Eidgenössische Technische Hochschule Zürich
Swiss Federal Institute of Technology Zurich

Distributed Computing

# Multiagent Reinforcement Learning in Financial Networks

Distributed Systems Lab


Bryan Yu

Bryayu@student.ethz.ch

Distributed Computing Group
Computer Engineering and Networks Laboratory
ETH Zürich


**Supervisors:**
Pál András Papp, Lukas Faber, Béni Egressy
Prof. Dr. Roger Wattenhofer

February 25, 2021

# Acknowledgements

Thank you Lukas Faber, Pál András Papp, Béni Egressy, and Professor Roger Wattenhofer for supporting my interest to study single-agent and multi-agent reinforcement learning. You've all been great at providing guidance and sharing enlightening discussions on on how to continue the research into the future.

# Abstract

Banking crises are complex events involving many banks and the network of interconnection between them. Furthermore, the health of the banking system is contingent on the current state of all its banks and the choices each bank decides to take, whether it is in the collective interest of the group or whether actions are taken solely based on self-interest.

In this thesis, we extend previous agent based models in inter-bank networks by training their behavior using multi-agent reinforcement learning methods. We first establish a performance baseline in our stylized environment using single-agent reinforcement. We then move to the multi-agent setting and empirically review the learned behavior of the agents.

Our preliminary results showed that simulations with a small number of banks result in higher concentration of inter-bank capitalization. Furthermore, our initial attempt at training agents using multi-agent reinforcement learning methods resulted in behaviors that do not yet appear representative of real world decision making.

# Contents

Chapter 1

# Introduction

In the modern economy, healthy functioning financial system facilitates the flow of trade and commerce. After a period of stability, lenders may feel a sense of security and loosen their lending criterion resulting in leveraged balance sheets with greater liabilities than assets. As a result, events such as the 1997 Asian financial crisis, 2008 U.S. financial crisis, 2009 European debt crisis, 2013 Cyprus banking crisis, 2018 Turkish currency and debt crisis, and other financial events have occurred in the recent history. Furthermore, as the world becomes increasingly connected, the ability to reason about complex adaptive systems will be an important tool in preparing and mitigating the effects of future financial crisis.

The study of financial systems modeled as inter-bank networks with bilateral relationships has been an important tool to reason about system dynamics. Previous approaches included studying network formations[1], the implications of network structure [1], dynamic general stochastic equilibrium(DGSE) modeling [2, 3], and agent-based models [4].

Developments in multi-agent reinforcement learning and machine learning has introduced new mechanisms for training agents in co-operative tasks [5], attending to various features in an agent's observation history [6, 7], and the ability to learn embeddings that capture the structure of a network [8, 9, 10, 11].

Our goal is to learn how to train inter-bank agents to learn policies using multi-agent reinforcement learning. By doing so, we hope to incorporate into the inter-bank network literature an approach to design heterogeneous agents with policies that considers individual and neighbouring features, network structure, and network dynamics.

Chapter 2

# Financial System

Financial systems are complex adaptive systems connecting actors such as banks, investors, firms, etc. Furthermore, the number of actors in the system, the strength of the connection between actors, and the creation/termination of connections are subject to change over time. To fit resource constraints, we focus on inter-bank networks, a subset of a complete financial systems, which aggregates the features of the banking system into agents, capitalization, and bilateral liabilities which can collectively be represented as a weighted directed graph [12].

We begin with a discussion of related works in Section 2.1. Section 2.2 discusses the how the structure of the banking system is modeled within the simulation. Section 2.3 discusses the decisions available to the bank-agents. Section 2.4 discusses how default is determined and its effects on the bank-agent. Section 2.5 discusses the clearing mechanism and how it is used to determine the position of the system.

## 2.1 Related Work

Inter-bank Network models focus on the banking sector and the interdependent connections between banks arising from inter-bank liabilities through market activity, portfolio holdings, and other financial instruments. This naturally lends itself to a network model representation with nodes depicting banks and links depicting inter-bank relations. Furthermore, network models allow research a window to build insights on how banking crises and contagion can arise.

Allen and Gale [1] and Frexis et al [13] pioneered theoretical work on the structure of inter-bank networks and suggested that inter-bank networks with high connectivity results in enhanced system resilience when faced with individual bank defaults because risk is spread evenly across all other banks. However, individual bank defaults can impact the financial health of other banks through knock-on effects arising from inter-bank liabilities. To approximate the impact, Eisenberg and Noe [14] introduces a clearing vector algorithm that provides a solution to the higher order feedback. Alternatively, Furfine [15] introduces a

sequential algorithm to arrive at a cleared system. However, Furfine's algorithm does not account for the simultaneity problem arising from higher order defaults.

Inter-bank liabilities can arise from a variety of channels such as direct linkages arising from direct loans or indirect linkages arising from banks holding the same asset or newer financial instruments such as credit default swaps (CDS). One method to account for the multi-channel nature of liabilities is through multi layer networks[16]. Montagna and Kok [17] applies multi-layer networks and agent-based models to allow for a more holistic approach and reveal non-linearities in the propagation of shocks which may be substantially larger than losses at the individual layer networks.

For a broader view of the literature, a number of surveys on inter-bank networks are available. Allen and Babus [18] presents an early survey of the inter-bank network literature noting the methods used and then knowledge base on banking crises, contagion, bubbles.

Upper's [19] survey focuses on simulation methodologies and assumptions used to test for contagion in inter-bank markets. Upper found that a majority of papers reviewed in the survey simulate the failure of individual banks which stands in contrast to real world banking crises with shocks that affect several banks simultaneously and that behavior foundations were absent in the analysis of contagion. Upper concludes that contagion arising from inter-bank liabilities is likely to be rare but allows that this finding may be due to strong underlying assumptions and rudimentary behaviors in the simulations. As such, the models surveyed are useful tools to analyze financial stability but cannot be relied on to predict crises.

Hüser's [20] presents a more recent survey focused on the theoretical aspect of how network structure affects contagion and how bank form connections faced with the possibility of contagion and systemic risks. Hüser reviews networks based on empirical studies and notes their inability to draw conclusions on generic relationships between contagion risk. Hüser next reviews the theoretical inter-bank network literature and notes the generalized insights learned such as which network structures tend to dampen or propagate defaults. Hüser then discusses methods to model the dynamic processes of how firms enter into obligations. In other words, the process of link formation with which Hüser identifies three main methods being (1) preferential attachment which conditions link formation on the characteristics of other nodes (2) strategic network formation where banks assess the cost and benefits of link formation in cases such as roll over decisions and (3) endogenous network formation with banks determining the amount of inter-bank lending/borrowing as an optimization of their balance sheets.

Bargigli [21] surveys the use of agent-based models in economics and network models. Local agent design characterizes agent behavior based on interactions within an agent's neighbourhood whereas global agent design characterizes agent behavior based on the behavior of all agents. Furthermore, agent behaviors can be

characterized as being exogenous(i.e. decisions are contingent on the utility of the neighbourhood), endogenous(i.e. decisions are contingent on individual agent's utility), deterministic, or stochastic. Agents can find themselves operating in a static network structure where everything is determined once or dynamic which allows the network to evolve over time. Bargigli reviews the way individual incentives affect link formation and the resulting network structures arising from agent interactions.

## 2.2 Banking Model

The inter-bank network, $I$, can be defined as the triple $I = (X, L, C)$[12] where

- X={1,2,...n} is the set of nodes and each node represents a bank
- C=$\{c_1, c_2, ..., c_n\}$ denotes the capitalization of bank $i$
- $L \subset \mathbb{R}^{n \times n}$ is the liabilities matrix. $L_{ij}$ is the weight of liability from bank i to bank j. We define all values along the diagonal to 0 because banks do not lend to themselves.

Thus the inter-bank network can be depicted as a weighted directed graph with nodes representing banks and weighted edges representing as per Figure 2.1. The capitalization of each bank capitalization can be depicted as per Figure 2.2 and the liabilities matrix similarly as per Figure 2.3.

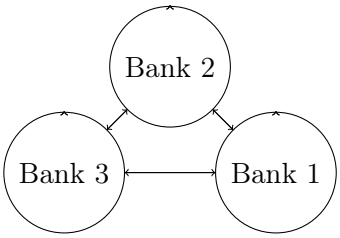



Figure 2.1: Graph representation of banking network

| Bank | Capitalization |
|---|---|
| 1 | 250 |
| 2 | 500 |
| 3 | 250 |

Figure 2.2: Capitalization matrix $C$ capturing the capitalization of each bank.

| Bank | 1 | 2 | 3 |
|---|---|---|---|
| 1 | 0 | 150 | 20 |
| 2 | 50 | 0 | 800 |
| 3 | 10 | 20 | 0 |

Figure 2.3: liabilities matrix $L$ depicting the amounts owed from the row to the column bank.

The inter-bank liability exposure of the $i$-th bank, $l_i$, is thus the sum of the $i$-th column of the liabilities matrix. Similarly, the inter-bank asset of the $j$-th bank, $\mu_j$, is then the sum of the $j$-th column.

$$l_i = \sum_{j=1}^{n} L_{ij} \tag{2.1}$$

$$\mu_j = \sum_{i=1}^{n} L_{ij} \tag{2.2}$$

A bank $j$ is in $\Psi_i^t$, the set of creditors to bank $i$, if $L_{ij}^t > 0$. Similarly, a bank $i$ is in $\Delta_j^t$, the set of debtors to bank $j$ if $L_{ij}^t > 0$.

$$\Psi_i^t := \{j \mid L_{ij}^t > 0\} \tag{2.3}$$

$$\Delta_j^t := \{i \mid L_{ij}^t > 0\} \tag{2.4}$$

## 2.3 Link Creation

In inter-bank networks, the liabilities captured by weighted edges are key to capturing the structure of the networks. Furthermore, inter-bank activity results in the continual evolution of the network with edges continuously being created and removed. We allow for dynamic network evolution by allocating agent $i$ a decision at every time step $t$ captured by the vector $a_i^t$. The joint action across all agents is then the $n \times n$ joint action matrix, $A^t$, as depicted in Figure 2.4.

| Bank | 1 | 2 | 3 |
|---|---|---|---|
| 1 | 0.50 | 0.25 | 0.25 |
| 2 | 0.10 | 0.90 | 0.00 |
| 3 | 0.05 | 0.5 | 0.5 |

Figure 2.4: Example of a joint action matrix, $A$, capturing each bank's creation and removal of liabilities at time $t$.

## 2.4 Default

Banking defaults is the event when banks are unable to repay their debts. At time $t$, the set of defaulted banks is defined as $\Omega^t$. Bank $i$ is in the set of defaulted banks if $\omega_i$, the net position of bank $i$, is negative as per Equation 2.5.

$$\omega_i = c_i + \mu_i - l_i \tag{2.5}$$

$$\Omega^t = \{i \mid \omega_i < 0\} \tag{2.6}$$

In case of default, we assume only a portion of total capitalization is recovered by the defaulted bank, $\omega_i$, through a discounted sale (assets are sold less than they are recorded at). As with Rogers & Veraart [22], the haircut multiplier $\alpha$ is applied to the capitalization $c_i$ of bank $i$. We assume that the inter-bank asset $a_i$ of bank $i$ is uncollectible. Thus $\rho_i$ is the recovered amount by bank $i$.

$$\rho_i = \alpha \cdot c_i \tag{2.7}$$

The defaulted bank $\omega_i$ is then liquidated and the recovered amount $\rho_i$ is distributed to $\Psi_i$, the creditors of bank $i$. Each creditor, $k$ receives a distribution in proportion to their share of bank $i$'s total liabilities $l_i$. The distribution from defaulted bank $\omega_i$ to creditor bank $k \in \Psi_i$ is thus $\psi_{ik}$ defined by Equation 2.8.

$$\psi_{ik} = \rho_i \cdot \frac{L_{ik}}{l_i} \tag{2.8}$$

## 2.5 System Value

We determine $\sigma$, the system value of the inter-bank network at time $t$, by approximating the value each bank would receive in an instantaneous dissolution. We take a stylized approach to account for higher order defaults due to knock-on effects by allowing a maximum three rounds of clearing or early halting if the set of debtors, $\Delta_i$, and creditors, $\Psi_i$, to bank $i$ do not change for all $i \in X$ with another round of clearing.

Defaulted bank $i \in \Omega^t$ are processed first by applying the haircut multiplier $\alpha$ to bank capitalization $c_i$. Then the recovered amount $\rho_i$ is distributed as per Section 2.4. For non-defaulted banks, $i \notin \Omega^t$, the inter-bank liabilities $l_i^t$ are paid as per the inter-bank exposure $L_{ij}^t$.

**Algorithm 1** Clearing Algorithm

Compute $\Psi_i^t$, $\Delta_i^t$ $\forall i$
**for** iteration = 1,...,3 **do**
  **for** $i \in \Omega^t$ **do**
    $\rho_i^t = \alpha \cdot c_i^t$
    $c_i^t = 0$
    **for** $j \in \Psi_i^t$ **do**
      $c_j^t = c_j^t + \psi_{ij}^t$
      $L_{ij} = 0$
    **end for**
  **end for**
  **for** $i \notin \Omega^t$ **do**
    **for** $j \in \Psi_i^t$ **do**
      $c_j^t = c_j^t + L_{ij}^t$
      $L_{ij} = 0$
    **end for**
  **end for**
  Compute $\Psi_i^{t+1}$, $\Delta_i^{t+1}$ $\forall i$
  **if** $\Psi_i^{t+1} = \Psi_i^t$ and $\Delta_i^{t+1} = \Delta_i^t$ $\forall i$ **then**
    break
  **end if**
**end for**
$\sigma^t = \sum C^t$

Chapter 3

# Methods

Reinforcement learning's (RL) accomplishments in games such as Starcraft[23] and Go[24] indicates the potential of learning machines to identify policies that can outperform humans. Single-agent reinforcement learning trains one agent operating in an environment and fits naturally into games such as chess and checkers. Alternatively, multi-agent reinforcement learning trains multiple agents acting in the same environment and fits naturally into situations such as competing banks. In this section we will discuss both single and multi-agent reinforcement and our motivation to learn how to apply these methods to train inter-bank network agents.

## 3.1 Reinforcement Learning

Reinforcement learning focuses on the interaction between an agent, the environment, and the usage of experience histories to learn decision making policies which maximize the accumulation of a reward [25]. As per Figure 3.1, at every time-step $t$, the environment is in a particular state $S_t$ which produces observations $o_t$ which the agent observes. Given the observations, the agent decides an action $\Lambda_t$ with which it determines the environment's transition into the next state $S_{t+1}$. Furthermore, the interaction with the environment triggers a reward signal $R_t$ which is received by the agent which is used to determine the value of action taken.

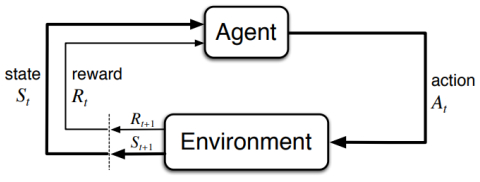



Figure 3.1: Reinforcement Learning agent-environment interaction. Image from "Reinforcement Learning" by Sutton Barto[25]

## 3.2 Single Agent Reinforcement Learning

We use single-agent reinforcement learning to establish a baseline to compare performance in the stylized inter-bank network. A practical interpretation of applying single-agent reinforcement learning can be motivated by anecdotal evidence of central actors such as JP Morgan in the Panic of 1907 [26] or the U.S. Federal Reserve under Geitner in 2008[27].

Single agent reinforcement learning, setting are typically denoted by the tuple $\mathbf{G} = (S, \Lambda, P, r, \gamma, p_0, T)$ where $S$ are the states, $\Lambda$ the action space, $P : S \times \Lambda \to S'$ is the transition function, $r : S \times \Lambda \to \mathbb{R}$ is the reward function, $p_0 : S \to \mathbb{R}$ the initial state distribution, $\gamma$ the discount factor, and T the time horizon. By interacting within this setting in the feedback loop in Figure 3.1, the agent uses past experiences to learn a policy, $\pi : S \times \Lambda \to \mathbb{R}$, which maximizes the discounted expected future return as per Equation 3.1.

$$\max_{\pi} \mathbb{L} \sum_{t}^{\infty} [\gamma^t \cdot r(s_t, \pi(s_t))] \tag{3.1}$$

Two popular approaches are value based and policy based methods. Value based methods attempt to learn the optimal value of a state $V^*$ or state-action pair $Q^*(s, a)$ with methods such as TD methods, Q-learning, and SARSA [25]. Policy based methods focus on learning an optimal policy $\pi^*$ mapping states to actions with algorithms such as REINFORCE and actor-critic approaches[25]. Policy based alleviates the cost of learning the value of the state action space and instead learns the parameters of an approximating function. However, this increases the variances amongst the learned policies[25].

### 3.2.1 Proximal Policy Optimization

We considered the family of policy based methods and identified Proximal policy optimization (PPO) [28] which addresses issues such as high correlation between states in an episode by sampling a minibatch from the experiences of multiple agents running in parallel, accounting for destructive policy updates by using clipped policy updates which restricts the size and effect of updates, and improves training times by allowing multiple simulations to run in parallel. Furthermore, PPO allows for the design of agents capable of acting in a continuous action space which supports the link creation mechanism.

**Algorithm 2** PPO, Actor-Critic Style from Schulman et al. [28]

**for** *iteration=1,2...* **do**
  **for** *agent in environment* $k \in \{1, 2, ..., K\}$ **do**
    Run policy $\pi_{\theta_{old}}$ in $k$ for $T$ timesteps
    Compute advantage estimates $\hat{A}_1, ..., \hat{A}_T$
  **end**
  Optimize surrogate L wrt $\theta$, with K epoch and minibatch size M $\leq NT$
  $\theta_{old} \leftarrow \theta$
**end**

### 3.2.2 Single-agent Environment

We prepare the inter-bank network, $I$, into the single-agent reinforcement learning setting, $\mathbf{G}$, by mappings as follows.

- States, $S$, are defined as all possible configurations of the inter-bank network $I$. We simulate situations in which n = 3, 10, or 25. Total inter-bank capitalization is defined as 1000 (i.e. $\sum_{i=0}^{n} c_i = 1000$). The total inter-bank capitalization is randomly distributed across all banks. Total inter-bank liabilities can take values such that 0 and 2000 (i.e. $0 < \sum_{i=0}^{n} l_i < 2000$). The total inter-bank liabilities is distributed randomly across all banks. The haircut multiplier, $\alpha$, is defined to be 0.50.

- Actions, $\Lambda$, are defined as all possible configurations of the joint action matrix $A \subset \mathbb{R}^{n \times n}$. The joint action matrix is normalized such that $\sum_i \Lambda_i = 1$.

- State transitions, $P(S^{t+1} \mid S^t, \Lambda)$, defines the deterministic transition from state $S^t$ to $S^{t+1}$ contingent on $\Lambda^t$.

- Rewards, $r$, is calculated under the *shared reward* scheme defined below as the change in the system value.

$$r = \sigma^{t+1} - \sigma^t$$

- Initial State, $p_0$, is defined probability that any state $s \in S$ is selected during the random initialization of the inter-bank network.

- Time-horizon, $T$, is defined as $T = 1$.

We motivate the choice of a one step time horizon noting that any set of actions, $\{a^t, a^{t+1}, ..., a^{t+k}\}$ can be captured by a single action $a \in \Lambda$. However, we recognize that longer time horizons may help the agent improve system value by moving into more recognizable states. We leave the study of longer time horizon to further research.

With regards to $\Lambda$, we opted for a continuous action space as it generalizes the distribution of a bank's capitalization. By normalizing each bank's action, $a_i \in \Lambda$ s.t. $\sum a_i = 1$, we represent the allocation decision of a bank $i$'s to itself and other banks $\{j | j \in X \ \ \& \ \ j \neq i\}$. This design was motivated by considering it to be the immediate rescue by recapitalization. We leave generalized link creation to future research which can be captured by introducing a new link creation action vector.

In the single-agent paradigm, the agent receives $S$ at every timestep $t$. As such, the agent has full-information and makes decisions given $C$ and $L$. Thus, the agent has adequate information to determine the set of defaulted banks, $\Omega^t$, and in response can form an appropriate joint-action $A$. After the environment receives $A$, the capitalization matrix is adjusted such that $c_i^{t+1} = \sum_{j=0}^{n} a_{ji} \cdot c_j$. Thus, the environment transitions to $S^{t+1} = (X^{t+1} = X^t, C^{t+1} = \{c_1^{t+1}, c_2^{t+1} ..., c_n^{t+1}\}, L^{t+1} = L^t)$.

Thus it is appropriate to design $r$ under a shared reward scheme to incentive the agent to maximize system value as it determines the decision of every bank $i \in X$.

## 3.3 Multi Agent Reinforcement Learning

Competing banks naturally naturally lends itself to the multi-agent reinforcement setting with profit-maximizing firms. Bank decision making is now contingent on individual incentives, history, and information. Actions that are optimal for the system as a whole may not ideal for an individual bank. (i.e. if the bank can acquire assets of another bankrupt firm in a sell-off).

Multi-agent environments are modeled as a decentralized partially observable Markov decision process (DEC-POMDP) [29] using a tuple $\mathbf{H} = (S, \mathit{\Lambda}, U, P, r, Z, O, n, \gamma)$ with which $S$ is the global state of the environment, $\mathit{\Lambda}$ the set of $n$ agents, $U$ the joint-action space, $P$ is the state transition function, $r$ is the reward function, $Z$ is the set of observations, and $\gamma \in [0, 1)$ is the discount factor. During every timestep $t$, each agent, $a \in \mathit{\Lambda}$, receives an individual observation $z_a \in Z$ according to the observation function $O(s, a) : S \times \mathit{\Lambda} \rightarrow Z$. Based on $z_a$, agents decide on an action $u_a$. Aggregated over all agents, the joint action vector $\mathbf{u} \in \mathbf{U} \equiv U^n$ is obtained and used by the the state transition

function $P : P(s' \mid s, \mathbf{u}) : S \times \mathbf{U} \times S \to [0, 1]$ which determines the next state of the environment given the current state and the joint actions. Each agent $a$ receives an individual reward from the environment from the reward function $r(s, \mathbf{u}) : S \times \mathbf{U} \to \mathbb{R}$. Agents collect their individual action-observation histories with which they learn a policy $\pi_a^t$ which seeks to maximize their discounted future reward $R_t = \sum_{i=0}^{\infty} \gamma^i r_{t+i}$.

In training multi-agent reinforcement learning agents, Tan [30]'s introduces independent Q-learning(IQL) applied a fully decentralized approach to learning by having each agent learn Q-values based on their individual observation history while treating other agents as part of the environment. By treating other agents as part of the environment, IQL does not have to learn having to learn in a large action space but faces problems arising from non-stationary learning where past observations may not be relevant for future decision making.

Kraemer et al. [5] introduces centralized training with decentralized execution(CTDE) for cooperative tasks which is motivated by the idea of rehearsals practices. [5] uses the example of stage performers, where during rehearsals the group is able to practice with additional equipment such as teleprompters, communicate with each other, take breaks, etc. During live performance, these practice tools are removed but the group is still able to perform with the knowledge they have learned. CTDE approaches are currently the SOTA in multi-agent reinforcement learning algorithms such as QTRAN[31], QMIX[32], VDN[33], COMA[34], MAAC[35]. The issue of non-stationarity is alleviated by allowing learning algorithms to share local action-observation histories, and additional information during training while restricting each agent's execution policy to their individual action-observation history.

### 3.3.1 MADDPG

MADDPG[36], is an actor critic approach which extends policy gradients into the multi-agent environment while using deep networks as function approximators. The actor is typically the component of the agent which determines the action to be taken whereas the critic is the component which evaluates the action taken by the actor.

Trained under the CTDE paradigm, MADDPG attempts to improve the predictability across agents to deal with the non-stationarity issue. During training, the critic is given access to information about the actions of other agents and returns a individual gradient used to update an agent's actor network. The centralized critic does not affect the agent during execution as the critic network is not used to determine an agent's actions.

### 3.3.2 Multi-agent Environment

We prepare the inter-bank network, $I$, into the multi-agent reinforcement learning setting, $\mathbf{H}$, by mappings as follows.

- States, $S$, is defined as all possible configurations of the inter-bank network $I$. Total inter-bank capitalization, $\sum_i c_i$, is defined as 1000 and is randomly distributed across all agents. Total inter-bank liabilities, $\sum_i L_i$, can take values such that 0 and 2000 and is randomly across all agents. The haircut multiplier, $\alpha$, is defined to be 0.50.

- Agents, $\Lambda$, are defined to be set with $n = 3$, 10, or 25. In our simulations, each agent is configured with the same network architecture presented in Section 4.2.

$$\Lambda = \{a_1, ..., a_n \mid n \in \{3, 10, 25\}\}$$

- Joint-actions, $U$, are defined as all possible configurations of the joint action matrix $A \subset \mathbb{R}^{n \times n}$. Each agent $a_i$ generates an action vector $u_i$ such that $\sum u_i = 1$. The environment aggregates the action vectors $\{u_1, u_2..., u_n\}$ to arrive at the joint action $U^t$.

- State transitions, $P(S^{t+1} \mid S^t, U)$, defines the deterministic transition from state $S^t$ to $S^{t+1}$ contingent on $U^t$.

- Rewards, $r$, are determined using two schemes - shared rewards or individual rewards. Shared rewards is defined as the change to the system value as follows:

$$r^s_{a_i}(t) = \sigma^{t+1} - \sigma^t$$

  Individual rewards is defined as the change in agent $a_i$'s individual net position $\omega_{a_i}$ from time $t$ to $t + 1$.

$$r^i_{a_i}(t) = \omega^{t+1}_i - \omega^t_i$$

- Observations, $Z$, is the defined to be all possible subsets of the set of states, $S$ (i.e. $Z = \{z \mid z \subset S\}$).

- Observation function, $O$, determines the observation observation $z$ is presented to agent $a_i$ at time $t$. Thus, agent $a_i$ receives the observation $O(s^t, a_i) = z^t_{a_i} = (c^t_{a_i}, L^t)$.

- Discount factor, $\gamma$, is defined to be 0.95. However, we define the time-horizon, $T$, such that $T = 1$. As such, the impact of the discount factor is negligible.

In the simulations, the configuration is generally similar between the single-agent and multi-agent environments. We point out and motivate the main points of differentiation (1) rewards and (2) observations.

We apply two reward schemes, shared and individual rewards, in the multi-agent environment. By applying a shared reward during learning, agents are induced to maximize the system value as a whole, or in other words, the health of the inter-bank network. However, competitive environments naturally lends itself to training agents under an individual reward system where individual returns drives decision making at a per-bank level.

In the multi-agent environment, agents achieve partial observability of the environment's state. Likewise in the real world, banks have access to their personal information technology systems, and only aggregate information reported by other banks (i.e. through financial reports or regulatory filings). Thus, agents are presented their individual capitalization and the liabilities matrix at every timestep.

Chapter 4

# Procedure and Results

We study how to train agents using multi-agent reinforcement learning such that agents can learn their behaviors based on their environment's dynamics. By doing so, this allows research to reduce the assumptions built into the agents. Section 4.1 discusses how the training and evaluation environments are instantiated. Section 4.2 discusses the deep network built into each agent and how agents are trained. Section 4.3 discusses the performance of the policy learned under single-agent reinforcement learning. Section 4.4 reports the performance obtained by agents trained under multi-agent reinforcement learning. Section 4.5 reports our findings and possible avenues for future research.

## 4.1 Environment

As per Chapter 2, a random instance of the inter-bank network is generated for training. We focus on inter-bank network that are structured as fully-connected graphs. During evaluation, we generate scenarios where at least one bank is in default.

Every instance of the inter-bank network is allowed a total capitalization of 1000 ($\sum_{i=1}^{n} c_i = 1000$) randomly distributed across each agent. The liabilities matrix is generated with the total inter-bank liability exposure taking a value between (0, 2000) ( $0 < \sum_{i=1}^{n} l_i < 2000$) that is randomly distributed across each agent. We define the haircut multiplier $\alpha$ to be 0.50.

During evaluation, the capitalization and liabilities matrix is similarly constructed. However, we include a check to ensure that at least one bank is facing default. Furthermore, we only allow instances where the total liabilities is less than bank capitalization as per Equation 4.1.

$$\sum_{i=1}^{n} c_i > \sum_{i=1}^{n} l_i \tag{4.1}$$

We determine the value of the system at time $t$ as per Section 2.5. Each

instance is ran for one time step, which allows agents to conduct one joint action. After which, each agent receives an individual reward signal depending on the incentive scheme.

Agents' learned policies are evaluated by averaging the system value over 100 instances of the evaluation environment. We collect the system value at instantiation and after adjusting the environment w.r.t. the joint action. The maximum system value for every evaluation instance is 1000 and as such sets a theoretical maximum to compare the impact of the agents' policies.

## 4.2 Training

In the single-agent environment, the agent is configured with two networks (1) actor (2) critic. The networks are defined as a multi-layer perceptron with two layers of sixty-four hidden nodes. A tanh activation is applied to the output layer.

With PPO, parallel instances of the environment are simulated simultaneously. Tuples of $(s^t, a, r, s^{t+1})$ are collected and when enough samples are collected, the networks is updated. We apply the default settings for PPO2 provided by stable-baselines [37].

In the multi-agent environment, each agent is configured with four networks (1) actor (2) critic (3) target actor (4) target critic. The actor and target actor network is defined as a multi-layer perceptron with four fully connected hidden layers and a tanh activation in the output layer. The critic and target critic network similarly uses four fully connected hidden layers with no activation in the output layer.

The actor and critic network are parameterized by $\theta_a$ and $\theta_c$ respectively. Similarly, the target actor and target critic are parameterized by $\theta_a^{'}$ and $\theta_c^{'}$ respectively.

During each time step, $t$, a buffer, $\mathbf{B}$, collects the tuple $(s^t, u, r, s^{t+1})$ and is used to alleviate the issue of high correlation between states.[23]

$$\mathbf{B} = \{(\mathbf{s}^t, \mathbf{u}^t, \mathbf{r}^t, \mathbf{s}^{t+1}) \; \forall \; t)\} \tag{4.2}$$

We define the batchsize, $b$, in our experiments to be 250. Thus, after $t > 250$, we take samples of size $b$ from the buffer to train each agent. Agents apply a soft update, parameterized by $\lambda$, during each training step as per Equation 4.3 in which the actor and critic networks moves towards the target actor and target critics.

$$\theta_a = (1 - \lambda) \cdot \theta_a^{'} + \lambda \cdot \theta_a \tag{4.3}$$

| Number of Banks | 3 | 10 | 25 |
|---|---|---|---|
| Average system value | 999.82 | 997.33 | 941.84 |
| Average initial value | 756.10 | 916.73 | 936.42 |

Table 4.1: Single agent performance - average net value of the system over 100 episodes.

| Number of Agents | 3 | 10 | 25 |
|---|---|---|---|
| Average under individual incentive | 815.44 | 906.19 | 902.67 |
| Average under system incentive | 940.13 | 899.65 | 924.34 |
| Average initial value | 756.10 | 916.73 | 936.42 |

Table 4.2: Multi agent performance - average net value of the system over 100 episodes.

$$\theta_c = (1 - \lambda) \cdot \theta_c^{'} + \lambda \cdot \theta_c \tag{4.4}$$

We define the critic loss to be the mean squared error in actual and predicted Q-values. The actor loss is defined to be the output of critic given $o_i$ and $u_i$.

## 4.3 Single Agent Performance

The simulations resulted in increases in the system value averaged across 100 evaluation instances. In the 3 bank simulation, on average the system value increased from 756.10 to 999.82. In the 10 bank simulation, on average the system value increased from 916.73 to 997.33. In the 25 bank simulation, on average the system value increased from 936.42 to 941.84. We note that while the system value increased in every scenario, the increase was less dramatic as the number of agents increased. Please refer to Table 4.1.

We note these results as reference points with which to analyse the multi-agent results.

## 4.4 Multi Agent Performance

With multi-agent reinforcement learning, in the 3 agent simulation the system value increased from an average of 756.10 to 940.13 and 815.44 under the system and individual incentive respectively. In the 10 agent simulation, the system value decreased from an average of 916.73 to 899.65 and 906.19 under the system and individual incentive respectively. In the 25 agent scenario, the system value decreased from 936.42 to 924.34 and 902.67 under the system and individual incentive respectively. Please refer to Table 4.2.

## 4.5 Discussion

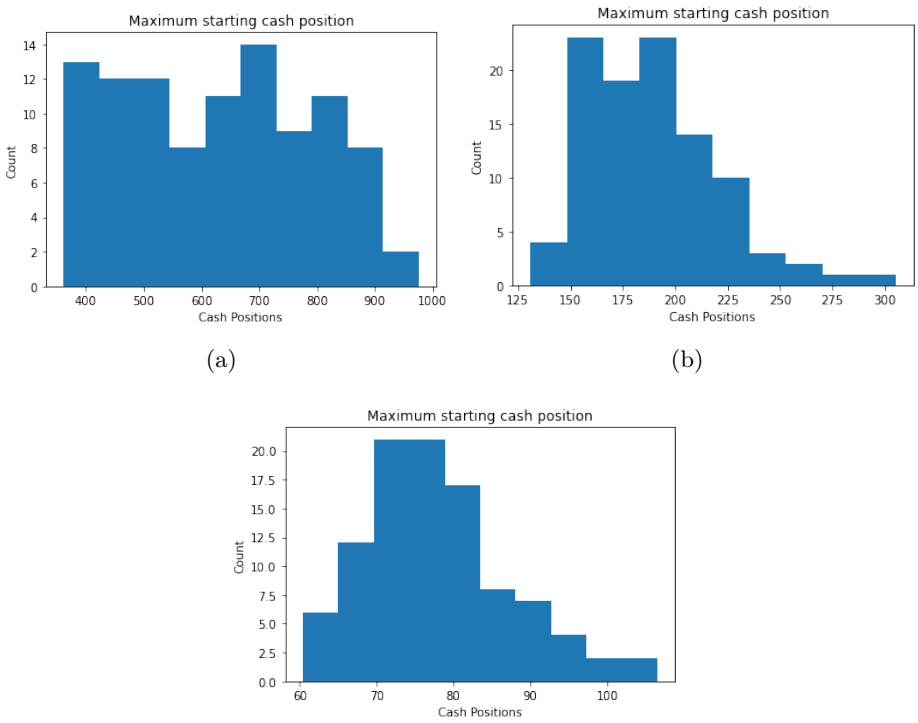



Figure 4.1: Histograms depicting the maximum capitalization allocated to a single agent when generating random instances of the financial environment. Figure 4.1 (a),(b),(c) depicts the 3, 10, and 25 agents simulations respectively. Note the higher concentrations in smaller agent simulations as the total inter-bank network capitalization, $C$, is distributed across fewer members.

We first review the distribution of capitalization, $C$, when instantiating an inter-bank network. We found in our simulations that inter-bank network capitalization were more concentrated in simulations with fewer agents. In our three agent simulation, a single agent could frequently receive over half of the total capitalization. Contrarily, in our twenty-five agent simulations, the capitalization was more evenly distributed. Please refer to Figure 4.1.

We next review the change in $\omega_i$, the net position of each agent. In three-agent simulation, we note that agents experienced a change in net position of over 90% of the total inter-bank network capitalization. In other words, the capitalization of the inter-bank network effectively transferred from one agent to another. The redistribution of capitalization becomes less extreme in simulations more agents. However, this may also be due to less concentrated capitalization

during environment instantiation.

We provide a speculation as to why agents may have learned this policy. That is, under shared rewards, the reward signal to each agent is the change in total system value. Thus, if the inter-bank capitalization is concentrated into one bank, then no haircut is applied in determining the system value. Further investigation is required to confirm this speculation.

Agents trained with individual rewards learn policies that results in lower concentration of total inter-bank capitalization. In the three-agent simulations, the maximum capitalization held by any agent averaged around the 70% as compared to 90% of inter-bank capitalization as depicted in Figure 4.3.

Overall, our empirical simulations shows that compared to single agent RL, multi-agent reinforcement learning's faces challenges in learning joint actions that maximizes a inter-bank network's system value. However, we note that this is reasonable considering the in-perfect information observed by each agent and coordination challenges in distributed execution. We note that one avenue for improving the agent's abilities is to introduce different networks in the actor/critic design such as graph neural networks which is able to represent aspects of a network's structure. Further research in this area can consider different environment design and initialization mechanisms, allowing for more general network configurations, and reward signals which takes into consideration the resulting inter-bank network concentrations. Another avenue of research is to introduce market mechanisms or reduce the action space available to agents. To create environments more reflective of real-world settings, instantiated environments can be generated that match network characteristics (i.e. degree, clustering, etc) as provided by real-world inter-bank datasets.

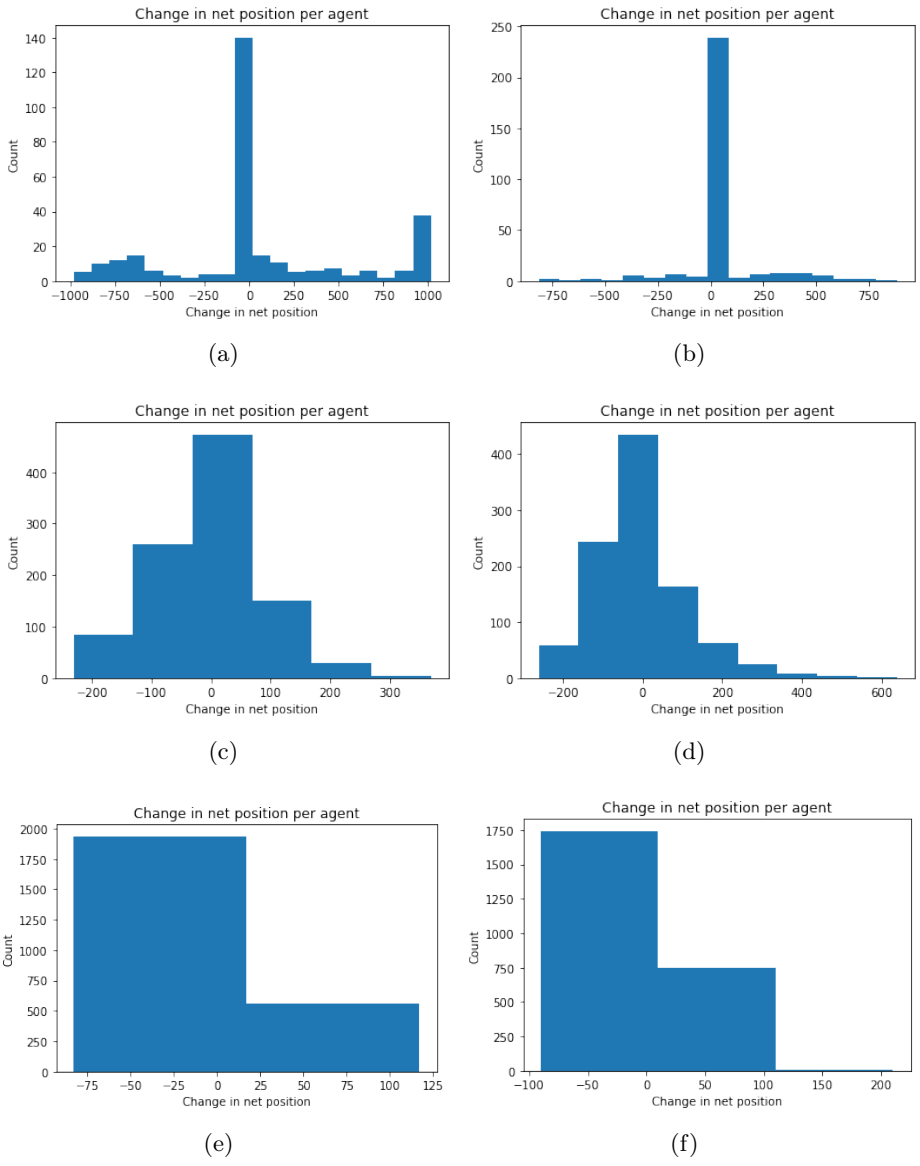
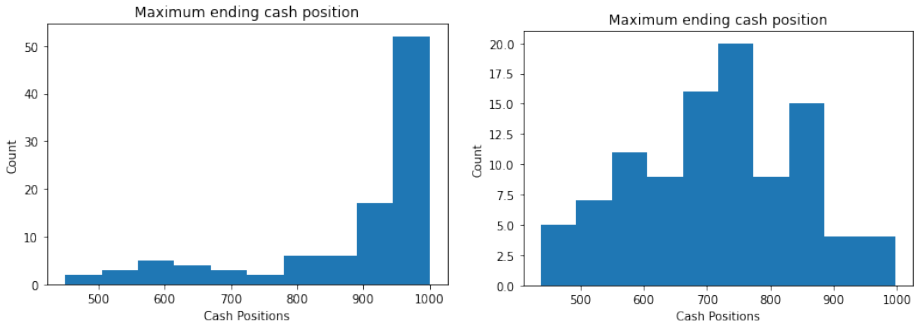



Figure 4.2: Histograms depicting change in net position per agent faced with shared rewards. We consider the change in net position to be a proxy capturing an agent's learned policy. (a, b) depicts the three-agent simulation. (c, d) depict ten-agent simulations. (e, f) depict twenty-five agent simulation. (a, c, e) were simulations with system rewards. (b, d, f) were simulations with individual rewards. Note in (a) that the learned policies exhibited behavior where an extreme change of net worth can occur. (c),(d),(e),(f) depict less extreme changes in net worth. However, this may be due to lower initial capitalization.




(a) (b)

Figure 4.3: Histograms depicting then ending capitalization in the three-agent simulation post-clearing. (a) depicts the concentrations under system rewards. (b) depicts the concentration under individual rewards. Note how the inter-bank capitalization is less concentrated under individual rewards.

Chapter 5

# Conclusion

We investigated the design of inter-bank network agents using multi-agent reinforcement learning under shared and individual reward schemes.

We found that agents trained with system rewards learned policies with unexpected behaviors in simulations with three agents. Specifically, agents learned behaviors that resulted in high concentration of total inter-bank capitalization, as well events where large amounts of net worth changed hands. However, the learned policies resulted in lower concentration of inter-bank capitalization as the number of agents in the simulation increased.

Under individual rewards, agents learned policies that resulted in lower concentrations of total inter-bank capitalization. Furthermore, agents exhibited behaviors that resulted in lower exchange of net worth.

The research of training inter-bank agents using multi-agent reinforcement learning remains an open question and many avenues are available to improve the policies learned by agents.

# Bibliography


[1] F. Allen and D. Gale, "Financial contagion," *Journal of political economy*, vol. 108, no. 1, pp. 1–33, 2000.

[2] O. De Bandt and M. Chahad, "A dgse model to assess the post-crisis regulation of universal banks," 2016.

[3] J.-Y. Gnabo and N. K. Scholtes, "Assessing the role of interbank network structure in business and financial cycle analysis," NBB Working Paper, Tech. Rep., 2016.

[4] G. Halaj, "Agent-based model of system-wide implications of funding risk," 2018.

[5] L. Kraemer and B. Banerjee, "Multi-agent reinforcement learning as a rehearsal for decentralized planning," *Neurocomputing*, vol. 190, pp. 82–94, 2016.

[6] A. Manchin, E. Abbasnejad, and A. van den Hengel, "Reinforcement learning with attention that works: A self-supervised approach," in *International Conference on Neural Information Processing*. Springer, 2019, pp. 223–230.

[7] A. Mott, D. Zoran, M. Chrzanowski, D. Wierstra, and D. J. Rezende, "Towards interpretable reinforcement learning using attention augmented agents," *arXiv preprint arXiv:1906.02500*, 2019.

[8] T. N. Kipf and M. Welling, "Semi-supervised classification with graph convolutional networks," *arXiv preprint arXiv:1609.02907*, 2016.

[9] F. Scarselli, M. Gori, A. C. Tsoi, M. Hagenbuchner, and G. Monfardini, "The graph neural network model," *IEEE transactions on neural networks*, vol. 20, no. 1, pp. 61–80, 2008.

[10] P. Veličković, G. Cucurull, A. Casanova, A. Romero, P. Lio, and Y. Bengio, "Graph attention networks," *arXiv preprint arXiv:1710.10903*, 2017.

[11] P. W. Battaglia, J. B. Hamrick, V. Bapst, A. Sanchez-Gonzalez, V. Zambaldi, M. Malinowski, A. Tacchetti, D. Raposo, A. Santoro, R. Faulkner *et al.*, "Relational inductive biases, deep learning, and graph networks," *arXiv preprint arXiv:1806.01261*, 2018.

[12] J. Leventides, M. Livada, and C. Poulios, *The Dynamics of Interbank Networks*. Cham: Springer International Publishing, 2020, pp. 369–395. [Online]. Available: https://doi.org/10.1007/978-3-030-55857-4_14

[13] X. Freixas, B. M. Parigi, and J.-C. Rochet, "Systemic risk, interbank relations, and liquidity provision by the central bank," *Journal of money, credit and banking*, pp. 611–638, 2000.

[14] L. Eisenberg and T. H. Noe, "Systemic risk in financial systems," *Management Science*, vol. 47, no. 2, pp. 236–249, 2001.

[15] C. H. Furfine, "Interbank exposures: Quantifying the risk of contagion," *Journal of money, credit and banking*, pp. 111–128, 2003.

[16] M. Kivelä, A. Arenas, M. Barthelemy, J. P. Gleeson, Y. Moreno, and M. A. Porter, "Multilayer networks," *Journal of complex networks*, vol. 2, no. 3, pp. 203–271, 2014.

[17] M. Montagna and C. Kok, "Multi-layered interbank model for assessing systemic risk," 2016.

[18] F. Allen, A. Babus, and E. Carletti, "Financial crises: theory and evidence," *Annu. Rev. Financ. Econ.*, vol. 1, no. 1, pp. 97–116, 2009.

[19] C. Upper, "Simulation methods to assess the danger of contagion in interbank markets," *Journal of Financial Stability*, vol. 7, no. 3, pp. 111–125, 2011.

[20] A.-C. Hüser, "Too interconnected to fail: A survey of the interbank networks literature," 2015.

[21] L. Bargigli and G. Tedeschi, "Interaction in agent-based economics: A survey on the network approach," *Physica A: Statistical Mechanics and its Applications*, vol. 399, pp. 1–15, 2014.

[22] L. C. Rogers and L. A. Veraart, "Failure and rescue in an interbank network," *Management Science*, vol. 59, no. 4, pp. 882–898, 2013.

[23] V. Mnih, K. Kavukcuoglu, D. Silver, A. A. Rusu, J. Veness, M. G. Bellemare, A. Graves, M. Riedmiller, A. K. Fidjeland, G. Ostrovski *et al.*, "Human-level control through deep reinforcement learning," *nature*, vol. 518, no. 7540, pp. 529–533, 2015.

[24] D. Silver, A. Huang, C. J. Maddison, A. Guez, L. Sifre, G. Van Den Driessche, J. Schrittwieser, I. Antonoglou, V. Panneershelvam, M. Lanctot *et al.*, "Mastering the game of go with deep neural networks and tree search," *nature*, vol. 529, no. 7587, pp. 484–489, 2016.

[25] R. S. Sutton and A. G. Barto, *Reinforcement learning: An introduction*. MIT press, 2018.

[26] R. Chernow, *The house of Morgan: An American banking dynasty and the rise of modern finance*. Grove/Atlantic, Inc., 2010.

[27] B. Herzog, "Timothy f. geither" stress test: reflections on financial crises"," 2015.

[28] J. Schulman, F. Wolski, P. Dhariwal, A. Radford, and O. Klimov, "Proximal policy optimization algorithms," *arXiv preprint arXiv:1707.06347*, 2017.

[29] F. A. Oliehoek and C. Amato, *A concise introduction to decentralized POMDPs*. Springer, 2016.

[30] M. Tan, "Multi-agent reinforcement learning: Independent vs. cooperative agents," in *Proceedings of the tenth international conference on machine learning*, 1993, pp. 330–337.

[31] K. Son, D. Kim, W. J. Kang, D. E. Hostallero, and Y. Yi, "Qtran: Learning to factorize with transformation for cooperative multi-agent reinforcement learning," in *International Conference on Machine Learning*. PMLR, 2019, pp. 5887–5896.

[32] T. Rashid, M. Samvelyan, C. Schroeder, G. Farquhar, J. Foerster, and S. Whiteson, "Qmix: Monotonic value function factorisation for deep multi-agent reinforcement learning," in *International Conference on Machine Learning*. PMLR, 2018, pp. 4295–4304.

[33] P. Sunehag, G. Lever, A. Gruslys, W. M. Czarnecki, V. Zambaldi, M. Jaderberg, M. Lanctot, N. Sonnerat, J. Z. Leibo, K. Tuyls *et al.*, "Value-decomposition networks for cooperative multi-agent learning," *arXiv preprint arXiv:1706.05296*, 2017.

[34] J. Foerster, G. Farquhar, T. Afouras, N. Nardelli, and S. Whiteson, "Counterfactual multi-agent policy gradients," in *Proceedings of the AAAI Conference on Artificial Intelligence*, vol. 32, no. 1, 2018.

[35] S. Iqbal and F. Sha, "Actor-attention-critic for multi-agent reinforcement learning," in *International Conference on Machine Learning*. PMLR, 2019, pp. 2961–2970.

[36] R. Lowe, Y. Wu, A. Tamar, J. Harb, P. Abbeel, and I. Mordatch, "Multi-agent actor-critic for mixed cooperative-competitive environments," *arXiv preprint arXiv:1706.02275*, 2017.

[37] A. Hill, A. Raffin, M. Ernestus, A. Gleave, A. Kanervisto, R. Traore, P. Dhariwal, C. Hesse, O. Klimov, A. Nichol, M. Plappert, A. Radford, J. Schulman, S. Sidor, and Y. Wu, "Stable baselines," https://github.com/hill-a/stable-baselines, 2018.

[38] J. Schulman, S. Levine, P. Abbeel, M. Jordan, and P. Moritz, "Trust region policy optimization," in *International conference on machine learning*. PMLR, 2015, pp. 1889–1897.

[39] S. Kakade and J. Langford, "Approximately optimal approximate reinforcement learning," in *In Proc. 19th International Conference on Machine Learning*. Citeseer, 2002.

[40] V. Mnih, A. P. Badia, M. Mirza, A. Graves, T. Lillicrap, T. Harley, D. Silver, and K. Kavukcuoglu, "Asynchronous methods for deep reinforcement learning," in *International conference on machine learning*. PMLR, 2016, pp. 1928–1937.

Appendix A

# PPO

Schulman et al. proposes a objective function which considers jointly (1) the policy surrogate (2) value function error term (3) and an entropy bonus to encourage exploration[28] as per Equation A.1.

$$L_t^{CLIP+VF+S}(\theta) = \hat{\mathbb{E}}_t[L_t^{CLIP}(\theta) - c_1 L_t^{VF}(\theta) + c_2 S[\pi_\theta](s_t)] \tag{A.1}$$

The policy surrogate builds on the work of [38, 39] which attempts to maximize an surrogate objective function while constraining the size of policy updates as per Equation A.6. Kakade et al. [39] introduced the conservative policy iteration objective as per Equation A.2.

$$L(\theta) = \hat{\mathbb{E}}_t\left[\frac{\pi_\theta(a_t|s_t)}{\pi_{\theta_{old}}(a_t|s_t)}\hat{A}_t\right] = \hat{\mathbb{E}}_t\left[r_t(\theta)\hat{A}_t\right] \tag{A.2}$$

$$r_t(\theta) = \frac{\pi_\theta(a_t|s_t)}{\pi_{\theta_{old}}(a_t|s_t)} \tag{A.3}$$

However, this would lead to large policy updates which translated to large changes in the policy between updates. As such, [38] restrictions the size of policy updates with constraints based on the KL divergence as per Equation A.4.

$$\max_\theta \hat{E}_t\left[\frac{\pi_\theta(a_t|s_t)}{\pi_{\theta_{old}}(a_t|s_t)}\hat{A}_t\right] \tag{A.4}$$

$$\text{subject to } \hat{\mathbb{E}}_t[KL[\pi_{\theta_{old}}(\cdot|s_t), \pi_\theta(\cdot|s_t)]] \leq \omega \tag{A.5}$$

In PPO[28] the benefits of constraining policy updates is achieved by clipping updates within $\epsilon$ as per Equation A.6.

$$L^{CLIP}(\theta) = \hat{\mathbb{E}}_t[min(r_t(\theta)\hat{A}_t, clip(r_t(\theta), 1-\epsilon, 1+\epsilon)\hat{A}_t] \tag{A.6}$$

The remaining term of PPO's objective function - the value function error term $L_t^{VF}(\theta)$ is defined as the squared error loss as per Equation A.7, the entropy bonus, $S[\pi_\theta](s_t)$ being a entropy factor to encourage exploration, and $c_1$ $c_2$ being coefficients.

$$(V_\theta(s_t) - V_t^{targ})^2 \tag{A.7}$$

The advantage value $\hat{A}_t$ is computed as per Equation A.8 which looks at a T length trajectory starting at t$\in [0, T]$[40].

$$\begin{aligned} \hat{A}_t = \omega_t + (\gamma\lambda)\omega_{t+1} + ... + (\gamma\lambda)^{T-t+1}\omega_{T-1} \\ \text{where } \omega_t = r_t + \gamma V(s_{t+1}) - V(s_t) \end{aligned} \tag{A.8}$$

Thus we have PPO as per Algorithm 2 which takes advantage of having N actors running in parallel collecting T time steps of observations. Across the NT observations, a minibatch of size M<NT is selected and used to calculate the advantage function $\hat{A}_t$, and train the network as per Equation A.1.

Appendix B

# MADDPG

In order to alleviate the highly correlated nature of observations, an experience replay buffer $D$ is applied which stores the experiences of all agents in the tuple $(\mathbf{x}, \mathbf{x}', \mathbf{u}, \mathbf{r})$ and is sampled from during training.

In MADDPG, each agent receives an individual observations $o_a$ and learns a policy $\pi_a$ which is used to determine $u_a$. Furthermore, $Q_a^\pi(\mathbf{x}, \mathbf{u})$ is a centralized action-value function that receives the actions of all agents $u_a \in \mathbf{U}$ as well as additional state information $\mathbf{x}$ and outputs the Q-value for agent $a$. In the continuous context, each of the $n$ agents' policies are parameterized by $\mu_{\theta_a}$. As such, the gradient of the expected return for agent $a$ is thus as per Equation B.1 and the update rule as per Equation B.2 with $\boldsymbol{\mu}' = \{\mu_{\theta'_a}\}$ being the delayed target policies.

$$\nabla_{\theta_a} J(\mu_a) = \mathbb{E}_{x,a\sim D}[\nabla_{\theta_a} \mu_a(u_a|o_a) \nabla_{u_a} Q_a^\mu(x, \mathbf{u})] \tag{B.1}$$

$$\mathcal{L}(\theta_a) = \mathbb{E}_{\mathbf{x},a,r,\mathbf{x}'}[(Q_i^\mu(\mathbf{x}, \mathbf{u}) - y)^2] \tag{B.2}$$

$$y = r_i + \gamma Q_i^{\mu'}(\mathbf{x}', \mathbf{u}')\Big|_{\mathbf{u}'_\mathbf{a}=\boldsymbol{\mu}'_a(o_a)} \tag{B.3}$$