Design of artificial intelligence agent for supply chain management using deep reinforcement learning based on NegMAS Library

Masterarbeit

KIT – Karlsruher Institut für Technologie
Fraunhofer IOSB – Fraunhofer-Institut für Optronik, Systemtechnik und Bildauswertung

Ning Yue

10. Mai 2021

Verantwortlicher Betreuer: Prof. Dr.-Ing. habil. Jürgen Beyerer
Betreuende Mitarbeiter: Dr.-Ing. Tim Zander
Prof. Dr.-Ing. Yasser Mohammad(Extern)

Erklärung der Selbstständigkeit

Hiermit versichere ich, dass ich die Arbeit selbständig verfasst habe und keine anderen als die angegebenen Quellen und Hilfsmittel benutzt habe, die wörtlich oder inhaltlich übernommenen Stellen als solche kenntlich gemacht habe und die Satzung des Karlsruher Instituts für Technologie zur Sicherung guter wissenschaftlicher Praxis in der gültigen Fassung beachtet habe.

Karlsruhe, den 10. Mai 2021

(Ning Yue)

Abstract

Consider an agent that can cooperate with others and autonomously negotiate, to reach an agreement. These agents could achieve great score, e.g. profitability of factory in realistic. Such agent is practice in complex, realistic environment such as supply chain management. In experiments come from this paper, learnable agents are proposed with Multi-Agent deep reinforcement learning method(QMIX and MADDPG) to achieve the goal. The strategy of agent will be improved when continuously interacting with others, central training but executing with local observation. The learned strategy enable autonomous agents to negotiate concurrently with multiple different type, unknow opponents in real-time, over complex multi-issues.

Kurzfassung

Falls die Abschlussarbeit auf Deutsch geschrieben wird, genügt die deutsche Kurzfassung.

Notation

General identifier

$\alpha, \ldots, \omega$	Skalare
$a, \ldots, z$	Skalar, Vektor, Funktionssymbol (oder Realisierung einer Zufallsvariablen)
$\boldsymbol{a}, \ldots, \boldsymbol{z}$	Zufallsvariable (skalar bzw. vektoriell)
$\hat{\boldsymbol{a}}, \ldots, \hat{\boldsymbol{z}}$	Schätzer für jeweilige Variable als Zufallsgröße
$\hat{a}, \ldots, \hat{z}$	Realisierter Schätzer für jeweilige Variable
$A, \ldots, Z$	Matrix
$\boldsymbol{A}, \ldots, \boldsymbol{Z}$	Matrix als Zufallsgröße
$\mathcal{A}, \ldots, \mathcal{Z}$	Menge
$\mathfrak{A}, \ldots, \mathfrak{Z}$	Mengensystem

Special identifier

t	Spezielle Bezeichner mit konkreter Bedeutung in dieser Arbeit, z. B. t Zeitindex

General quantities

$\mathbb{C}$	Menge der komplexen Zahlen
$\mathbb{H}$	Poincaré Halbebene
$\mathbb{N}$	Menge der natürliche Zahlen (ohne Null)
$\mathbb{N}_0$	Menge der natürliche Zahlen mit Null
$\mathbb{Q}$	Menge der rationalen Zahlen
$\mathbb{Q}^{>0}$, $\mathbb{Q}^{<0}$	Menge der positiven bzw. negative rationalen Zahlen
$\mathbb{R}$	Menge der reellen Zahlen
$\mathbb{R}^{>0}$, $\mathbb{R}^{<0}$	Menge der positiven bzw. negative reellen Zahlen
$\mathbb{Z}$	Menge der ganzen Zahlen

Special symbols

$\mathfrak{N}(\mu, \sigma^2)$	Normalverteilung mit Erwartungswert μ und Varianz σ
$\mathfrak{F}_{r,s}$	Fisher-Verteilung mir r Zähler- und s Nennerfreiheitsgraden
$\mathfrak{t}_s$	Student-t-Verteilung mit s Freiheitsgraden
δ_ξ	Ein-Punkt-Maß an der Stelle ξ
χ^2_s	χ^2-Verteilung mit s Freiheitsgraden

Contents

1 Introduction

A supply chain is a network of suppliers, factories, warehouses, distribution centers and retailers, through which raw materials are acquired, transformed, produced and delivered to the Customer[]. In the network we could find many entities whose could be considered as agents in the Multi-agent systems (MAS). Multi-agent System(MAS) are suitable…, there are many approaches are proposed in order to solve the problem in the supply chain mangement system. Such as negotiation-based Multi-agent System[].

The Supply Chain Management (SCM) world designed in SCML by `Yasser Mohammad` simulates a supply chain consisting of multiple factories that buy and sell products from one another. The factories are represented by autonomous agents that act as factory managers. Agents are given some target quantity to either buy or sell and they negotiate with other agents to secure the needed supplies or sales. Their goal is to turn a profit, and the agent with the highest profit (averaged over multiple simulations) wins [Moh+19]. It is characterized by profit-maximizing agents that inhabit a complex, dynamic, negotiation environment[]. There are two games built on the top of NegMAS which is the library for developing autonomous negotiation agents embedded in simulation environments. .....

1.0.1 Motivation

Negotiation is a complex problem, in which the variety of settings and opponents that may be encountered prohibits the use of a single predefined negotiation strategy. Hence the agent should be able to learn such a strategy autonomously[]. The development of current machine learning algorithms and increased hardware resource make it possible, model the realistic environment to evaluate the problem with computer system. According to the modeled realistic environment, it will be easier to find more possible solutions.

In this work, we use some modeled negotiation environments, such as single agent environment(bilateral negotiation), and analyze whether deep reinforcement learning can be used to let agent learns some strategies autonomously in these environments. In contrast to single agent environment, in the supply chain environment, there are many agents with the same goal. After analyzing the simple environment, we need to explore whether multi-agent deep

reinforcement learning can be used to obtain better results in multi-agent environment.

How good strategy can be learned by deep reinforcement learning in single agent environment(bilateral negotiation)?

How good strategy can be learned by multi-agent deep reinforcement learning in multi-agent environment(concurrent negotiation)?

What is the difference between deep reinforcement learning strategies and other heuristic strategies?

1.0.2 Outline of this Work

In the following, the other chapters of this work are listed and their content briefly presented.

Chapter 2: Background: This chapter contains basic knowledge and concepts that are necessary to understand the thesis. Firstly, some concepts from game theory are listed. These concepts are often discussed and used in autonomous negotiation. Secondly, utility function, some negotiation mechanisms are described in the section on autonomous negotiation. In addition, the basics and the historical development of artificial intelligence are presented. The focus of this chapter is on reinforcement learning.

Chapter 3: Related Work

Chapter 4: Analyze

Chapter 5: Methods and Experiments

Chapter 6: Conclusions and Future Work

2 Background

2.1 Game theory

2.1.1 Nash Equilibrium

The concept of a Nash equilibrium plays a central role in noncooperative game theory[]. The definition in simple setting of a finite player is described as follow. I players indexed by i=1,...,I. The strategy of agent i is si choosed from Ni pure strategies. A strategy profile of all agents written as s = (s1, s2, sI), si|si' for the strategy profile (s1, ... si-1, si+1, sI), or the s with the part of i changed from si to si'. For each player i and strategy s, ui(s) denotes i's expected utility[].

Proposition 2.1 (Nash Equilibrium) *For each agent i and si', $u_i(s) >= u_i(s|s_i')$*

In terms of words description, the definition of Nash equilibrium is that if other agents do not change its strategy, then no single agent can obtain higher utility.

The learning process of RL-Agent in this paper can be considered as an incomplete information static non-cooperative game.

2.1.2 Pareto Efficient

Proposition 2.2 (Pareto Efficient) *no other feasible allocation $\{x_1',...,x_n'\}$ where, for utility function u_i for each agent i, $u_i(x_i)$ for all $i \in \{1,...,n\}$ with $u_i(x_i') > u_i(x_i)$ for some i[].*

Pareto Efficient is a state at which resources in a system are optimized in a way that one dimension cannot improve without a second worsening.

2.1.3 Markov Games

Methods mentioned in the paper are based on a multi-agent extension of Markov decision processes(MDPs) called partially observable Markov games. There are N players indexed by $n = 1,2,...,N$.

2.2 Autonomous Negotiaion

Negotiation is an important process in coordinating behavior and represents a principal topic in the field of multi-agent system research. There has been extensive research in the area of automated negotiating agents.

Automated agents can be used side-by-side with a human negotiator embarking on an important negotiation task. They can alleviate some of the effort required of people during negotiations and also assist people who are less qualified in the negotiation process. There may even be situations in which automated negotiators can replace the human negotiators. Thus, success in developing an automated agent with negotiation capabilities has great advantages and implications[].

Through the negotiation agents, many problems that arise in real or simulated domain can be solved. In industrial domains, In commerical domains, the Supply Chain Management System (SCMS) functionality is implemented through agent-based negotiation environment, in which contracts can be singed through negotiation between agents. Many papers describe ongoing effort in developing a Multi-agent System (MAS) for supply chain management[SCML].

In game domains, bilateral negotiation in [GENIUS]

2.2.1 Utility Function

Utility function is an important concept in economics. It measures preferences over a set of goods and services. Utility represents the satisfaction that consumers receive for choosing and consuming a product or service[]. In NegMAS and SCML, utility function could measure either single offer or set of offers.

Utility is measured in units called utils, but calculating the benefit or satisfaction that consumers receive from is abstract and difficult to pinpoint[]. In the package NegMAS, SCML are built-in some utility functions, through inheritance of these it is easy to design new Utility function by developer. Such as linear utiliy function and real utility function OneShotUfun designed in SCMLOneShotWorld.

[1] linear utility function [2] real utility function OneShotUfun

It is a important point for designing a new Agent in autonomous negotiation environments. For heuristic agents utility function is a keypoint to measure preferences. For reinforment learning agents utiliy function conducts the behavior of learnable agents, used as a part of reward function, significantly affect the design and evaluation of RL-Agent.

2.2.2 Rubinstein bargaining mechanism

Rubinstein bargaining mechanism is widely cited for multi-round bilateral negotiation [Rub82]. Two agents in the mechanism which has an infinite time horizon have to reach an agreement.

To the state of nash equilibrium.

2.2.3 Stacked alternating offers mechanism(SAOM)

SAOM is also named as stacked alternating offers protocol. Agents can only take their action when it is their turn. SAOM allows negotiating agents to evaluate only the most recent offer in their turn and accordingly they can either accept offer, make a count offer or walk away.

In the SCML OneShotWord, at the first step all of the agents will propose a first offer.

2.3 Artificial Intelligence

Artificial Intelligence is a broad branch of computer science that is focused on a machine's capability to produce rational behavior from external inputs. The goal of AI is to create systems that can perform tasks that would otherwise require human intelligence[].

There is a set of three related items that sometimes are erroneously used interchangeably, namely artificial intelligence, machine learning, and neural networks. According to Encyclopaedia Britannica, AI defines the ability of a digital computer or computer-controlled robot to perform tasks commonly associated with intelligent beings. On the other hand, according to H.A. Simon, one of the pioneers of the field, machine learning is a "field of study that gives computers the ability to learn without being explicitly programmed"

2.3.1 Sub-areas

Fig. 2.1 shows the relationship of artificial intelligence, machine learning and deep learning.

Artificial Intelligence

Artificial intelligence, also called machine intelligence, can be understood by an intelligence, unlike the natural intelligence shown by humans and animals, which is demonstrated by machines. It looks at ways of designing intelligent devices and systems that can address problems creatively that are often treated as a human prerogative. Thus, AI means that a machine somehow imitates human behavior.

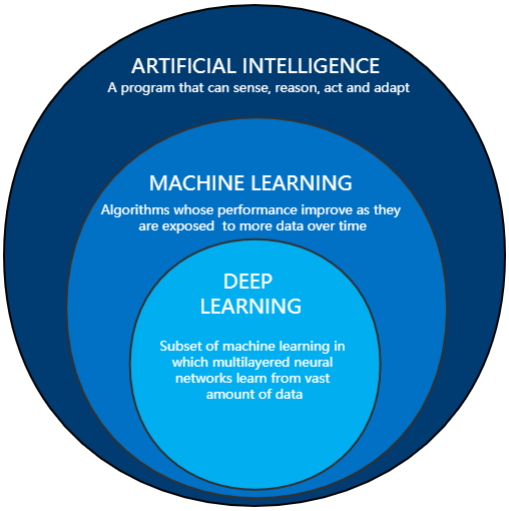

Figure 2.1: Sub-areas of artificial intelligence [SUM20]

Machine Learning

Machine learning is an AI subset and consists of techniques that enable computers to recognize data and supply AI applications. Different algorithms (e.g., neural networks) contribute to problem resolution in ML.

Deep Learning

Deep learning, often called deep neural learning or deep neural network, is a subset of machine learning that uses neural networks to evaluate various factors with a similar framework to a human neural system. It has networks that can learn from unstructured or unlabeled data without supervision.

2.3.2 Methods

Supervised Learning:

Unsupervised Learning:

Semi-supervised Learning:

Reinforcement Learning:

2.3.3 Application Field

eCommerce

Logistics and Supply Chain

Artificial intelligence (AI) researchers have paid a great deal of attention to automated negotiation over the past decade and a number of prominent models have been proposed in the literature.

Tools for computer science

2.4 Artificial Neural Network

2.5 Reinforcement Learning

2.5.1 Q-Learning

$$\mathcal{L}(\theta) = \sum_{i=1}^{b} \left[(y_i - Q(s, a \mid \theta))^2 \right] \tag{2.1}$$

2.5.2 Policy Gradient PG

$$\nabla_\theta J(\theta) = \mathbb{E}_{s \sim p^\pi, a \sim \pi_\theta} \left[\nabla_\theta \log \boldsymbol{\pi}_\theta(a \mid s) Q^{\boldsymbol{\pi}}(s, a) \right] \tag{2.2}$$

2.5.3 Deep Reinforcement Learning (DRL)

Deep Q-Networks

Proximal Policy Optimization (PPO)

Disadvantage: the distrubition of action changed too quickly, when the reward is always positiv or negative, some possible action will be disapper. PPO use some contraint tricks to avoid it.

such as clip of policy. The loss function based on PPO-clip is as follow.

$$L\left(s, a, \theta_k, \theta\right) = \min\left(\frac{\pi_\theta(a \mid s)}{\pi_{\theta_k}(a \mid s)} A^{\pi_{\theta_k}}(s, a), \quad \operatorname{clip}\left(\frac{\pi_\theta(a \mid s)}{\pi_{\theta_k}(a \mid s)}, 1-\varepsilon, 1+\varepsilon\right) A^{\pi_{\theta_k}}(s, a)\right) \tag{2.3}$$

DPG, DDPG, MADDPG

DPG: create a μ function to determine the action instead the sample in PG.

DDPG: use Actor-Critic model to create target and execute network. a variant of DPG. policy μ and critic Q_μ are approximated with deep neural networks.

MADDPG: DDPG used in multi-agents environments.

MADPG

$$\nabla_{\theta_i} J\left(\theta_i\right) = \mathbb{E}_{s \sim p^\mu, a_i \sim \pi_i}\left[\nabla_{\theta_i} \log \pi_i\left(a_i \mid o_i\right) Q_i^\pi\left(\mathbf{x}, a_1, \ldots, a_N\right)\right] \tag{2.4}$$

Extension to deterministic policies

$$\nabla_{\theta_i} J\left(\boldsymbol{\mu}_i\right) = \mathbb{E}_{\mathbf{x}, a \sim \mathcal{D}}\left[\nabla_{\theta_i} \boldsymbol{\mu}_i\left(a_i \mid o_i\right) \nabla_{a_i} Q_i^\mu\left(\mathbf{x}, a_1, \ldots, a_N\right)\Big|_{a_i = \boldsymbol{\mu}_i\left(o_i\right)}\right] \tag{2.5}$$

Loss function is

$$\mathcal{L}\left(\theta_i\right) = \mathbb{E}_{\mathbf{x}, a, r, \mathbf{x}'}\left[\left(Q_i^{\boldsymbol{\mu}}\left(\mathbf{x}, a_1, \ldots, a_N\right) - y\right)^2\right], \quad y = r_i + \gamma Q_i^{\boldsymbol{\mu}'}\left(\mathbf{x}', a_1', \ldots, a_N'\right)\Big|_{a_j' = \boldsymbol{\mu}_j'\left(o_j\right)} \tag{2.6}$$

Value Decomposition Networks

Mixing Network used in QMIX

QMIX

2.6 Platform and Library

2.6.1 GENIUS

GENIUS: An integrated environment for supporting the design of generic automated negotiators [Lin+14].

2.6.2 NegMAS

NegMAS can model situated simultaneous negotiations such as SCM. Nevertheless, it can model simpler bilateral and multi-lateral negotiations.

NegMAS is a python library for developing autonomous negotiation agents embedded in simulation environments. The name negmas stands for either NEGotiation MultiAgent System

or NEGotiations Managed by Agent Simulations. The main goal of NegMAS is to advance the state of the art in situated simultaneous negotiations. Nevertheless, it can; and is being used; for modeling simpler bilateral and multi-lateral negotiations, preference elicitation , etc.

NegMAS and Bilateral Negotiation NegMAS implements natively SAOM which can be set as bilateral negotiation mechanism.

2.6.3 SCML

A supply chain is a sequence of processes by which raw materials are converted into finished goods. A supply chain is usually managed by multiple independent entities, whose coordination is called **supply chain management(SCM)**. SCM exemplifies situated negotiation. The SCM world was built on top of an opensource automated negotiation platform called NegMAS to serve as a common benchmark environment for the study of situated negotiation [Moh+19]. This repository is the official platform for running ANAC Supply Chain Management Leagues. It will contain a package called scmlXXXX for the competition run in year XXXX. For example scml2019 will contain all files related to the 2019's version of the competition. There are three main different versions of SCML, which have different designs.

- SCML2020-OneShot
- SCML2020/2021
- SCML2019

SCML and Concurrent Bilateral Negotiations

SCML was originally developed as a part of NegMAS, from the version ? it was splited as an independent project to research SCM. SCML realized a SCM World to simulate the SCM process.

There are many agents which has same type in the SCM World.

Researchers have also developed many negotiation agents such as Agent1[], Agent2[], Agent3[] in GENIUS, Agent4[], Agent5[], Agent6[] in NegMAS.

2.6.4 PyTorch

PyTorch is an open source machine learning library and framework which performs immediate execution of dynamic tensor computations with automatic differentiation and GPU acceleration, and does so while maintaining performance comparable to the fastest current libraries for deep learning. [Pas+19]. While considering performance, it is also easier to apply and debug.

2.6.5 OpenAI Gym

OpenAI Gym is a toolkit for developing and comparing reinforcement learning algorithms [Bro+16].

Environment

The core gym interface is Env, which is the unified environment interface. The following are the Env methods that developers should implement [Pas+19].

STEP: The

RESET:

RENDER:

CLOSE:

SEED:

Stable Baselines

The stable baselines developed in the project stable-baselines [Hil+18]. All implemented algorithms with characteristic discrete/continuous actions are shown in 2.1.

Name	Box	Discrete
A2C	Yes	Yes
ACER	No	Yes
ACKTR	Yes	Yes
DDPG	Yes	No
DQN	No	Yes
HER	Yes	Yes
GAIL	Yes	Yes
PPO1	Yes	Yes
PPO2	Yes	Yes
SAC	Yes	No
TD3	Yes	No
TRPO	Yes	Yes

Table 2.1: stable baselines algorithms

2.6.6 Ray

Ray is packaged with the following libraries for accelerating machine learning workloads.

- Tune: Scalable Hyperparameter Tuning
- RLlib: Scalable Reinforcement Learning
- RaySGD: Distributed Training Wrappers
- Ray Serve: Scalable and Programmable Serving

3 Related Work

In this chapter, related work on the relevant topics of this work is presented and discussed. The topics include autonomous negotiation, multi-agent reinforcement learning. At the last section, the work of reinforcement learning used in autonomous negotiation is presented.

3.1 Heuristic Negotiation Strategies for Autonomous Negotiation

3.1.1 Time-based Strategy (Aspiration Negotiator)

Type of aspiration is boulware.

3.1.2 Concurrent Negotiation Strategy (CNS)

In a concurrent negotiation environment, an agent will negotiate with many opponents at the same time(one-to-many). One issue is how to coordinate all these negotiations. The author of the paper [Wil+12] designed an intuitive model with two key parts, namely the `Coordinator` and `Negotiation Thread`.

Negotiation Threads: The strategy of each negotiation thread is an extension of a recently published, principled, adaptive bilateral negotiation agent. This agent was designed to be used in a similarly complex environment, but only for negotiations against a single opponent.

Coordinator: The role of the coordinator is to calculate the best time, t_i and utility value, u_i at that time, for each thread. To do so, it uses the probability distributions received from the individual threads, which predict future utilities offered by the opponents.

3.1.3 Conclusion

From the analysis of the heuristic negotiation strategy in a specific field, we can get some important parameters, such as time, offer by opponent, that need to be considered as the information used in the RL algorithm.

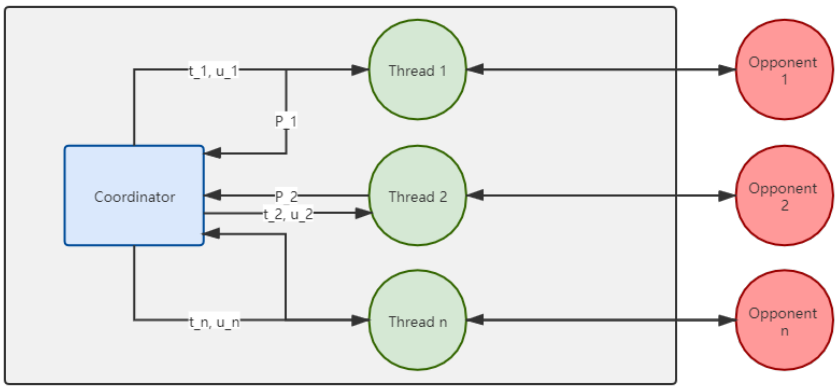

Figure 3.1: Architecture of the concurrent negotiation agent, best time: t_i and utility value: u_i, probability distributions: P [Wil+12].

3.2 Reinforcement Learning used in Autonomous Negotiation

NegoSI: A novel algorithm named negotiation-based MARL with sparse interactions (NegoSI) is presented by Luowei Zhou. In contrast to traditional sparse-interaction based MARL algorithms, NegoSI adopts the equilibrium concept and makes it possible for agents to select the non-strict Equilibrium Dominating Strategy Profile (non-strict EDSP) or Meta equilibrium for their joint actions [Zho+17].

RLBOA: From the paper [Bak+19] A Modular Reinforcement Learning Framework for Autonomous Negotiating Agents.

ANEGMA: Work by [Bag+20]. A novel DRL-inspired agent model called ANEGMA, which allows the buyer to develop an adaptive strategy to effectively use against its opponents (which use fixed-but-unknown strategies) during concurrent negotiations in an environment with incomplete information .

3.3 Challenges in Deep Reinforcement Learning

3.3.1 Sparse Reward

Utility value as part of reward function reward shaping Curiosity Driven Imitation Learning

3.3.2 Non-stational environment

The strategy of single agent is changed during training Multi-agent environment is non-statinal Multi-agent deep reinforcement learning

3.3.3 Huge action space

Action embedding, discrete action replaced by continious action space.

4 Analyze

4.1 Environments

Two environments are developed for comparing the DRL algorithms used in this thesis: single-agent bilateral negotiation environment (SBE) and multi-agent concurrent bilateral negotiation environment (MCBE). The details are described in section 4.2.3 and 4.3.1. In addition to these environments, some methods have been implemented to make the training logic clearer, such as `Game` in section 4.2.4 and `Scenario` in section 4.3.2.

4.2 NegMAS with OpenAI Gym

NegMAS has implemented some negotiation mechanisms and specific simulated world, such as SAOM and SCML (Now as an independent project). In order to compare the algorithms in specific simulated world more easily, an interface is needed to connect NegMAS and RL algorithms. This interface and all algorithms can be rewritten from scratch, but it is very time-consuming and not ideal. The second option is to implement some RL framework interfaces, which will reduce a lot of work. OpenAI realize the environmental standardization and comparison of algorithms with the help of toolkit OpenAI Gym [Bro+16]. Although OpenAI Gym is not enough to complete the work in this thesis, the baseline algorithms and the environmental interface in the package greatly speed up the work. In this section, the implementation of environment and assisted methods used in bilateral negotiation will be presented.

With the help of OpenAI Gym, a bilateral negotiation environment can be developed on the top of SAOM to research reinforcement learning algorithms. OpenAI Gym implements many baseline algorithms, which can be easily tested in a custom environment.

4.2.1 Negotiation Issues

NegMAS provoides some classes and methodes to design issues flexibly. In SBE following issues are used:

Price Integer between two values, such as (10, 20)

Quantity Integer between two values, such as (1, 10)

Time Relative step between zero and maximal step.

4.2.2 Model

The model consists of five parts, environment SBE, negotiation game, negotiation mechanism, negotiator and reinforcement learning algorithms. Except for the negotiation mechanism mentioned and implemented in sections 2.2 and 2.6.2, others parts will be introduced step by step in following sections.

Entire model is shown in 4.1.

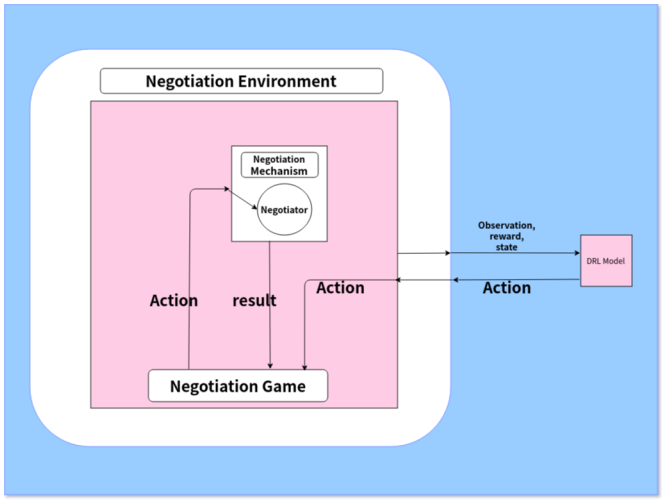

Figure 4.1: Environment for single agent bilateral negotiation based on NegMAS

4.2.3 Single-Agent Environment

The interface of the OpenAI Gym Environments is designed for one agent as standard. Nevertheless, it must be examined how the SBE can be represented via this interface and controlled by the controller. The methods of the interface for the SBE are therefore defined below.

STEP Firstly, set up but not perform the action received from RL model for the negotiator. Then, run the negotiation mechanism (such as SAOM) for one step. All actions will be perfomed by the negotiation mechanism. Finally, the function returns four parameters.

- Observation: Offer proposed by opponent and current realitive time.

- Reward: Utility value of the current offer and extra reward when an agreement is reached.
- Done: Reach the final state or there is no agreement within the maximum running time.
- Info: State of the negotiation mechanism, extra info used for evaluation.

RESET Resets the environment(SBE) and other related parameters(in Game) to an initial state after every episode and returns an initial observation.

RENDER Define how to display the output of the environment. Multiple modes can be used:

CLOSE

SEED

4.2.4 Game

In addition to implementing the official OpenAI Gym `Env` interface, class Game is designed to control the entire negotiation mechanism. The purpose of this design is to reduce the modification of negotiatioin mechanism of NegMAS.

STEP

4.2.5 Challenges of the environment

4.2.6 Analysis of the reinforcement learning algorithms

4.2.7 Conclusion

4.3 SCML with OpenAI Gym

4.3.1 Multi-Agent Environment

Official api methods, and add new method `run` to run one episode.

Step:

RESET:

CLOSE:

SEED:

4.3.2 Scenario

4.3.3 Challenges of the environment

Combination with NegMAS

4.3.4 Analysis of the reinforcement learning algorithms

4.3.5 Conclusion

5 Methods and Experiments

5.1 Single-Agent Bilateral Negotiation Environment (SBE)

In this environment, agent represents the negotiator in negotiation mechanism.

5.1.1 Independent Negotiator in NegMAS

In the environment has just single learnable DRL negotiator. All of RL algorithms can be tested in this specific environment. In the experiment of this thesis, DQN and PPO were tested in four learning scenarios:

- single issue, acceptance strategy
- single issue, offer strategy
- multi-issues, acceptance strategy
- multi-issues, offer startegy

5.1.2 Experiment

MyDRLNegotiator vs AspirationNegotiator

single issue

Negotiation mechanism is SAOM, split the learning strategy as two parts, acceptance strategy and offer strategy,

Acceptance strategy: actions of agent are `Accept offer`, `Wait` and `Reject offer`. Observation of agent are offer of opponent and current time(running time, or current step of negotiation). Algorithms are DQN (blue) and PPO (red). Mean episode reward is shown in 5.1

Offer strategy: Actions of negotiator are all outcomes set in mechanism. The observation is same as defined in the acceptance strategy. Before training the agent, normalize action and observation. Algorithm is DQN and PPO.

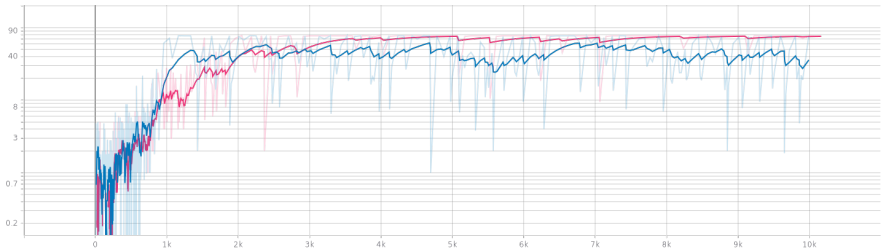

Figure 5.1: Episode mean reward of acceptance strategy under single issue

multi issues

5.1.3 Evaluation

5.2 Multi-Agent Concurrent Bilateral Negotiation Environment (MCBE)

In this environment, agent represents the factory manager and negotiation controller in standard SCML and SCML OneShot, respectively.

The environment provides all informations that are needed for reinforcement learning. Action, Observation and Reward are passed through callback functions defined in the class `Scenario`. In addition, class `Scenario` contains predefined informations, such as structure of supply chain network, parameters of learnable agents and so on.

MCBE is shown as in.

5.2.1 MADDPG in SCML

Shown in 5.2

Actors output actions as inputs to related agents interacting with the environment. Agents interacting with environment outputs the observation and reward as the inputs to related agents trained in the model.

The same agent interacting with environment may be have many related trainable agents as the part of agent(e.g. one seller, one buyer) in the model. The detail of interactive logic is shown below in 5.3

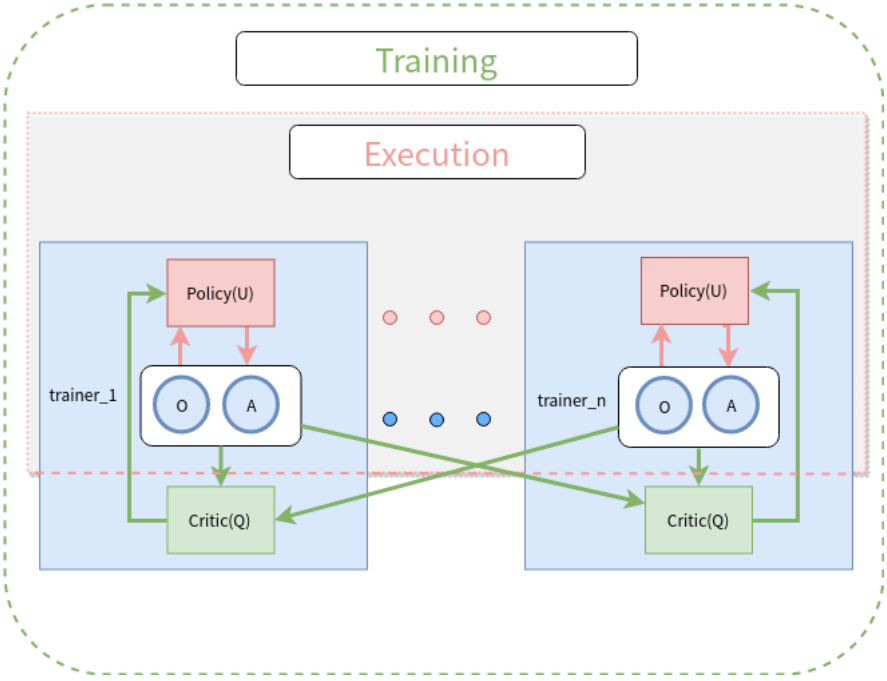

Figure 5.2: maddpg used in multi agents concurrent negotiation based on standard SCML

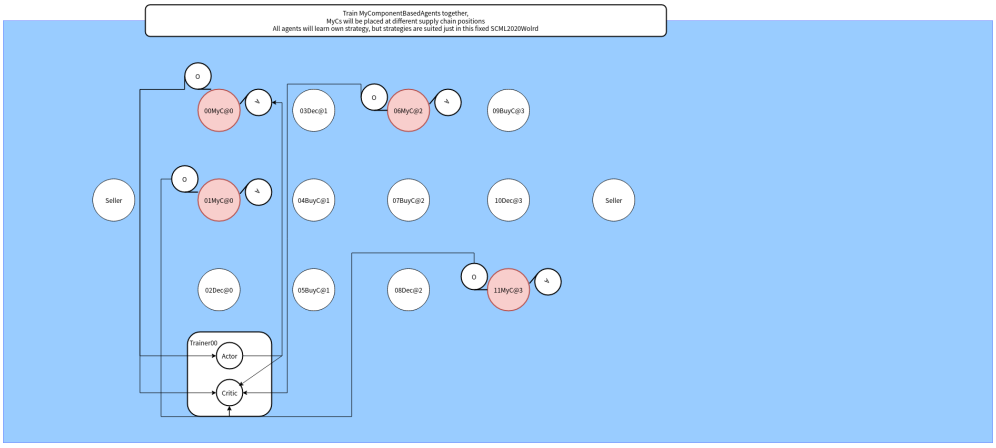

Figure 5.3: Interactive logic based on the perspective of SCML

Algorithmus 1 : How to write algorithms

Data : this text

Result : how to write algorithm with LaTeX2e

```
initialization;
while not at end of this document do
    read current;
    if understand then
        go to next section;
        current section becomes this one;
    else
        go back to the beginning of current section;
    end
end
```

5.2.2 QMIX in SCML

test

5.2.3 Experiment

Concurrent Neogtiations in standard SCML

Concurrent Negotiations in OneShot SCML

self-play

Episode mean reward curve is shown in 5.4

play with other agent

My agent vs GreedyOneShotAgent

Episode mean reward curve is shown in 5.5

5.2.4 Evaluation

5.3 Conclusion

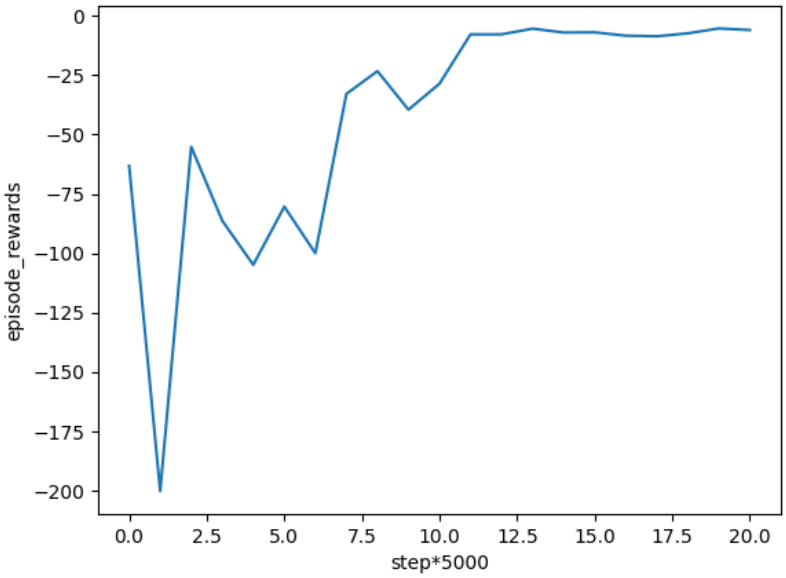

Figure 5.4: Episode mean reward of self paly under SCML OneShot

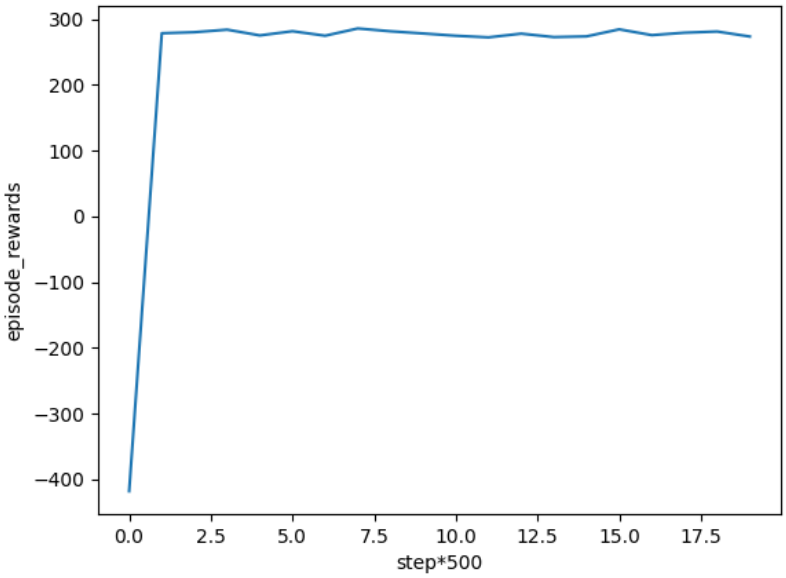

Figure 5.5: Episode mean reward of my agent vs GreedyOneShotAgent under SCML OneShot

6 Conclusions and Future Work

6.1 Others goal

In the SCM league, profit-maximizing is the goal of RL-agent, in game theory we could get many goals, such as welfare-maximization, pareto optimality. How to achieve these goals with RL-methods based on the developed environments in NegMAS and SCML?

6.2 Evaluation

Many metrics in the filed multi agent could be used for evaluating the agents proposed in this paper, such as....

6.3 Design of reward function

Reward function is an important part of realizing of RL-Agent, In the future could develop a more effective reward function, such as method proposed in the paper by [].

6.4 Complex environment

6.5 Huge scale high performance learning

Ray and reverb

Bibliography

[Bag+20] Pallavi Bagga et al. *A Deep Reinforcement Learning Approach to Concurrent Bilateral Negotiation*. 2020. arXiv: 2001.11785 [cs.MA].

[Bak+19] Jasper Bakker et al. "RLBOA: A Modular Reinforcement Learning Framework for Autonomous Negotiating Agents." In: *AAMAS*. 2019.

[Bro+16] Greg Brockman et al. *OpenAI Gym*. 2016. arXiv: 1606.01540 [cs.LG].

[Hil+18] Ashley Hill et al. *Stable Baselines*. https://github.com/hill-a/stable-baselines. 2018.

[Lin+14] Raz Lin et al. "Genius: An Integrated Environment for Supporting the Design of Generic Automated Negotiators." In: *Computational Intelligence* 30 (Feb. 2014), pp. 48–70.

[Moh+19] Yasser Mohammad et al. "Supply Chain Management World." In: Oct. 2019, pp. 153–169.

[Pas+19] Adam Paszke et al. "PyTorch: An Imperative Style, High-Performance Deep Learning Library." In: *Advances in Neural Information Processing Systems*. Ed. by H. Wallach et al. Vol. 32. Curran Associates, Inc., 2019. URL: https://proceedings.neurips.cc/paper/2019/file/bdbca288fee7f92f2bfa9f7012727740-Paper.pdf.

[Rub82] Ariel Rubinstein. "Perfect Equilibrium in A Bargaining Model." In: *Econometrica* 50 (Feb. 1982), pp. 97–109.

[SUM20] SUMAN. *Is machine learning required for deep learning?* 2020. URL: https://ai.stackexchange.com/questions/15859/is-machine-learning-required-for-deep-learning.

[Wil+12] Colin Williams et al. "Negotiating Concurrently with Unknown Opponents in Complex, Real-Time Domains." In: May 2012.

[Zho+17] L. Zhou et al. "Multiagent Reinforcement Learning With Sparse Interactions by Negotiation and Knowledge Transfer." In: *IEEE Transactions on Cybernetics* 47.5 (May 2017), pp. 1238–1250.

List of Tables

List of Figures

List of Theorems

Listings

Glossary

ANAC The International Automated Negotiating Agents Competition (ANAC) is an annual event, held in conjunction with the International Joint Conference on Autonomous Agents and Multi-Agent Systems (AAMAS), or the International Joint Conference on Artificial Intelligence (IJCAI). The ANAC competition brings together researchers from the negotiation community and provides unique benchmarks for evaluating practical negotiation strategies in multi-issue domains. The competitions have spawned novel research in AI in the field of autonomous agent design which are available to the wider research community. 9

ANEGMA ANEGMA 14

CNS Concurrent negotiation strategy. xii, 13

DQN Deep Q-Value Network 21

DRL Deep reinforcement learning. xi, 7, 17, 21

MADDPG Multi Agent Deep Deterministic Policy Gradient v, xii, 8, 22

MCBE Multi-agent concurrent bilateral negotiation environment. xii, 17, 22, 23

MDPs Markov decision process. 3

NegMAS NEGotiation MultiAgent System xi, xii, 8, 9, 17, 19, 20

NegoSI negotiation-based MARL with sparse interactions (NegoSI) 14

OpenAI Gym OpenAI Gym is a toolkit for reinforcement learning research. It includes a growing collection of benchmark problems that expose a common interface, and a website where people can share their results and compare the performance of algorithms. xi, xii, 10, 17, 19

PCA Principal Component Analysis.

PDF Portable Document Format.

PG Policy Gradient xi, 7

PPO Proximal Policy Optimization 7, 21

PyTorch Python machine learning framework, developed by... xi, 9

QMIX Monotonic Value Function Factorisation for Deep Multi-Agent Reinforcement Learning v, xii, 8, 24

Ray Ray provides a simple, universal API for building distributed applications. xi, 11

RL Reinforcement learning. 13, 17, 18, 21

RLBOA RLBOA 14

SAOM Stacked Alternating Offers Protocol, namely in SCML also as Stacked Alternating Offers Mechanism. 5, 9, 17, 18, 21

SBE Single-agent bilateral negotiation environment. xii, 17, 18, 21

SCM Supply Chain Management 1, 8, 9

SCML Supply Chain Management League one of ANAC 2020 and 2021 leagues @ IJCAI 2020 and 2021. xi, xii, 1, 4, 9, 17, 19, 22–24, 33