LMU

LUDWIG-
MAXIMILIANS-
UNIVERSITÄT
MÜNCHEN

INSTITUT FÜR INFORMATIK
LEHRSTUHL FÜR MOBILE UND VERTEILTE SYSTEME

MASTER THESIS

Exploring the impact of markets on the credit assignment problem in a multiagent environment

Zarah Zahreddin

Entwurf vom August 30, 2021

LMU LUDWIG-MAXIMILIANS-UNIVERSITÄT MÜNCHEN

INSTITUT FÜR INFORMATIK
LEHRSTUHL FÜR MOBILE UND VERTEILTE SYSTEME

MASTER THESIS

Exploring the impact of markets on the credit assignment problem in a multiagent environment

Zarah Zahreddin

Professor:	Prof. Dr. Claudia Linnhoff-Popien
Supervisor:	Kyrill Schmid Robert Müller
Submission Date:	31. December 2021

Abstract

Hier kommt der Abstract hin. Hier kommt der Abstract hin.

Contents

1 Introduction

- Motivation
- Goal
- Research Question
- Structure

2 Related Work

- Definiton of field of research
- Scientific Scope
- Which comparable work in research exists?
- Seperation from other works

2.1 Reinforcement learning

rl components

Reinforcement learning (RL) consists of the following components: an environment, one or multiple agents and time sensitive information like environment states s_t, agent actions a_t and rewards r_t. Initially an agent has no knowledge of good or bad actions and learns as it acts in an environment. Based on the executed action the environment changes and returns the new state s'_t and a reward back to the agent. The agent processes those environment signals and chooses the next action with the goal of maximizing the rewards. Hence RL is a cycle which ends once a condition is reached.

policy

Sutton and Barto, the authors of “Reinforcement learning: An introduction” [SB18], define the agents action selection with respect to the current state as a policy π. They explain further that a policy could be as simple as a lookup table, mapping states to actions or it could contain a complicated search process for the best decision. In most cases it is of stochastic nature, mapping actions and states with probabilities. During environment interactions agents gain rewards, which then can be used to update the policy accordingly. For example, should the reward, which is a number, be low or even negative it could be interpreted as a penalty. In return the policy could then be adapted to set a very low probability for that action in combination with that certain state. So next time the agent finds itself in that state the bad action is not very likely to be chosen again.

value function

While rewards only rate the immediate situation, a value function $v(s)$ can be used to estimate the long term value of a state. The resulting value is the total reward an agent could get down the line, starting from a state s. This is of importance, since an action could result in a new state with high reward but afterwards could lie a low reward streak. Or the opposite could be the case, where an action resulting in a low reward could subsequently yield high rewards. Sutton and Barto claim, that the reward estimation is a key function of RL, since the goal is to achieve as much reward as possible.

exploration vs exploitation

A strategy an agent can form after it has gained some knowledge is to execute actions it knows will result in good reward. This strategy is called exploitation. In order to achieve this knowledge an agent first needs to explore. In RL a challenge lies between harmonizing the level of exploration and exploitation.

actions that results in high rewards are always picked, leading to a greedy behavior. This leads to exploitation of the learned A common trade-off in RL is between exploitation and exploration. In order for the agent to gain a lot of reward it would make sense to always choose actions resulting in high rewards. This is often described as being greedy and exploiting the expirience the agent has gained.

2.1.1 Environment Interaction

Describe and explain the reward system -> Agent observes, takes action, env reacts and returns reward

2.1.2 Policy Optimization vs Action selection optimization

Describe and explain the reward system -> Agent observes, takes action, env reacts and returns reward

2.1.3 Multiagent settings

Describe and explain the settings, mixed/cooperation

2.2 Credit assignment problem

2.3 Markets

3 Background

- Describe the technical basis of your work
- Do not tell a historical story - make it short

3.1 Proximal Policy Optimization

intro

In 2017 Schulman et al. introduced the concept of Proximal Policy Optimization (PPO) in the article "Proximal Policy Optimization Algorithms"[SWD$^+$17]. This section is solely based on that article in order to explain the Algorithm. Policy optimization is the improvement of the action selection strategy π based on the current state s_t. This is achieved by rotating two steps: 1. sampling data from the policy and 2. optimizing that data through several epochs.

TRPO, Advantage func

The origin of PPO lies in a similar approach called Trust Region Policy Optimization (TRPO). TRPO strives to maximize the following function:

$$\underset{\theta}{maximize}\, \hat{\mathbb{E}}_t[\frac{\pi_\theta(a_t \mid s_t)}{\pi_{\theta_{old}}(a_t \mid s_t)}\hat{A}_t - \beta\, KL[\pi_{\theta_{old}}(\cdot \mid s_t), \pi_\theta(\cdot \mid s_t)]] \tag{3.1}$$

with $\hat{A}_t$ as an estimator of the advantage function. The advantage function often calculated with the state-value function $V(s)$, a reward r and a discount coefficient λ over a period of Time t

$$\hat{A}_t = -V(s_t) + r_t + \lambda r_{t+1} + \ldots + \lambda^{T-t+1} r_{T-1} + \lambda^{T-t} V(s_T) \tag{3.2}$$

The fraction in the Minuend of (3.1) can be replaced by $r(\theta)$ and represents the probability ratio of an action in the current policy in comparison to the old policy, with θ being a policy parameter. The result of $r(\theta)$ is greater than one, if an action is very probable in the current policy. Otherwise the outcome lies between zero and one. Schulman et al. further describe that TRPO maximizes the "surrogate" objective

$$L^{CPI}(\theta) = \hat{\mathbb{E}}_t[\frac{\pi_\theta(a_t \mid s_t)}{\pi_{\theta_{old}}(a_t \mid s_t)}\hat{A}_t] = \hat{\mathbb{E}}_t[r(\theta)\hat{A}_t] \tag{3.3}$$

However, maximized on its own without a penalty this results in a large outcome and leads to drastic policy updates.

problem TRPO

In order to stay in a trust region, as the name suggests, a penalty is subtracted from the surrogate function (3.3). The penalty is the Subtrahend of equation (3.1) and contains the fixed coefficient β. Regardless of the function details and outcome of KL, the coefficient β is hard to choose, since different problems require different penalty degrees. Even in a single problem it could be necessary to adapt the coefficient, due to changes within the setting.

PPO

Therefore Schulman et al. introduced

$$L^{CLIP}(\theta) = \hat{\mathbb{E}}_t[\min(r(\theta)\hat{A}_t, clip(r(\theta), 1-\epsilon, 1+\epsilon)\hat{A}_t)] \tag{3.4}$$

which is very similar to (3.1) but does not require coefficients. The first part of min contains L^{CPI} (3.3). The second part contains a *clip* function which narrows the space of policy mutation with the small hyperparameter ϵ. After applying the clip function $r(\theta)$ lies between $[1-\epsilon, 1+\epsilon]$. Calculating the minimum of the clipped and unclipped probability ratio produces the lower bound of the unclipped $r(\theta)$, preventing the policy to change drastically.

PPO Algo

PPO is defined by the following equation

$$L_t^{CLIP+VF+S}(\theta) = \hat{\mathbb{E}}_t[L_t^{CLIP}(\theta) - c_1 L_t^{VF}(\theta) + c_2 S[\pi_\theta](s_t)] \tag{3.5}$$

with c_1 and c_2 as coefficients. The authors point out that the loss function $L_t^{VF} = (V_\theta(s_t) - V_t^{targ})^2$ combines the policy surrogate and the value function error term and is necessary once a neural network shares parameters between policy and value function. Finally an entropy bonus S is added to ensure exploration. Schulman et al. continues to show an example of an Algorithm using PPO, see Fig. 3.1. N detonates (parallel) actors collecting data in T timesteps in each Iteration. Afterwards the policy is optimized in K epochs by computing the Loss function (3.5) on the corresponding NT timesteps of data, using a minibatch.

Algorithm 1 PPO, Actor-Critic Style

```
for iteration=$1, 2, \ldots$ do
    for actor=$1, 2, \ldots, N$ do
        Run policy $\pi_{\theta_{\text{old}}}$ in environment for $T$ timesteps
        Compute advantage estimates $\hat{A}_1, \ldots, \hat{A}_T$
    end for
    Optimize surrogate $L$ wrt $\theta$, with $K$ epochs and minibatch size $M \leq NT$
    $\theta_{\text{old}} \leftarrow \theta$
end for
```

Figure 3.1: Exemplary use of PPO, as shown in “Proximal Policy Optimization Algorithms”[SWD+17]

3.2 Deep Q-Learning

4 Concept

- What is your plan?
- How do you proof that it worked? -> Metric and Experiments

5 Implementation

- How exactly did you do it?
- Experiment parameters
- Experiment setup
- No need to mention framework, software libraries or tools

6 Results

- Result presentation
- Description of images and charts

7 Discussion

- Are the findings as expected?
- Why are the things as they were observed?
- New experiments that provide further insights
- Make your results more comprehensible

8 Conclusion

(Briefly summarize your work, its implications and outline future work)
- What have you done?
- How did you do it?
- What were the results?
- What does that imply?
- Future work

List of Figures

List of Tables

Listings

Bibliography

[SB18] Richard S Sutton and Andrew G Barto. *Reinforcement learning: An introduction.* MIT press, 2018.

[SWD+17] John Schulman, Filip Wolski, Prafulla Dhariwal, Alec Radford, and Oleg Klimov. Proximal policy optimization algorithms. *arXiv preprint arXiv:1707.06347*, 2017.