predict the cumulative reward for a given state $\check{V}_\emptyset^\pi(s)$ it is also called as state value. The below diagram represents how actor and critic model interact with each other.

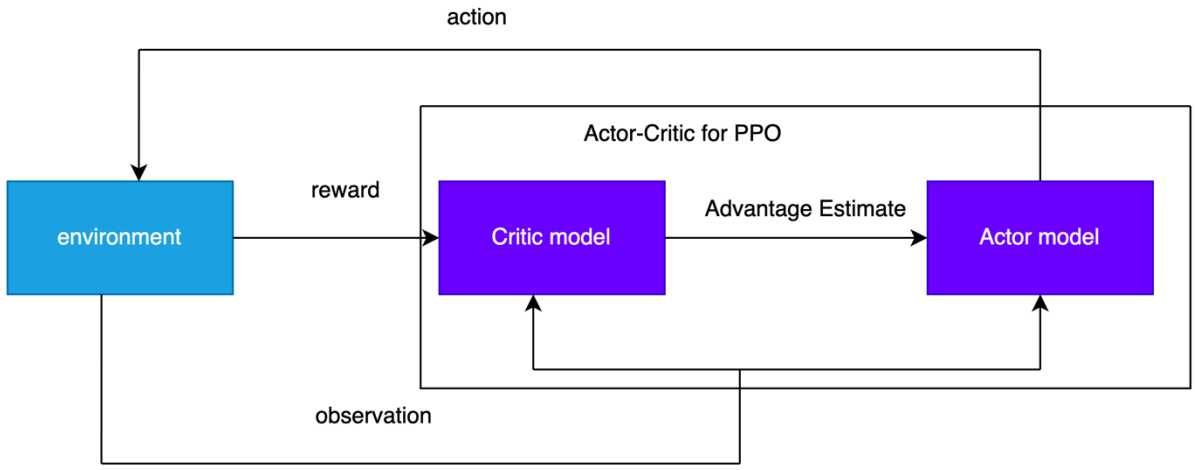

Figure 7: Agent Actor-Critic model

Advantage Actor-Critic (A2C) algorithm uses the actor-critic approach to update its policy, but it faces some drawbacks like slow convergence, high variance and sensitive to hyperparameter changes. Therefore, the A2C policy is sensitive to recent update which will make the policy to deviate more from the old policy learned from past experience. Hence it leads to unstable learning to remedy this drawback the PPO algorithm uses clipped surrogate objective $L^{Clip}(\theta)$ with actor-critic technique which restricts the PPO policy to make large update when compared to old policy. This approach leads to stable learning.

$$L^{Clip}(\theta) = \widehat{\mathbb{E}_t}[\min(r_t(\theta)\widehat{A_t}, clip(r_t(\theta), 1-\epsilon, 1+\epsilon)\widehat{A_t})]$$

$$r_t(\theta) = \frac{\pi_\theta(a_t|s_t)}{\pi_{\theta_{old}}(a_t|s_t)}$$

$\pi_\theta(a_t|s_t)$ is the new policy and $\pi_{\theta_{old}}(a_t|s_t)$ is old policy. The probability ratio of new policy and old policy is used to evaluate how much the new policy is deviated from old policy.

- If $r_t(\theta) > 1$ then the current policy is deviated more from the old policy.
- if $0 < r_t(\theta) < 1$ then current policy is more likely the same compared to old policy.

The Advantage $\widehat{A_t}$ is a numerical value used to estimate how better the current action when compared to the average action in a particular state s_t. If the advantage value is positive for the given state, then current action will yield better results than the average action for the given state. If the advantage value is negative, then the current action performance is suboptimal when compared to the average action for the given observation.

The $clip(r_t(\theta), 1-\epsilon, 1+\epsilon)$ function helps to clip the probability ratio $r_t(\theta)$ and restrict the $r_t(\theta)$ value to move away from the interval range $[1-\epsilon, 1+\epsilon]$.

4.4.1.1 Deep Neural Network Model

In this research the actor and critic components of PPO algorithm were trained using Neural Network model. The current observation of the environment act as an input for both actor and critic model.

The observation returned by the environment has both spatial information and non-spatial information. For instance, the location of the agent and resources were spatial information. In contrast, the number of coins earned by the agent and the resources collected by the agent were non-spatial information.

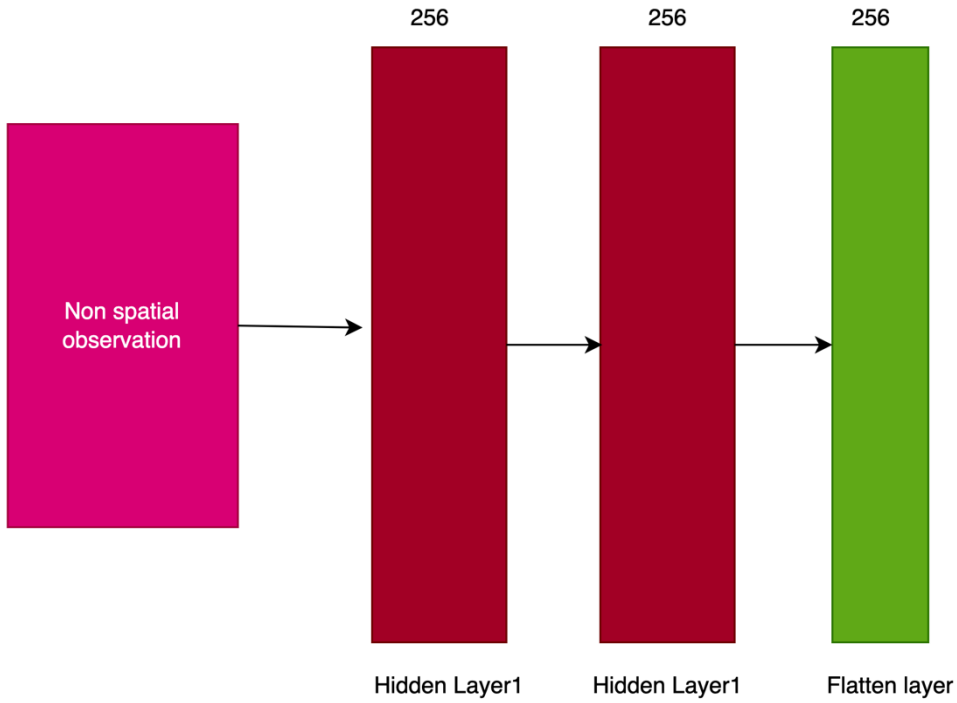

Figure 8: Deep Neural Network Model for Non-spatial observation

4.4.1.1.1 Non spatial model

The Deep Neural Network (DNN) was a non-linear model that could learn abstract and complex features. The Neural Network models also provide an opportunity to combine the Neural Network of different architectures like DNN and Long Short-Term Memory (LSTM) model, which can be used to learn both complex features from the DNN model and temporal features from the LSTM model. In this research Non-spatial model was created to find the complex features based on non-spatial input. Two non-spatial models will be created for both Actor and Critic Components. The Non-Spatial model consists of two fully connected hidden layers (Figure 8). The Non-spatial model last layer is a flattened layer which will be used as an input for another DNN which will be discussed in section 4.4.1.2.

The activation functions were used to calculate the output of the neuron within a certain range, which helps the neural network model to learn complex relationships in the data. The Hyperbolic Tangent (Tanh) activation function is symmetric with respect to the origin. The Tanh function is better than the sigmoid function for input with negative values because the output range of the Tanh function is [-1, 1], but for the sigmoid function, the output range is

[0, 1]. Hence, the output for the sigmoid function will be zero for negative input, but for the Tanh function, it will be close to -1. Therefore, the Tanh function can capture both positive and negative relationships in the data. The agent observation data can contain negative values. Therefore, the Tanh activation function is used for the non-spatial DNN model.

$$\tanh(x) = \frac{2}{1+e^{-2x}} - 1$$

4.4.1.1.2 Spatial model

Convolutional Neural Network (CNN) is used to train spatial observational data because CNN can capture the patterns in the spatial data, and it is highly scalable. Each layer in the CNN network can capture different features of complex spatial structures.

The two-dimensional simulation environment contains many spatial information like the agent's current location, path followed by the agent, location of wood, stone, water, house, trade, and initial location of the agents. Therefore, there are 8 kinds of spatial information that were returned as an observation from the environment. The dimension of the environment is 25×25 so the total dimension including all spatial information will be $25 \times 25 \times 8$.

The Convolutional filter is used to capture complex patterns in the data. Each filter has a height, width and depth. The height and width depend upon the filer size, whereas the depth of the filter will depend upon the dimension of the input data. The values in the filter and the input were multiplied element-wise, and then a single numerical value was obtained by adding all the values. The value obtained will be placed in a seperate feature map. The feature maps collected after applying the filter will form the next layer of CCN.

Multiple filters will help the CNN layer to capture different features. If the size of filter is gradually increased, then lower layers would capture small features using that information the higher layers can capture high-level features.

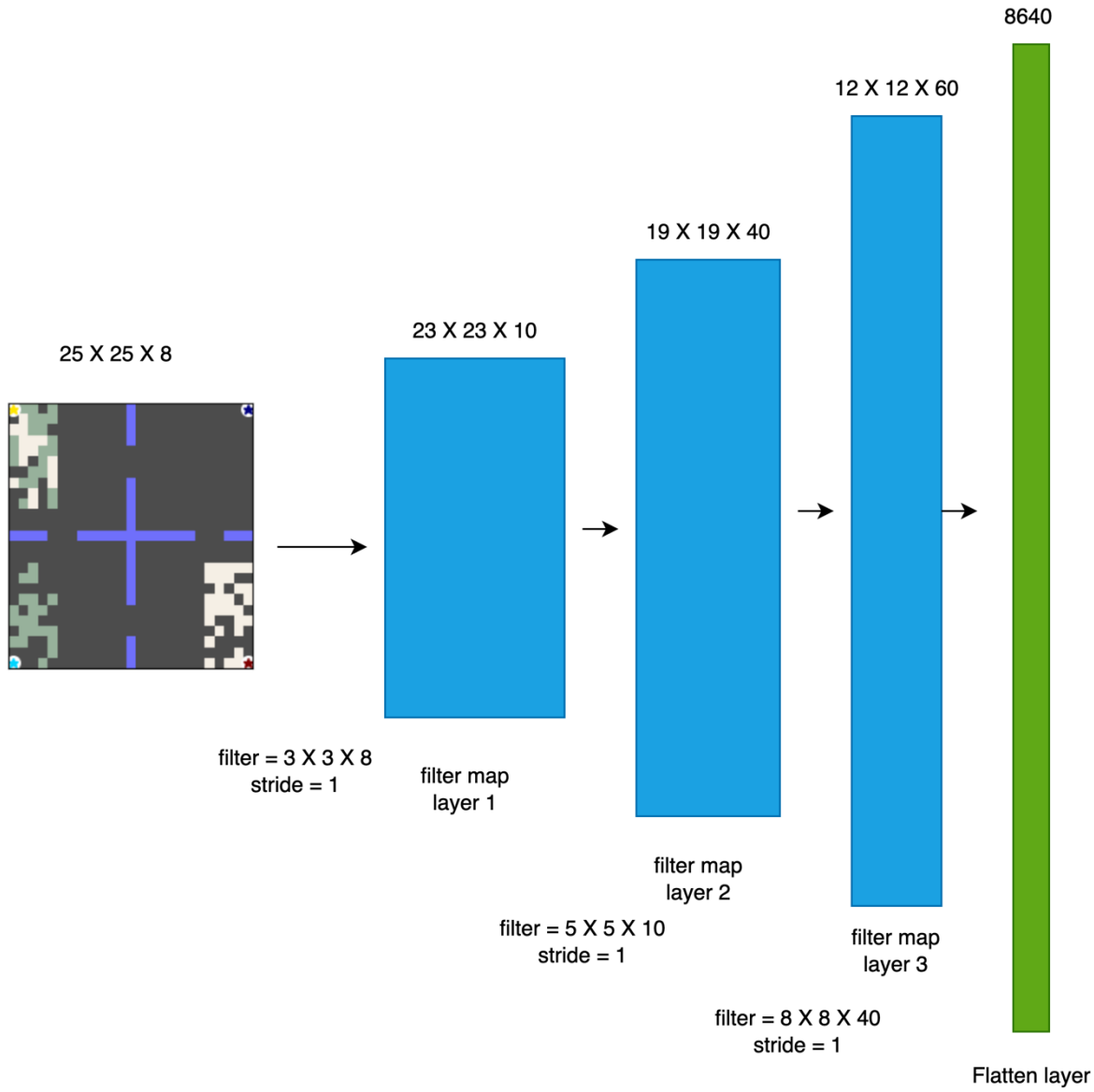

Figure 9: Convolutional Neural Network Model for spatial observation

The stride determines how filter should move horizontally and vertically in the input data. For instance, if the value of the stride is 1 then the filter will move one pixel at a time. The below calculation defines the number of filter and its dimension in three feature map layers that were used in this project.

Feature map layer 1:

The filter of size $3 \times 3 \times 8$ is applied to the input data of dimension $25 \times 25 \times 8$ with stride value 1.

Input dimension $(h \times w \times d) = 25 \times 25 \times 8$

Filter size $(f_h \times f_w \times d) = 3 \times 3 \times 8$

No of filters = 10

Output dimension $(M_h \times M_w \times M_d) = \frac{(h - f_h)}{stride} + 1 \times \frac{(w - f_w)}{stride} + 1 \times no\ of\ filters$

$$= \frac{(25 - 3)}{1} + 1 \times \frac{(25 - 3)}{1} + 1 \times 10$$

$$= 23 \times 23 \times 10$$

Feature map layer 2:

The filter of size $5 \times 5 \times 10$ is applied to the feature map layer 1 of dimension $23 \times 23 \times 10$ with stride value 1.

Input dimension $(h \times w \times d) = 23 \times 23 \times 10$

Filter size $(f_h \times f_w \times d) = 5 \times 5 \times 10$

No of filters = 40

Output dimension $(M_h \times M_w \times M_d) = \frac{(h - f_h)}{stride} + 1 \times \frac{(w - f_w)}{stride} + 1 \times no\ of\ filters$

$$= \frac{(23 - 5)}{1} + 1 \times \frac{(23 - 5)}{1} + 1 \times 40$$

$$= 19 \times 19 \times 40$$

Feature map layer 2:

The filter of size $8 \times 8 \times 40$ is applied to the feature map layer 2 of dimension $19 \times 19 \times 40$ with stride value 1.

Input dimension $(h \times w \times d) = 19 \times 19 \times 40$

Filter size $(f_h \times f_w \times d) = 8 \times 8 \times 40$

No of filters = 60

Output dimension $(M_h \times M_w \times M_d) = \frac{(h - f_h)}{stride} + 1 \times \frac{(w - f_w)}{stride} + 1 \times no\ of\ filters$

$$= \frac{(19 - 8)}{1} + 1 \times \frac{(19 - 8)}{1} + 1 \times 60$$

$$= 12 \times 12 \times 60$$

The feature map layer 3 is then flattened and will be used as input for another DNN model which will be discussed in section 4.4.1.2.

The Rectified Linear Unit (ReLU) activation function is used for the Convolutional Neural Network, which helps the network to learn non-linear relationships in the data. During backpropagation using the ReLU activation function, the derivative for negative input is 0 and the derivative for positive input is 1. Therefore, computing gradient using ReLU activation function is simple, stable, and converges faster compared to other activation function.

$$R(x) = \max(0, x)$$

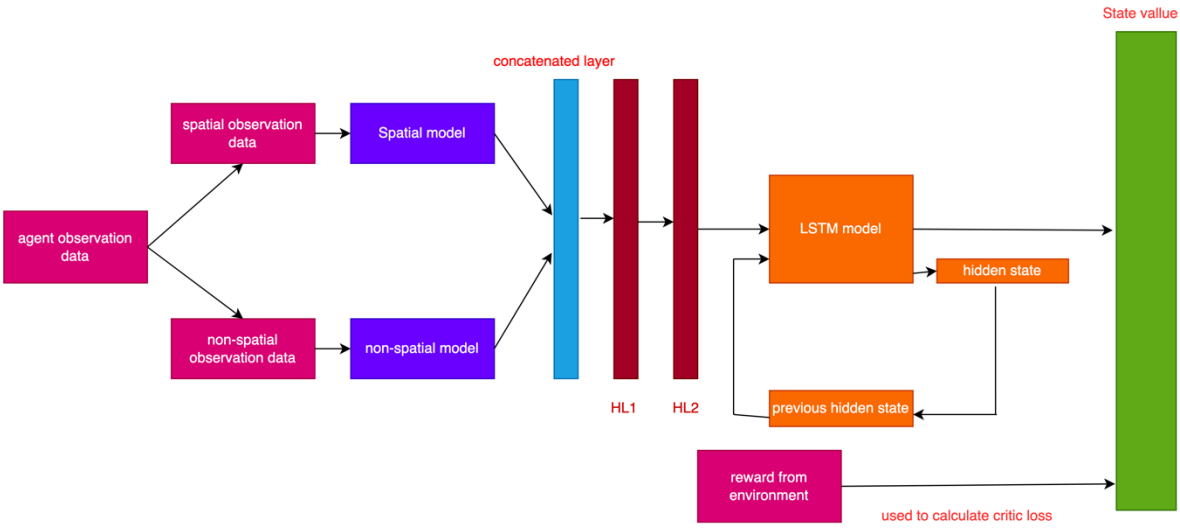

Figure 10: Critic Model

The Figure 10 depicts the critic model. The critic model process spatial and non-spatial observation using DNN and CNN models and the output of the two models were concatenated and it is used an input for DNN which contains two hidden layers. The hidden layers then connected to LSTM model. The output of the LSTM model is the state value. The reward from the environment used to calculate critic loss function.

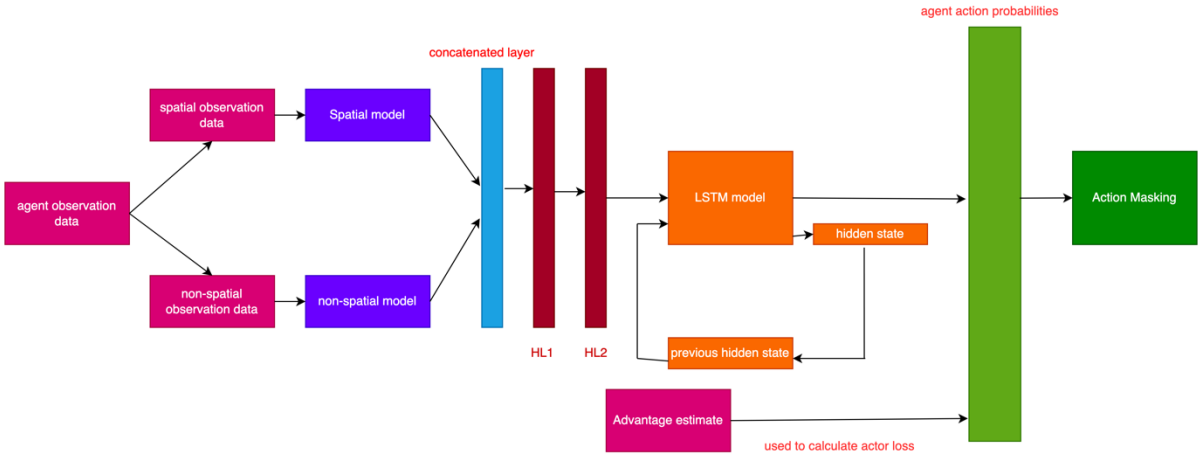

Figure 11: Actor Model

The figure 11 depicts actor model which has same connection as critic model but the output of the actor model is agent action probabilities. The advantage estimate is used to calculate actor loss function.

4.4.1.2 LSTM model

The simulation environment is a Markov process. Once the state transitions to the next state, the environment will forget the previous state. The previous observations present valuable information and will be helpful for an agent to take future actions. Therefore, the Long Short-Term Memory (LSTM) model is used with the Deep Learning model to learn the PPO policy efficiently. The LSTM is a Recurrent Neural Network model that can remember important long-term and short-term sequences. The critical component of the LSTM model is the memory block (i.e. cell). The number of memory blocks represents the number of valuable short-term and long-term information that an LSTM model can hold. The cell values can be updated and deleted by the LSTM model. The sequence length determines the number of transition steps that an agent can remember. In this research, LSTM memory block size 20 is used. Therefore, the LSTM model can store some of the most important 20 information related to short-term

and long-term sequences. The sequence length is defined as 10 because the previous ten memory sequence is more than enough for an agent to make a good decision, and training the model would not be computationally expensive.

The flattened layers from the spatial and non-spatial models were concatenated and used as input for another DNN with two hidden layers. This combination can result in a more complete and richer representation of the data and facilitate capturing complex linkages and patterns in both spatial and non-spatial data. The hidden layers were then connected to the LSTM model (Figure 11,12). The output of the LSTM model is passed as an input for output layer of agent action probabilities $\pi_\theta(a|s)$ if it is an actor model (Figure 11), whereas LSTM model will be connected to a single output neuron (Figure 10), which is used to predict the state value $\check{V}_\emptyset^\pi(s)$ for an agent policy in critic model. Therefore, two neural network models will be created for agent and critic components.

4.4.1.3 Action Masking

The worker and planner agent are restricted from taking some actions based on the current state of the environment. For instance, a worker agent cannot pass through the water in the environment. Therefore, to restrict the agent from taking invalid action, the probability of action returned by the actor model is added with the action mask array. The action mask array consists of large negative values for invalid actions and zero values for valid actions. Therefore, after the addition of an action mask with the probability of actions from the actor model, the probability of an invalid action will be very low (figure 11). Therefore, the agent will not choose invalid actions.

4.4.1.4 Training Actor and Critic model

The Neural Network model will learn efficiently if it has large amounts of labeled data but in reinforcement learning it is very hard to get labeled data. The data in reinforcement learning were collected only by interacting with the environment. Initially, Neural Network model parameters were set to zero for both actor and critic model and for a mini batch of iterations data such as state s_t, action a_t, probability of action for the given state $\pi_\theta(a|s)$, state value $\check{V}_\theta^\pi(s)$, and rewards r_t were collected. The collected data were used to train actor and critic

model by using actor and critic loss functions. The trained actor and critic model is used to predict value of $\pi_\theta(a|s)$ and $\check{V}_\theta^\pi(s)$, for the next batch of observations and again the collected data from the batch will be used to train actor and critic model this process will take place iteratively. The Stochastic Gradient Descent (SGD) mini batch size of 21 is used to train actor and critic model in this research. The loss function used for actor model is clipped surrogative objective $L^{Clip}(\theta)$.

The loss function for the critic model is.

$L_{critic}(\theta) = \frac{1}{2}\mathbb{E}[(\check{V}_\theta^\pi(s) - \check{V}_t)^2]$

$\check{V}_\theta^\pi(s)$ is the predicted state value for a given state from critic model.

$\check{V}_t$ is the estimated state value calculated using the collected batch data.

Each agent policy consists of an actor and critic components. These two components will be used to train the policy for an agent.

4.5 Curriculum Learning

Training worker agents without any strategy leads to poor performance. The agents could not find a good strategy to earn more coins because agent exploration and resource collection will increase labour costs. Therefore, negative rewards will be assigned for those actions. Hence, the worker agent policy finds it challenging to learn a good policy. Therefore, curriculum learning is employed to guide the agent to concentrate more on future rewards than initial negative rewards. In curriculum learning, the smaller task is assigned first, and then the complexity of the task will be gradually increased. The curriculum learning facilitates stabilized learning and improved exploration. The *TaskSettableEnv* class from the Rllib framework is used to set curriculum tasks for an agent when training the agent policy. The *EconomicsWrapper* class inherits the *TaskSettableEnv* (Figure 6) to enable the multiagent wrapper class to set curriculum tasks when training the worker agent policy.

The abstract methods of *TaskSettableEnv* that were implemented in *EconomicsWrapper* class were:

- set_task: The set_task function will be triggered whenever next iteration of training starts for the agent policy. Based on agent's performance task will be assigned.
- get_task: This function returns the current task of the worker agents.

Incremental curriculum learning is used to train the worker agent with a sequence of tasks and increment the task complexity by monitoring the agent's performance.

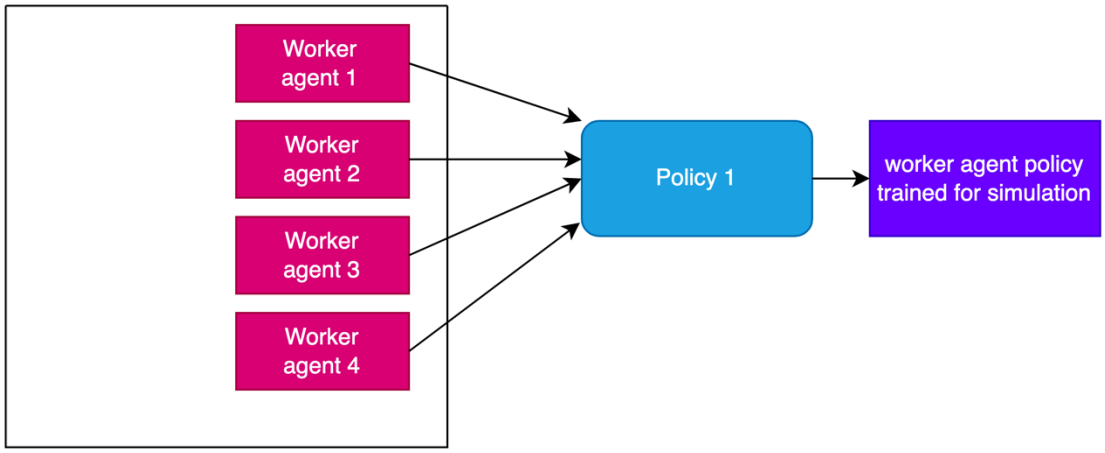

Figure 12: worker agent curriculum learning policy.

Figure 12 depicts that worker agents were trained using single policy in curriculum learning.

4.5.1 Resource Collection task

Four worker agents were used in this simulation environment. Each agent was initially assigned to four corners of the two-dimensional environment. Some of the agents have more resources under close proximity, but some agents will not have any resources nearby. The important task the agent needs to learn is that collecting resources will increase their income in future. In order to collect more resources, all agents should be placed near resources. Therefore, all agent's locations were initialized near resources randomly. The curriculum task conditions were described below.

- Task 1: The condition of task 1 is that all the agents should have at least three resources nearby. The distance between the agent and resource must be less than 3 cells in the 2D simulation. Another condition during training is that the agent should receive at least a total reward of 5.
- Task 2: The agent's location condition was similar to task 1, but the total reward condition for an agent is set to 10. For tasks 3, 4, 5, the agent's reward condition value was gradually increased.

The agent will move to the next task once they finish their previous tasks. The agent's proximity to resources will improve the probability of the agent's actions to collect more resources, and the agent will receive a positive reward by selling or building houses using the collected resources. Therefore, in this curriculum task, agents have learned how to earn income.

4.5.2 Exploration task

Exploration strategy is another most important strategy the agent must learn. The water resource is one of the main obstacles for an agent to explore the environment because the agent cannot pass through the water resource. In order to train the agent to explore the environment, agent locations were not initialized near resources. Instead, the locations of the agent were set to four corners of the environment. Therefore, the agent can collect resources only by exploration. The reward condition for the agent is gradually increased after each task.

During curriculum learning, the planner agent actions were removed, and their rewards were set to zero. Therefore, only the worker agent's rewards were taken into consideration when learning the policy. The four worker agents were trained using the same policy (Figure 12), which will help the agents to share their knowledge and experience. For instance, if agent 1 learned how to explore the resources in the environment, that knowledge can be shared with agent 2. The agent 2 can use the shared knowledge to make better decisions in the future. The worker agent is trained using the PPO algorithm. The parameters of the curriculum learning policy of the PPO algorithm were not selected by hyperparameter tuning because the agents were randomly initialized. Therefore, performance found using tuning will be biased. Hence, the parameters that were used to give the best results in many reinforcement learning problems were used in the Curriculum learning policy. The worker agent policy trained using curriculum learning will be used for final simulation when comparing UBI and CCT economic policy. The table represents the parameters used in curriculum learning.

parameters	value
Clip (ϵ)	0.3
SGD mini batch size	21
Learning rate	5×10^{-5}

Table 5: Curriculum learning parameters

Table 5 represents the parameter values that were used in curriculum learning. The clip parameter is used to restrict large policy update. SGD mini batch parameter is the number of episode data which is used to train actor and critic model for each batch. Learning rate determines scale of gradient steps in updating the policy.

4.6 Two Level Learning

The main objective of this research is to compare Universal Basic Income and conditional Cash Transfer policy by optimizing and balancing both productivity and equality. The worker agents were trained using curriculum learning to optimize productivity. The objective of the planner agent is to improve the social welfare of the worker agents by optimizing tax rates for both UBI and CCT economic policies.

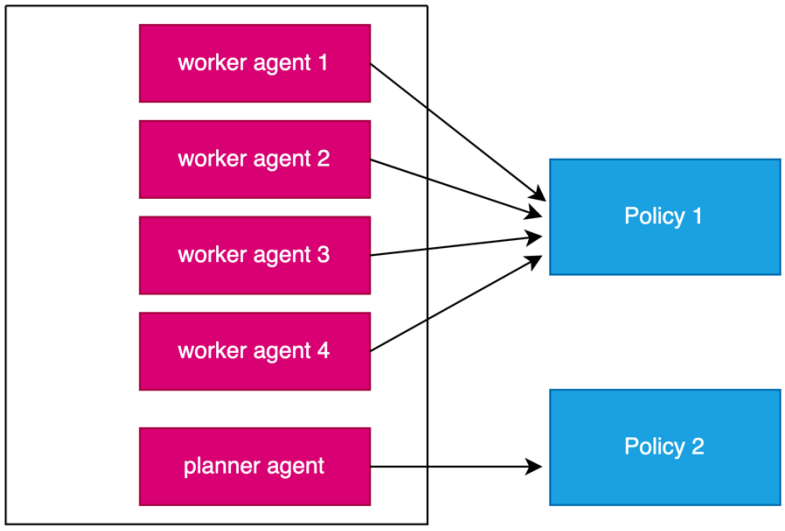

Figure 13: Two level learning

4.6.1 Adaptive Learning

The planner agent cannot be trained separately because the planner agent cannot optimize the tax rate without the worker agent's income. The planner agent will try to adapt its behaviour based on the actions of the worker agent. The reward of the planner agent is $productivity \times equality$. Therefore, the planner agent reward is also based on worker agent's income (i.e. productivity). The planner agent has no prior knowledge about tax strategies it will try to learn the policy only by trial and error. The planner and worker agents were trained parallelly using separate policy (Figure 13).

The first level of learning is the policy learned by the worker agent because it concentrates only on productivity (Figure 12). The policy learned by the planner agent is considered as a second level of learning because planner actions will depend upon the worker agent (Figure 13). The model for UBI and CCT were trained separately based on UBI and CCT functionality in the simulation environment (Figure 14). The planner agent trained using UBI and CCT functionality were used for final simulation to compare two redistribution policies.

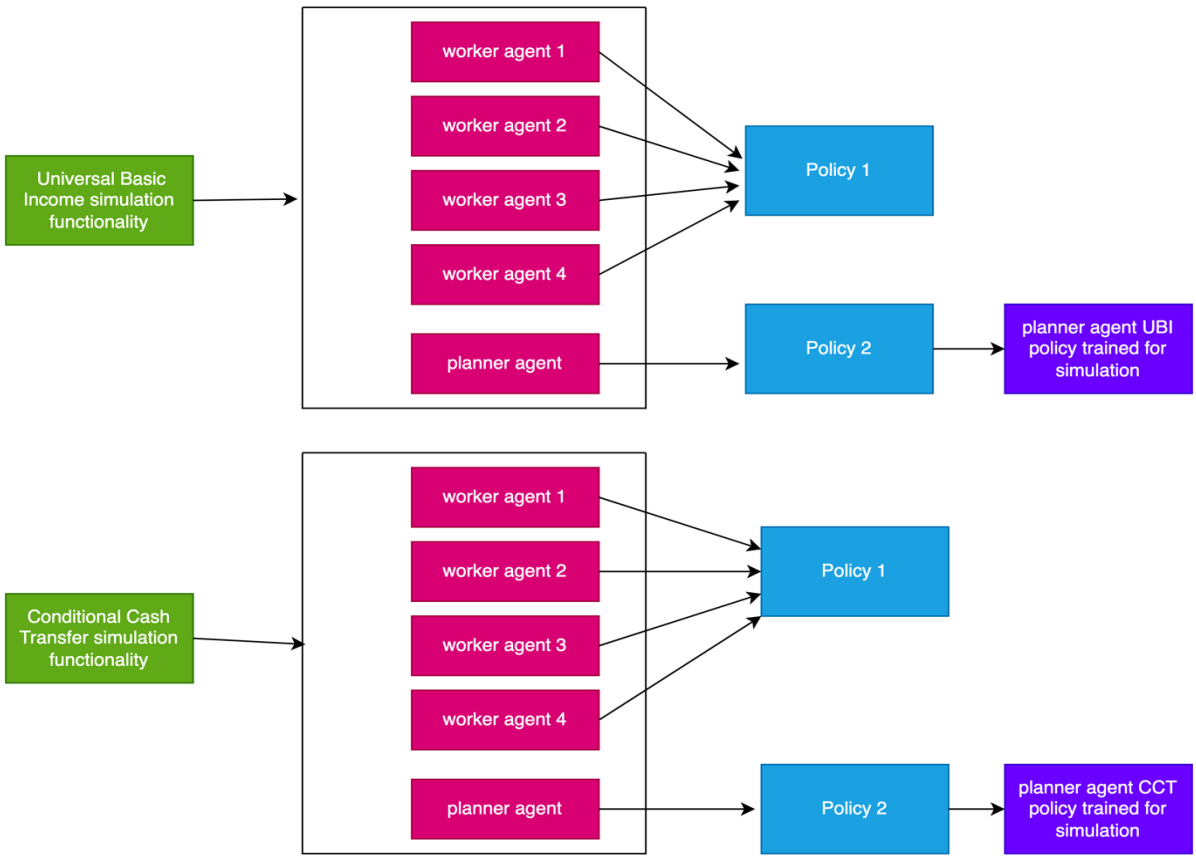

Figure 14: UBI and CCT social planner policy

4.7 Model Evaluation

Reinforcement learning modal evaluation is important to track the performance of the modal for a given objective.

4.7.1 Hyperparameter Tuning

The hyperparameter tuning technique is employed to find the best parameters for training batch size, learning rate and discount factor for the PPO algorithm. In this research, a randomized search strategy is used for hyperparameter tuning because, given the complexity of the neural network models, technique like grid search is computationally expensive.

The performance of the hyperparameters were assessed based on the evaluation metric. The average reward mean is the evaluation metric that was used to tune hyperparameters.

$$average\ reward\ mean = \frac{Total\ rewards}{no\ of\ episodes}$$

4.7.2 Entropy

Entropy is the measure of randomness or uncertainty in the agent policy. The exploration strategy of an agent can be determined by entropy value. The agent tends to explore more if its entropy value is higher. If the entropy value is low, the agent will exploit the learned knowledge more instead of exploration.

$$H\big(\pi(a|s)\big) = -\sum_a \pi(a|s)\,.\log\left(\pi(a|s)\right)$$

$H\big(\pi(a|s)\big)$ is the entropy for a policy π and state s.

$\pi(a|s)$ is the probability of agent actions in a state s.

$$\frac{\pi_\theta(a_t|s_t)}{\pi_{\theta_{old}}(a_t|s_t)}$$

4.7.3 KL Divergence

Kullback-Leibler (KL) Divergence is used to measure difference between two probability distribution. In this research KL divergence is used to measure difference between new policy and old policy of an agent. If the value of the KL divergence value is low, it indicates agent policy is making very small updates to the policy. If the KL divergence value is low, then it means the policy is stable.

$$D_{Kl}\left(\pi_\theta(a|s) \mid \pi_{\theta_{old}}(a|s)\right) = \sum_s \pi_\theta(a|s) \log\left(\frac{\pi_\theta(a|s)}{\pi_{\theta_{old}}(a|s)}\right)$$

$\pi_\theta(a|s)$ probability of agent actions based on current policy.

$\pi_{\theta_{old}}(a|s)$ probability of agent actions based on old policy.

4.7.4 Learning curve

The learning curve evaluates the agent's performance after each training iteration. If the learning curve shows an upward trend, then it means the agent can improve its performance after each training. If the learning curve shows a downward trend, then there is a problem with learning strategy changes like improving exploration strategies could be tried to improve the performance of the agent. The episode reward mean is used to evaluate the performance of the policy after each iteration.

The simulations of UBI and CCT were conducted using the trained agents and the ensuing simulation outcomes were subjected to comparative analysis and thorough examination.

5 Results

The simulated results of Universal basic Income and Conditional Cash Transfer policy and the reinforcement learning modal performance were discussed below.

5.1 Curriculum Learning results

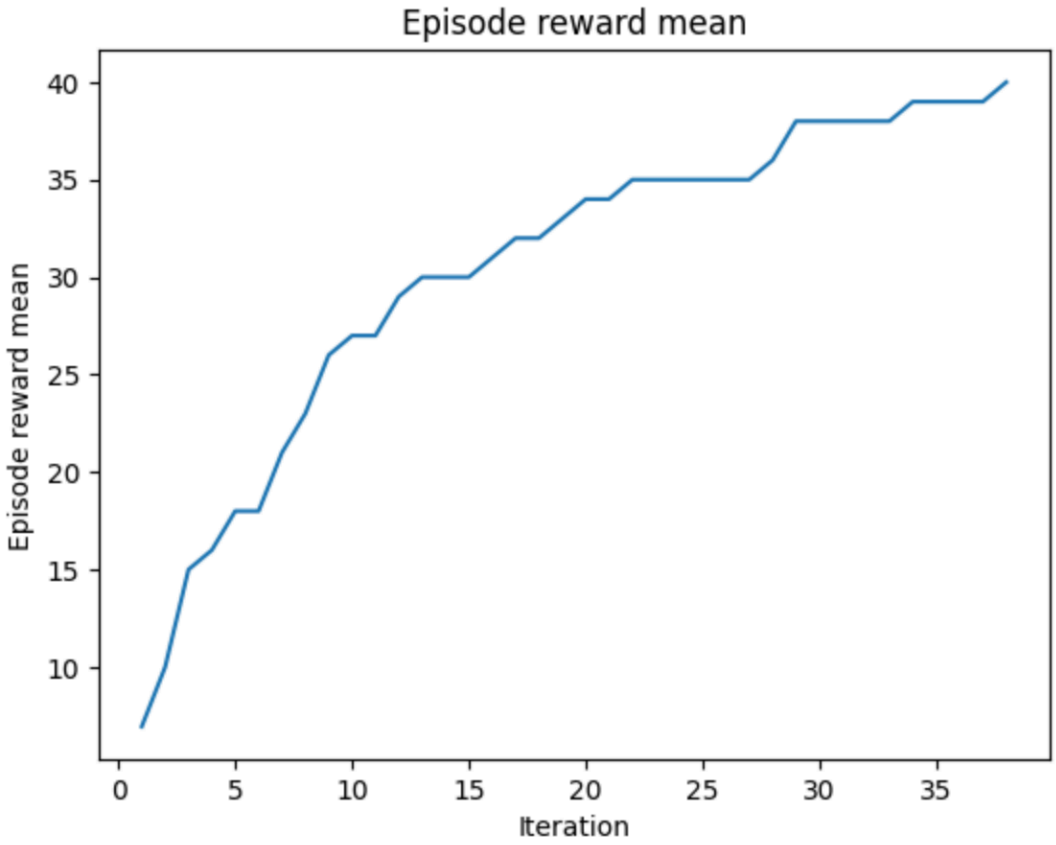

Figure 15: worker agent learning curve.

The above chart shows the performance of the agent is steadily increasing and there is no fluctuation. After the 30th iteration, there is no significant increase in performance because the agent reached its optimal policy at this stage. At the beginning of the learning curve, there is a rapid improvement in agent performance because the curriculum learning task guides the agent to learn basic tasks within a short period of time (Figure 15).

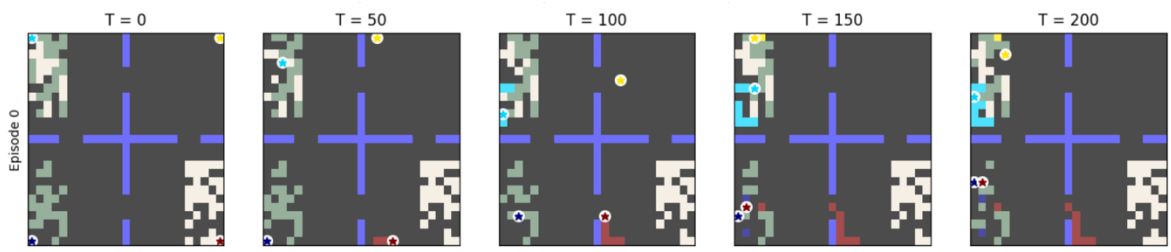

Figure 16: curriculum learning simulation.

The above simulation diagram of curriculum learning shows the performance of the agent at each 50 timesteps. The blue agent able to collect more wood and stone resources which were nearby and constructed many houses. In contrast, the yellow agent in the second quadrant has no resources. Therefore, it explores and move to the 1st quadrant to collect resources. This shows that agent learned to collect resources by using both exploration and exploitation strategies. Hence, the objective of worker agent in optimizing productivity is achieved.

metrics	**Universal Basic Income**	**Conditional Cash Transfer**
Episode reward mean	42	55

Table 6: Two level learning model reward

metrics	**Universal Basic Income**		**Conditional Cash Transfer**	
	worker policy	**planner policy**	**worker policy**	**planner policy**
Policy reward mean	5.82	18.8	7.8	24
Policy loss	-0.004	-0.03	-0.01	-0.04
Value function loss	3.15	5.4	3.5	5.5
KL divergence	0.2	0.2	0.3	0.3
Entropy	12	9.2	12	9.2

Table 7: policy results of worker and planner agent in two-level learning

5.2 Two Level Learning results

The entropy values for worker agents were high. Therefore, the worker agent explores more actions when compared to the planner agent (Table 7). The policy loss and KL divergence values were close to zero, which indicates the policies were stable. The mean worker policy reward for the CCT policy is greater than the UBI policy. However, worker policy from two-level learning is not used for the final UBI and CCT comparison analysis. The CCT planner policy performance is better than UBI planner policy (Table 7). The episode reward mean for both redistribution policies was greater than 40 (Table 6). The figure 30 shows that the percentage of equality achieved by both UBI and CCT planner agents is above 80%. Hence, the planner agent is able to optimize the social welfare of the worker agent.

5.3 Redistribution policy simulation results

The simulation results of the Universal Basic Income and Conditional Cash Transfer policies show that the productivity of the Universal Basic Income and Conditional Cash Transfer policies was similar during the initial stages (Figure 29). However, after some timesteps, the productivity of Universal Basic Income is slightly higher than Conditional Cash Transfer policy. The Free Market (i.e. tax free policy) has the highest productivity compared to UBI and CCT policies (Figure 29). This clearly shows that redistribution policies have a negative effect on productivity.

The redistribution policies improved people's income during initial stages. However, redistribution policies do not perform well in long-term in terms of productivity. According to the simulation, the social welfare of the people is improved with the help of redistribution policies (Figure 30). In CCT policy, low-income individuals started to earn more income within a short period of time. However, wealthy people started to lose their income at a faster rate in CCT when compared to UBI policy (Figure 26). The UBI policy has a low inequality percentage when compared to the CCT policy. The CCT policy promotes consumption when compared to the UBI and Free Market policy.

6 Discussion

The impact of the economic policies were systematically analyzed, interpreted and the findings were discussed briefly.

6.1 Free Market simulation

A Free Market is an economic system in which the government have little or no authority to enact taxes or regulate the market. The buyers and sellers determine the price of goods and services. In this economic simulation, the free market is used to find the equality and productivity of the worker agents without tax and redistribution of coins.

Figure 17: Free market policy simulation.

Figure 17 depicts the simulation of Free market policy for each 50 timestep interval.

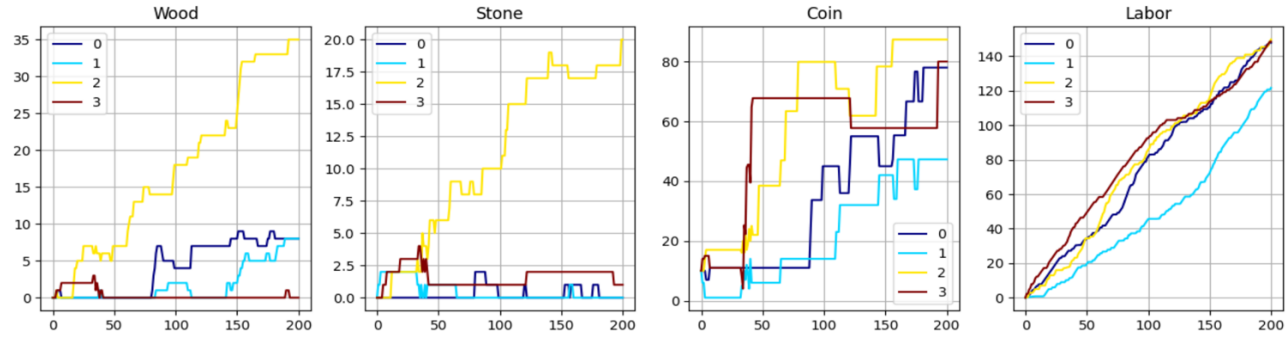

Figure 18: Agent resources and labour cost in Free Market

Figure 18 represents the wood(a), stone(b), coin(c) resources, and labour cost of agents at each time step for Free market policy.

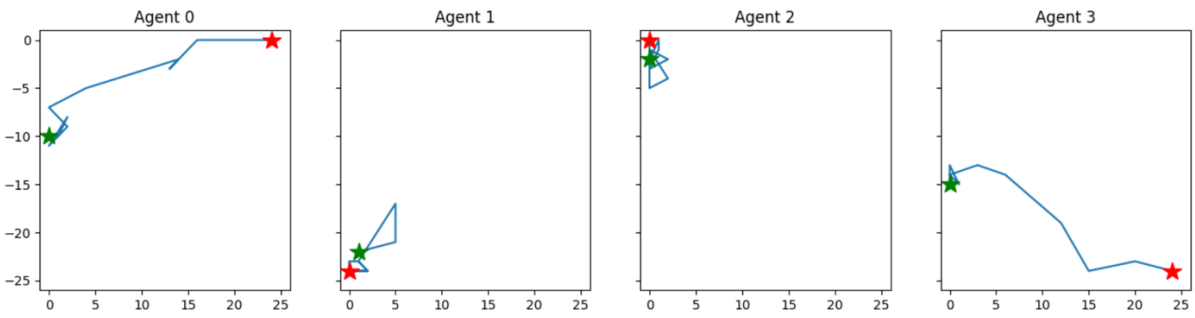

Figure 19: Agent path in Free Market

Figure 19 represents (a) agent 0, (b) agent 1, (c) agent 2, and (d) agent 3 paths taken by agents during Free Market simulation experiment. The red start indicates starting location and green star indicated location of the agent at the end of the simulation.

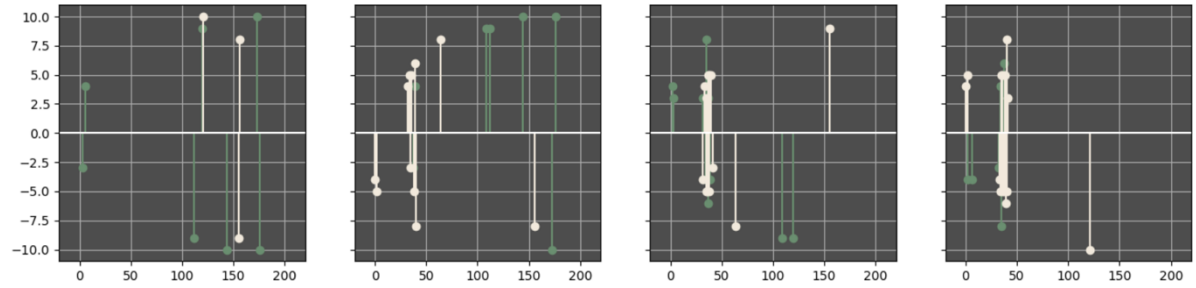

Figure 20: Free Market trading activities.

The Figure 20 represents (a) agent 0, (b) agent 1, (c) agent 2, (d) agent 3 trading activities in Free Market policy. The green and white line indicates wood and stone resources that were traded at a specific timestep.

Figure 17 represents the change in simulation dynamics at intervals of 50 timesteps. Initially, all the worker agents are in the four corners of the 2D simulation world. From Figure 17, it is obvious that the yellow agent has both wood and stone resources nearby. Therefore, the yellow agent exploited this opportunity and collected more wood and stone resources when compared to other agents. In contrast, the dark blue agent has no resources nearby, and it received no funds from the government. Therefore, the dark blue agent moved to the left quadrant and started to collect both wood and stone resources (Figure 17). The red agent collected some stone resources in the fourth quadrant and moved to the third quadrant to collect wood resources. The light blue agent did not do much exploration. It just collected wood resources that were nearby. The houses built by the agent were represented by the square boxes of the

agent's colour. Figure 19 represents the path taken by the agents during the simulation. The red star indicates the agent's location initially, and the green star represents the agent's location at the end of the simulation.

The coin resource is considered as an income for agents. The figure 18 describes the coins earned by dark blue and light blue agents is less compared to red and yellow agents. However, the dark blue and light blue agent's income started to rise after 100 timesteps. Overall, the productivity of agents was improving over the timesteps and the income gap between the agents is very high. Figure 20 shows the trading activities of agents at different timesteps. The green line represents wood resources, while the white line represents the stone resource. If the green line stops at value 5 in y-axis, it means the agent sold a wood resource for 5 coins. If the value is -5, then it represents the agent bought a wood resource for 5 coins. The x-axis represents the timesteps. Trading activity is rare in the Free Market because low-income agents do not have much money to buy resources. Therefore, the flow of resources between the agents is limited (Figure 20).

6.2 Universal Basic Income simulation

The Universal Basic Income (UBI) policy provides a fixed amount of funds to all agents at each timestep. The tax collected from the agents was used to fund the policy.

Figure 21: Universal Basic Income policy simulation.

Figure 21 depicts the simulation of UBI policy for each 50 timestep interval.

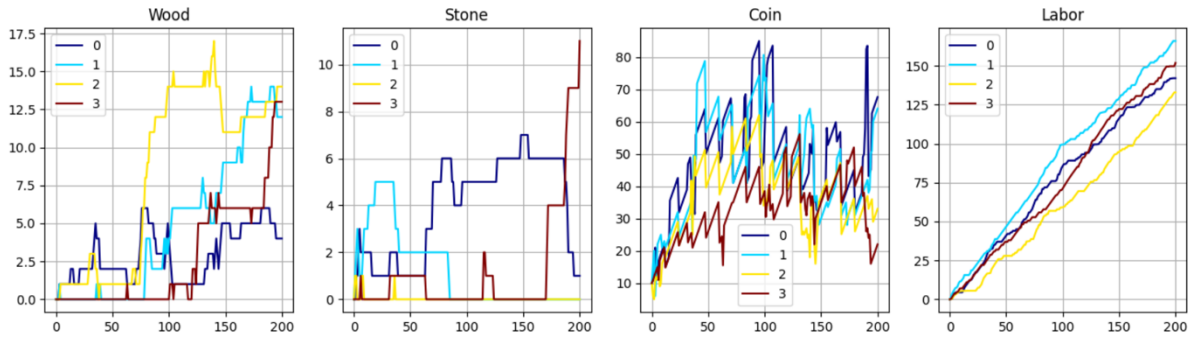

Figure 22: Agent resources and labour cost in UBI

Figure 22 represents the wood(a), stone(b), coin(c) resources, and labour cost of agents at each time step for UBI policy.

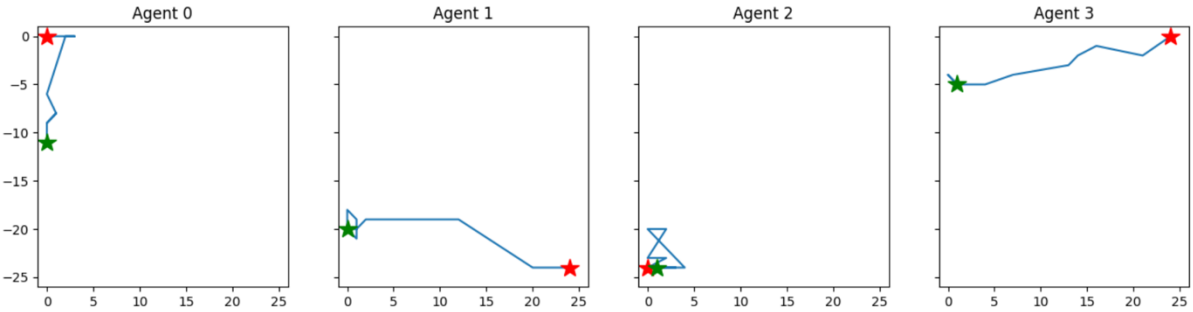

Figure 23: Agent path in UBI

Figure 23 represents (a) agent 0, (b) agent 1, (c) agent 2, and (d) agent 3 paths taken by agents during UBI simulation experiment. The red start indicates starting location and green star indicated location of the agent at the end of the simulation.

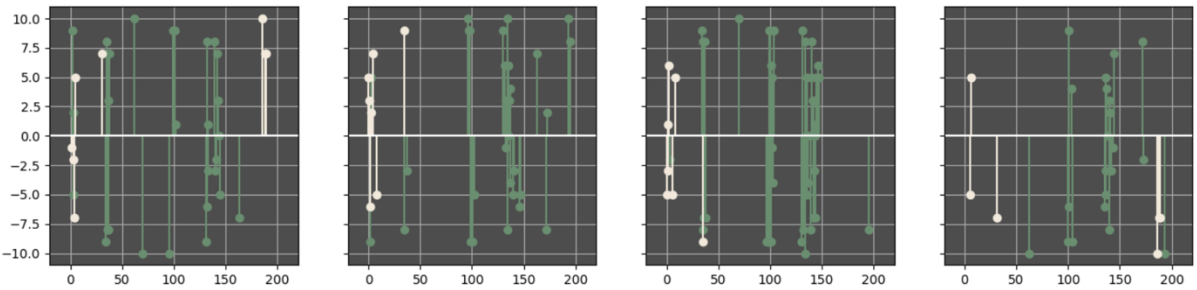

Figure 24: UBI trading activities.

The Figure 24 represents (a) agent 0, (b) agent 1, (c) agent 2, (d) agent 3 trading activities in UBI policy. The green and white line indicates wood and stone resources that were traded at a specific timestep.

The red agent in the top right corner does not have any wood or stone resources. Hence, in search of resources, the red agent moved to the first quadrant after 100 timesteps. However, the red agent coin resource saw a gradual rise from timestep 0 to 100 because of the funds it received from UBI policy at each timestep (Figure 22). In Free market policy, the coins earned by the agents are not intervened by income tax. Hence, the coin earned by the agent only depends upon the agent's action. In contrast, the coin earned by the agent in the UBI policy depends upon tax collected from the agent, the amount of funds received and the agent's actions. The income of the agent (i.e. coin) shows some seasonal behaviour because of the tax collected at each 12 timestep interval. Although there is seasonal behaviour, the coin earned by the agents tends to increase. However, the increasing trend began to fall gradually after 100 timesteps. Figure 22 shows that the agent's participation in collecting wood was increased after 100 timesteps. The red and light blue agents from the right side moved to the left side in search of resources, and they performed more labour when compared to the other two agents. The path taken by the UBI agents is similar to the Free market policy (Figure 23,19). The income gap between the agents in UBI is not very high when compared to the Free market. The trading activities are high compared to the Free Market because low-income agents will receive the UBI fund, and it helps the agent to buy resources. The trading activity is high on specific timesteps like 100 and 150 because the agents struggle to collect resources after 100th time step, so the agents begin to trade resources. The decline in income at 100th and 150th time step is due to the agent spending most of its earned income on buying resources.

6.3 Conditional Cash Transfer simulation

The Conditional Cash Transfer (CCT) program provides fund to low-income agents at each time step.

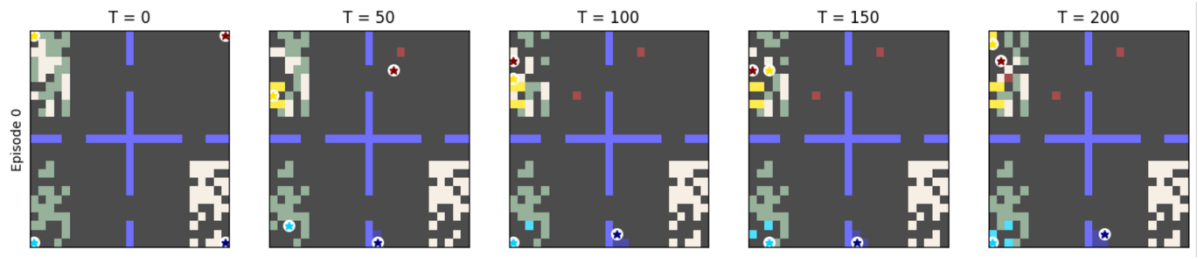

Figure 25: Conditional Cash Transfer policy simulation.

Figure 25 depicts the simulation of CCT policy for each 50 timestep interval.

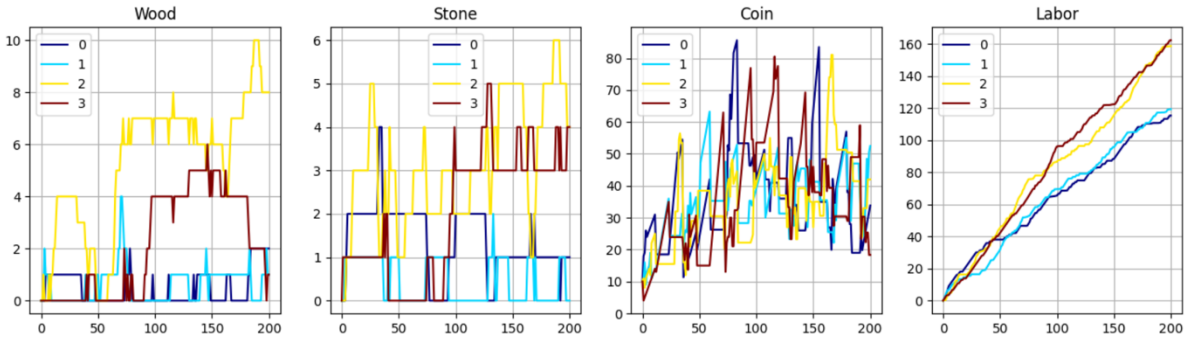

Figure 26: Agent resources and labour cost in CCT

Figure 26 represents the wood(a), stone(b), coin(c) resources, and labour cost of agents at each time step for CCT policy.

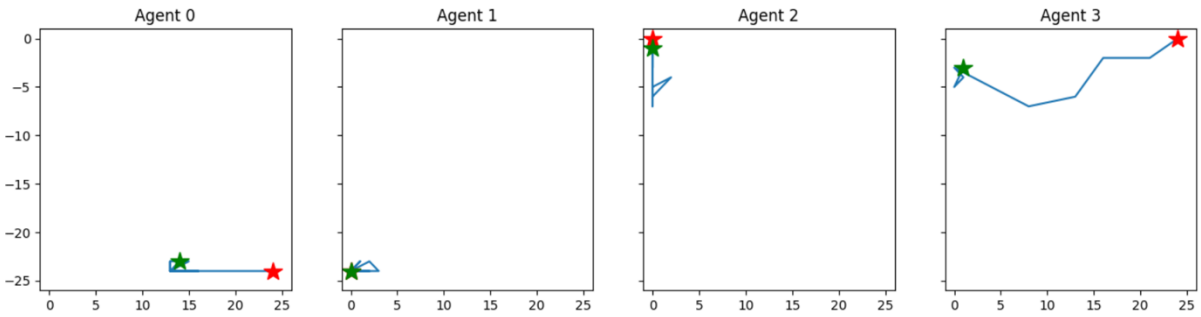

Figure 27: Agent path in CCT

Figure 27 represents (a) agent 0, (b) agent 1, (c) agent 2, and (d) agent 3 paths taken by agents during CCT simulation experiment. The red start indicates starting location and green star indicated location of the agent at the end of the simulation.

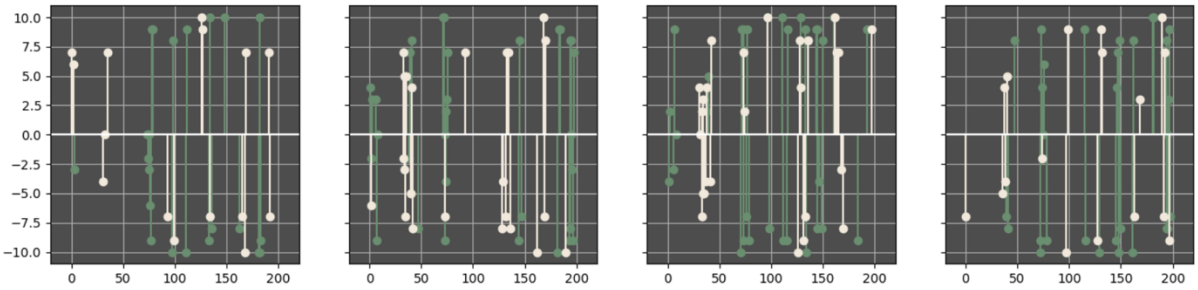

Figure 28: CCT trading activities.

The Figure 28 represents (a) agent 0, (b) agent 1, (c) agent 2, (d) agent 3 trading activities in CCT policy. The green and white line indicates wood and stone resources that were traded at a specific timestep.

During the initial stages of the simulation, the red agent did not have any resources nearby, but the funds it received from the CCT policy were very high compared to the UBI policy. Therefore, the red agent used the CCT fund to buy stone and wood resources and built two houses before reaching 100 timesteps. The income earned by the red agent (i.e initially agent with no resource nearby) in CCT policy is the highest when compared to UBI and Free Market. Figure 26 shows that the income of the agents fluctuated throughout the simulation. The low-income agent will receive more funds, which helps the agent to trade or build houses, and subsequently, the income of that agent will rise, whereas the income of the high-income individual will fall because the agent is excluded from receiving funds, and additional tax will be collected from the agent. This leads to an income fluctuation in CCT policy. The fluctuation in the agent's income also paves the way for more trading activities when compared to UBI and Free Market policy. The supply and demand of the resources are balanced because of frequent trading activities.

6.4 Productivity

The productivity of the agent is the total amount of coins that an agent has earned at each timestep.

$$Productivity_{t=1} = \sum_{i=0}^{n-1} coin_{t=1}^{i}$$

t： time step

n： number of agents

$\mathbf{coin}_{t=1}^{i}$ ：amount of coin that an agen 0 has at timestep i

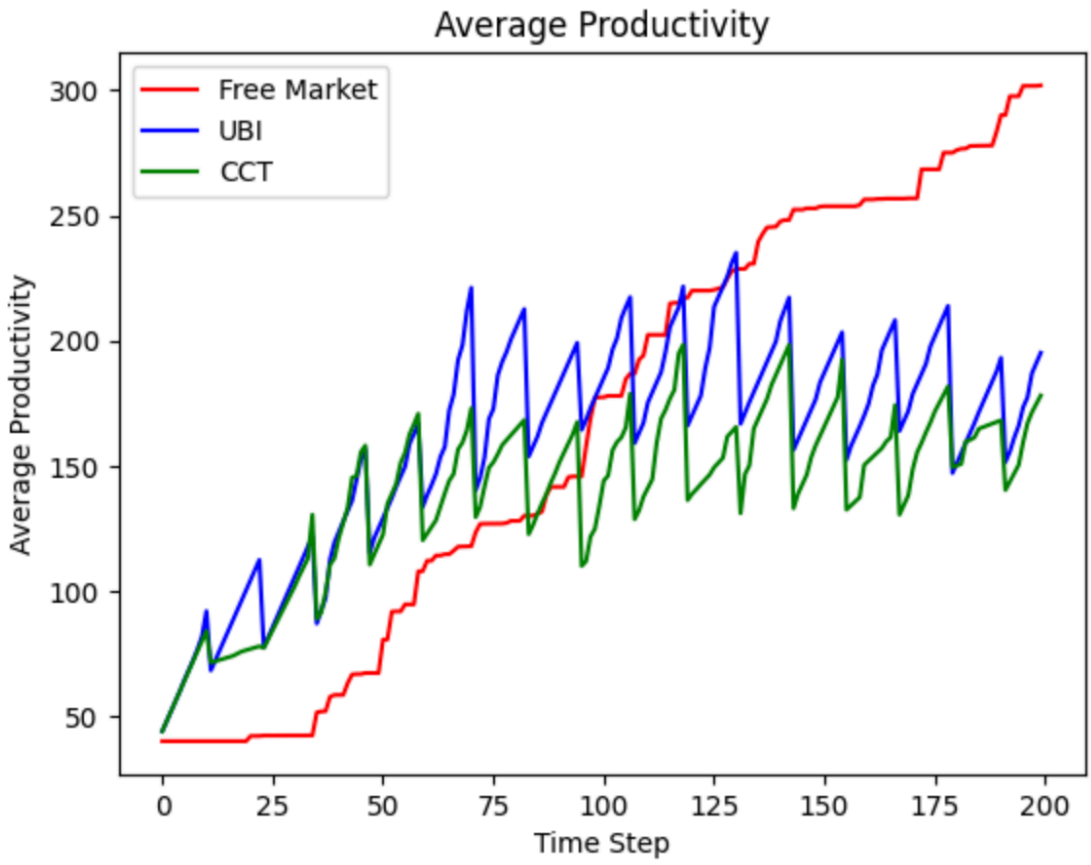

Figure 29: Average productivity at each timestep

Figure 29 represents the average productivity at each timestep for UBI, CCT and Free Market policies. The average productivity value is calculated by averaging ten simulation experiment results for each policy.

Initially, the UBI and CCT policies had similar productivity results. However, after the 60th timestep, the productivity of UBI is greater than the CCT policy (Figure 29). The productivity of CCT and UBI does not see growth after 100th timestep. The Productivity of the Free Market tends to increase steadily, and it reaches nearly 300 at 200th timestep (Figure 29). The productivity results clearly states that redistribution policies such as UBI and CCT will hinder economic growth, but it may not be visible during the initial stages.

6.5 Income Equality:

The income equality is used to measure the wealth distribution of the agents.

$$Equality\ percentage = Equality * 100$$

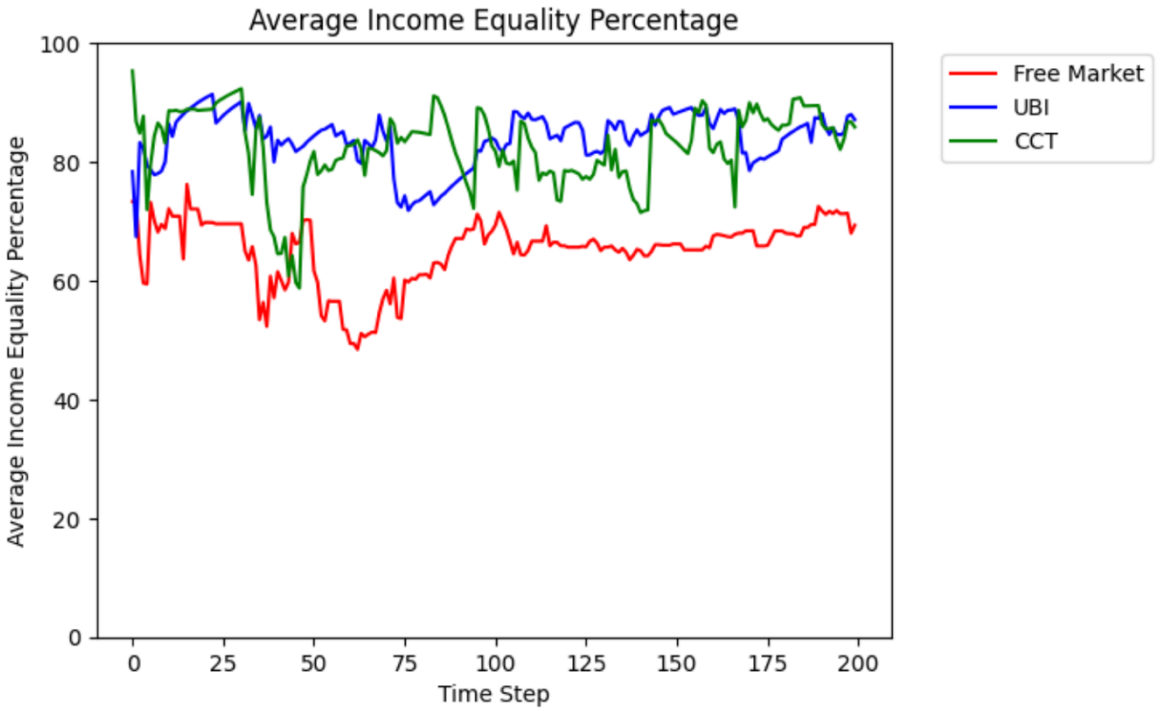

Figure 30: Average Income Equality Percentage at each timestep

Figure 30 represents the average income equality percentage at each timestep for UBI, CCT and Free Market policies. The average income equality percentage value is calculated by averaging ten simulation experiment results for each policy.

The equality percentage of UBI and CCT policy is above 80% throughout the simulation period. There is no clear difference between UBI and CCT policy in equality percentage. However, the CCT policy equality percentage showed more fluctuation compared to the UBI policy. The Free market has the lowest equality percentage when compared to other policies. The equality percentage of the Free market fell to 50% at the 55th timestep, but after that, the equality percentage started to rise gradually and reached nearly 70%.

7 Conclusion

This research was conducted to analyze the impact of Universal Basic Income and Conditional Cash Transfer policy by balancing both productivity and equality. The existing research studies were based on numerical calculation and microsimulation based on a single objective. This research looked at advanced research techniques which were able to optimize multi-objective problems in a simulation environment. The research question was to analyze the impact of both redistribution policies in the short-term and long-term and whether redistribution policies promote productivity and consumption.

In order to answer the research questions, an advanced simulation environment which reflects the economic world was selected, and then the functionality of UBI and CCT policies was added to the simulation environment. The multi-agent design was created to train multiple agents in the simulation environment based on two objectives, equality and productivity. The Neural Network model was used to train agents using the PPO algorithm. The trained agents were used to perform simulation for UBI and CCT policies, and the results of the simulation were analyzed.

The simulation results showed that productivity of UBI is slightly higher than CCT policy. In long-term redistribution policies have a negative impact on productivity. The social welfare of the people was improved greatly with the help of redistribution policies. The CCT policy promotes consumption. Therefore, the resources were not accumulated only by high-income individuals. The resources were circulated among all people.

7.1 Limitation

- Due to GPU (Graphics Processing Unit) resource constraints, the reinforcement learning model is not trained on a distributed environment with multiple GPUs. The model could have given even better results if it was trained on multiple GPUs.
- The simulation environment is designed to reflect the real world, but in reality, it's not possible to add every complexity of the physical world.
- The AI agents are not trained using human-behavioral data. Therefore, the agent's action could not reflect complete human behaviour.

7.3 Future works

7.3.1 Estimating Social Welfare

In this research social welfare is considered as $Productivity \times Equality$ but social welfare would also include public health, happiness, and other factors. Therefore, multiple factors which were related to social welfare could be added to give a clear definition of social welfare.

7.3.2 Agent Relationship

In this research, all agents were individuals, and they do not have any relationship between them, but in the real world, people have families and friends, and their actions are heavily biased based on their relationship. Therefore, adding an agent relationship to the environment would help to reflect human behaviour.

7.3.3 Testing policy on small-scale

The policy designed by the agent can be implemented on a small scale, and feedback from the implemented policy can be used to validate the model.

7.7.4 Exploring policy combinations

In this research, the impact of a single policy can be analyzed, but the governments would implement a new policy while upholding existing policy. Therefore, flexibility to add or removing new policies during the simulation would help the economist to analyse the complex behaviour of the economic policies.

Bibliography

[1] Benhabib, J. (2003) *The Tradeoff Between Inequality and Growth*, *ANNALS OF ECONOMICS AND FINANCE*.

[2] Boadway, R., Cuff, K. and Koebel, K. (no date) *Designing a Basic Income Guarantee for Canada*.

[3] Buşoniu, L., Babuška, R. and De Schutter, B. (2008) 'A comprehensive survey of multiagent reinforcement learning', *IEEE Transactions on Systems, Man and Cybernetics Part C: Applications and Reviews*, pp. 156–172. Available at: https://doi.org/10.1109/TSMCC.2007.913919.

[4] Colin F. Camerer (2011) *Behavioral Game Theory*. Princeton University Press.

[5] Das, J., Do, Q.T. and Özler, B. (2005) 'Reassessing conditional cash transfer programs', *World Bank Research Observer*, 20(1), pp. 57–80. Available at: https://doi.org/10.1093/wbro/lki005.

[6] DeScioli, P., Shaw, A. and Delton, A.W. (2018) 'Share the Wealth: Redistribution Can Increase Economic Efficiency', *Political Behavior*, 40(2), pp. 279–300. Available at: https://doi.org/10.1007/s11109-017-9392-x.

[7] Dyson, B. and Chang, N. Bin (2005) 'Forecasting municipal solid waste generation in a fast-growing urban region with system dynamics modeling', *Waste Management*, 25(7), pp. 669–679. Available at: https://doi.org/10.1016/j.wasman.2004.10.005.

[8] Espeholt, L. *et al.* (2018) 'IMPALA: Scalable Distributed Deep-RL with Importance Weighted Actor-Learner Architectures', *arXiv.org* [Preprint].

[9] FRANKLIN, S. (1997) 'AUTONOMOUS AGENTS AS EMBODIED AI', *Cybernetics and systems*, 28(6), pp. 499–520. Available at: https://doi.org/10.1080/019697297126029.

[10] Hall, R.P. *et al.* (2019) 'Universal basic income and inclusive capitalism: Consequences for sustainability', *Sustainability (Switzerland)*, 11(16). Available at: https://doi.org/10.3390/su11164481.

[11] Hanna, R. and Olken, B.A. (2018) 'Universal basic incomes versus targeted transfers: Anti-poverty programs in developing countries', in *Journal of Economic Perspectives*. American Economic Association, pp. 201–226. Available at: https://doi.org/10.1257/jep.32.4.201.

[12] Hodge, V.J., Hawkins, R. and Alexander, R. (2021) ‘Deep reinforcement learning for drone navigation using sensor data’, *Neural Computing and Applications*, 33(6), pp. 2015–2033. Available at: https://doi.org/10.1007/s00521-020-05097-x.

[13] Holland, J.H. and Miller, J.H. (1991) ‘Artificial Adaptive Agents in Economic Theory’, *The American economic review*, 81(2), pp. 365–370.

[14] Kakwani, N. and Son, H.H. (2022) ‘Introduction’, in *Economic Inequality and Poverty*. Oxford University PressOxford, pp. 1–2. Available at: https://doi.org/10.1093/oso/9780198852841.003.0001.

[15] León-Castro, E. *et al.* (2019) ‘Modelling and simulation in business, economics and management’, *Technological and Economic Development of Economy*, 25(4), pp. 571–575. Available at: https://doi.org/10.3846/tede.2019.9365.

[16] Marinescu, I. (2018) ‘No Strings Attached: The Behavioral Effects of U.S. Unconditional Cash Transfer Programs’, *NBER Working Paper Series*, p. 24337. Available at: https://doi.org/10.3386/w24337.

[17] Mnih, V. *et al.* (2013) ‘Playing Atari with Deep Reinforcement Learning’. Available at: http://arxiv.org/abs/1312.5602.

[18] Mourtzis, D., Doukas, M. and Bernidaki, D. (2014) ‘Simulation in Manufacturing: Review and Challenges’, *Procedia CIRP*, 25, pp. 213–229. Available at: https://doi.org/10.1016/j.procir.2014.10.032.

[19] Narvekar, S. (2017) *Curriculum Learning in Reinforcement Learning*. Available at: http://www.intplay.com/uploadedFiles/Game.

[20] Neves, J.A. *et al.* (2022) ‘The Brazilian cash transfer program (Bolsa Família): A tool for reducing inequalities and achieving social rights in Brazil’, *Global Public Health*, 17(1), pp. 26–42. Available at: https://doi.org/10.1080/17441692.2020.1850828.

[21] OpenAI *et al.* (2019) ‘Dota 2 with Large Scale Deep Reinforcement Learning’. Available at: http://arxiv.org/abs/1912.06680.

[22] Rawlings, L.B. and Rubio, G.M. (2005) ‘Evaluating the impact of conditional cash transfer programs’, *World Bank Research Observer*, 20(1), pp. 29–55. Available at: https://doi.org/10.1093/wbro/lki001.

[23] Riedmiller, M. (2005) ‘Neural Fitted Q Iteration – First Experiences with a Data Efficient Neural Reinforcement Learning Method’, in *Lecture notes in computer science*. 1ère éd. Berlin, Heidelberg: Springer Berlin Heidelberg (Lecture Notes in Computer Science), pp. 317–328. Available at: https://doi.org/10.1007/11564096_32.

[24] Saez, E. (2002) ‘Optimal Income Transfer Programs: Intensive versus Extensive Labor Supply Responses’, *The Quarterly journal of economics*, 117(3), pp. 1039–1073. Available at: https://doi.org/10.1162/003355302760193959.

[25] Schulman, J. *et al.* (2017) ‘Proximal Policy Optimization Algorithms’. Available at: http://arxiv.org/abs/1707.06347.

[26] Sloman, J., Guest, J. and Garratt, D. (2021) *Economics*. Pearson Education.

[27] Špeciánová, J. (2018) ‘Unconditional basic income in the Czech Republic: What type of taxes could fund it? A theoretical tax analysis’, *Basic Income Studies*, 13(1). Available at: https://doi.org/10.1515/bis-2017-0024.

[28] Stoeffler, Q., Mills, B. and del Ninno, C. (2016) ‘Reaching the Poor: Cash Transfer Program Targeting in Cameroon’, *World Development*, 83, pp. 244–263. Available at: https://doi.org/10.1016/j.worlddev.2016.01.012.

[29] Su, J. *et al.* (2022) ‘Deep multi-agent reinforcement learning for multi-level preventive maintenance in manufacturing systems[Formula presented]’, *Expert Systems with Applications*, 192. Available at: https://doi.org/10.1016/j.eswa.2021.116323.

[30] Sutton, R.S. and Barto, A.G. (2018) *Reinforcement learning: an introduction*. The MIT Press (Adaptive computation and machine learning).

[31] Tesfatsion, L. (2006) ‘Chapter 16 Agent-Based Computational Economics: A Constructive Approach to Economic Theory’, *Handbook of Computational Economics*, pp. 831–880. Available at: https://doi.org/10.1016/S1574-0021(05)02016-2.

[32] Trott, A. *et al.* (2021) ‘Building a Foundation for Data-Driven, Interpretable, and Robust Policy Design using the AI Economist’. Available at: http://arxiv.org/abs/2108.02904.

[33] White, A. (no date) *How to deal with labour in the superstar (digital) economy?* Available at: https://orcid.org/0000-0002-0793-0142.

[34] Zheng, S. *et al.* (2020) ‘The AI Economist: Improving Equality and Productivity with AI-Driven Tax Policies’. Available at: http://arxiv.org/abs/2004.13332.

[35] Zhu, Z. *et al.* (2023) ‘Transfer Learning in Deep Reinforcement Learning: A Survey’, *IEEE Transactions on Pattern Analysis and Machine Intelligence*, pp. 1–20. Available at: https://doi.org/10.1109/TPAMI.2023.3292075.