啤酒厂供应链管理报告

刘智琦-2300012860 [1]

Abstract

本文旨在探究并对比不同深度强化学习（DRL）算法在经典的多级供应链库存管理问题（啤酒博弈）中的表现。我们构建了一个包含三级企业的供应链仿真环境，并部署了三种不同的 DRL 策略：标准的深度 Q 网络（DQN）、改进的 Double DQN（DDQN）以及基于独立学习的多智能体 DQN（IDQN）。实验结果表明，与基线 DQN 相比，Double DQN 表现出更优的性能和学习稳定性。更有趣的是，在 IDQN 设定下，系统自发地涌现出了经典的“牛鞭效应”现象，即供应链上游企业的订单波动被显著放大。此外，我们观察到处于供应链中游的智能体在这种非合作博弈中处于劣势地位，获得的奖励最低。本研究不仅验证了 DRL 在解决复杂库存管理问题上的潜力，也揭示了多智能体系统在模拟真实世界经济现象中的价值。

1. 引言

供应链管理中的库存优化是一个经典且极具挑战性的问题。其中，“啤酒博弈”生动地展示了由于信息不对称和延迟所导致的“牛鞭效应”（Bullwhip Effect）——即需求波动沿着供应链从下游向上游逐级放大的现象，这会极大地增加库存持有成本和缺货损失。传统的库存管理方法（如（s, S）策略）往往依赖于简化的假设，难以适应复杂多变的市场环境。

近年来，以深度强化学习（DRL）为代表的人工智能技术，为解决此类复杂的动态决策问题提供了新的思路。DRL 智能体通过与环境的直接交互进行学习，无需对系统进行精确的数学建模，展现出巨大的潜力。

本研究旨在探索不同 DRL 算法在多级供应链库存管理问题上的有效性。我们提出以下研究方案：

- **研究环境**：构建一个包含三级企业（零售商、分销商、制造商）的供应链仿真环境。
- **基线模型**：训练一个标准的 DQN 智能体控制其中一个企业，而其他企业采取随机策略。
- **改进模型 1 (Double DQN)**：使用 Double DQN 算法代替标准 DQN，以缓解 Q 值过高估计问题，并与基线进行比较。
- **改进模型 2 (IDQN)**：将供应链中的每个企业都建模为一个独立的 Double DQN 智能体，使其在共享环境中同时学习。这构成了一个多智能体强化学习（MARL）系统，我们旨在观察在此设定下是否会涌现出牛鞭效应等复杂的系统行为。

通过对比这些方法的学习效率、最终策略行为和系统整体性能，我们期望为 DRL 在智慧供应链领域的应用提供有价值的见解。

2. 方法论

2.1. 供应链仿真环境

我们基于 ‘my_dqn.py‘中的 ‘Env‘类构建仿真环境。

[*]Equal contribution [1] 信息科学技术学院. Correspondence to: 刘智琦-2300012860 <2300012860@stu.pku.edu.cn>.

Proceedings of the 39th International Conference on Machine Learning, Baltimore, Maryland, USA, PMLR 162, 2022.

- **结构**: 一个由 $N = 3$ 个企业组成的线性供应链。企业 0（下游）面向外部客户，企业 i 向企业 $i+1$（上游）订货。

- **状态空间**: 每个智能体在每个时间步 t 观察到的状态 $S_{i,t}$ 是一个包含 3 个维度的向量: $[q_{i,t}, d^s_{i,t}, I_{i,t}]$，分别代表企业 i 向上游的订单量、已满足的需求量和当前库存水平。

- **动作空间**: 每个智能体的动作 $a_{i,t}$ 是其向上游企业发出的订单量，被离散化为整数区间 $[1, 30]$。

- **需求模型**: 最下游企业（企业 0）面临的需求服从泊松分布 $D_t \sim \text{Pois}(\lambda = 10)$。对于上游企业 $i > 0$，其面临的需求等于其下游企业 $i-1$ 的订单量。

- **奖励函数**: 奖励函数旨在最大化企业利润。在每个时间步，企业 i 的奖励 $R_{i,t}$ 被设计为:

$$R_{i,t} = \overbrace{p_i \cdot d^s_{i,t}}^{\text{销售收入}} - \overbrace{p_{i+1} \cdot q_{i,t}}^{\text{采购成本}} - \overbrace{h \cdot I_{i,t}}^{\text{库存成本}} - \overbrace{c \cdot (d_{i,t} - d^s_{i,t})}^{\text{缺货成本}}$$

  其中 p_i 是售价, h 是单位库存持有成本, c 是单位缺货成本。

2.2. 强化学习智能体

2.2.1. Q 网络结构

所有智能体共享相同的神经网络结构，这是一个包含两个隐藏层的多层感知机（MLP）:

- 输入层: 状态维度 (3)
- 隐藏层 1: 128 个神经元, ReLU 激活
- 隐藏层 2: 128 个神经元, ReLU 激活
- 输出层: 动作空间维度 (30)，输出每个动作的 Q 值

2.2.2. DQN 与 Double DQN

我们同时实现了标准 DQN 和 Double DQN。两者都使用了经验回放和目标网络。其核心区别在于 Q 值的计算方式。

标准 DQN 的目标 Q 值计算如下:

$$Q_{\text{target}} = r + \gamma \max_{a'} Q_{\text{target\_net}}(s', a')$$

这种方式可能导致对 Q 值的过高估计。

Double DQN 通过解耦动作选择和 Q 值评估来缓解此问题。它使用当前网络选择最优动作，然后用目标网络评估该动作的 Q 值:

$$a^* = \arg\max_{a'} Q_{\text{current\_net}}(s', a')$$
$$Q_{\text{target}} = r + \gamma Q_{\text{target\_net}}(s', a^*)$$

2.2.3. 核心代码: Double DQN 的学习过程

Double DQN 与 DQN 的主要区别体现在 ‘learn‘函数中对 ‘Q_targets_next‘的计算上。

```
class DoubleDQNAgent(DQNAgent):
    def learn(self, experiences):
        # ... (数据预处理部分省略) ...

        # Double DQN:
        # 用主网络选择最佳动作
        best_actions_next = self.q_network(
            next_states).argmax(1).unsqueeze(1)
        # 用目标网络计算该动作的Q值
        Q_targets_next = self.target_network(
            next_states).gather(1, best_actions_next)

        # 计算TD目标
        Q_targets = rewards + \
            (self.gamma * Q_targets_next * (1 - dones))

        # ... (损失计算和网络更新部分省略) ...
```

2.2.4. 独立学习的多智能体 (IDQN)

在 IDQN 设定中，供应链中的每个企业都被之为一个独立的 Double DQN 智能体。它们在同一个环境中并行运作，各自根据自己的局部观察做出决策并获得相应奖励，独立地更新自己的 Q 网络。这种“去中心化”的训练方式更贴近现实世界中供应链各方独立决策的情景。

3. 实验设置

为了保证实验结果的可复现性，我们设置了固定的随机种子 ‘set_seed(42)‘。

3.1. 训练参数

所有实验均采用以下超参数进行训练：

- 回合数 (Episodes): 2000
- 每回合最大步数 (Max Steps): 100
- 学习率 (Learning Rate): 0.001
- 折扣因子 (γ): 0.99
- 经验回放缓冲区大小: 10000
- 批大小 (Batch Size): 64
- 目标网络软更新系数 (τ): 0.001
- 探索率 (ϵ): 从 1.0 线性衰减至 0.01
- 价格列表 (P): [10, 8, 6, 4] (企业 0-2 售价及最终供应商成本)
- 库存持有成本 (h): 0.5
- 缺货成本 (c): 2

3.2. 评价方案

我们从三个维度对模型进行评估：

1. **学习曲线**：智能体在训练过程中获得的奖励变化情况，反映了学习的效率和稳定性。
2. **最终策略**：智能体学习完成后，在无探索的测试环境中的具体订购行为。
3. **综合性能**：对比不同模型在测试环境下的平均奖励、库存水平、订单量和需求满足情况。

4. 结果与分析

4.1. 学习性能对比

图 1 展示了三种策略的学习曲线。从图中可以看出，

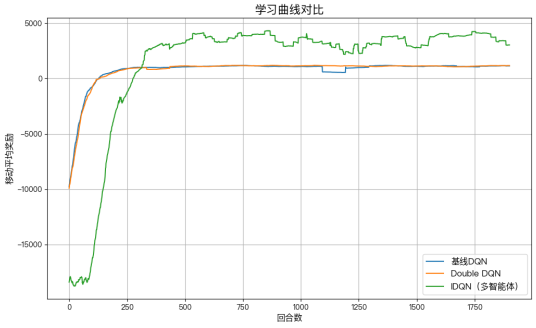

Figure 1. 不同算法的学习曲线对比。蓝线为基线 DQN，橙线为 Double DQN，绿线为 IDQN 系统的总奖励。所有曲线均为 100 个回合的移动平均值。

所有智能体都表现出明显的学习趋势。基线 DQN 和 Double DQN 的奖励曲线快速从巨大的负值（由于初始阶段大量缺货和不当库存导致的高成本）上升到稳定的正值。Double DQN 的曲线相比基线 DQN 更加平滑，最终收敛到的平均奖励也略高，验证了其算法的优越性。IDQN 的总系统奖励（所有智能体奖励之和）同样呈现稳步上升，表明多智能体系统整体上也在朝着更优的策略演进。

4.2. 策略分析：牛鞭效应的涌现

图 2 对比了 IDQN 模式下，供应链中上、中、下游三个企业在一次典型测试回合中的订购策略和它们各自面临的需求。

该图清晰地揭示了**牛鞭效应**的涌现。

- **下游（企业 0）**：其订单量（蓝色实线）基本跟

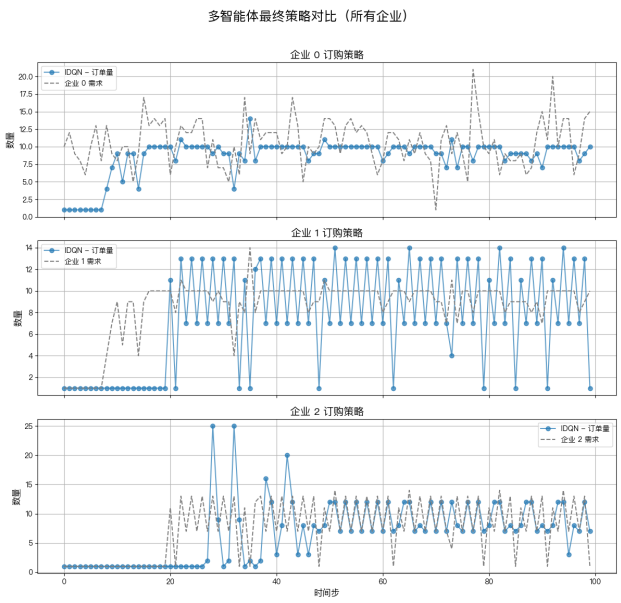

Figure 2. IDQN 模式下各企业的订购策略。从上至下分别为企业 0（下游）、企业 1（中游）、企业 2（上游）。虚线代表需求，实线代表该企业的订购量。

随其面临的外部需求（灰色虚线），波动相对较小。

- **中游（企业 1）**：其面临的需求是企业 0 的订单。可以看到，企业 1 的订单量（橙色实线）波动幅度已经明显大于其下游企业 0。
- **上游（企业 2）**：作为最上游的制造商，其订单量（绿色实线）的波动最为剧烈，远大于最原始的市场需求波动。

这种订单需求逐级放大的现象是在没有中心化协调、各智能体仅基于自身局部信息进行独立学习的情况下自发形成的，有力地模拟了真实世界中的牛鞭效应

4.3. 综合性能测试

我们对训练完成的模型进行了 10 个回合的综合测试，最终平均得分如下：

- **基线 DQN**: 1073.65
- **Double DQN**: 1180.65
- **IDQN（系统总分）**: 4933.45

Double DQN 的性能优于基线 DQN。而 IDQN 的系统总分远高于单个智能体，这符合预期，因为它是三个智能体奖励的总和。更有趣的是 IDQN 中各个智能体的得分情况。

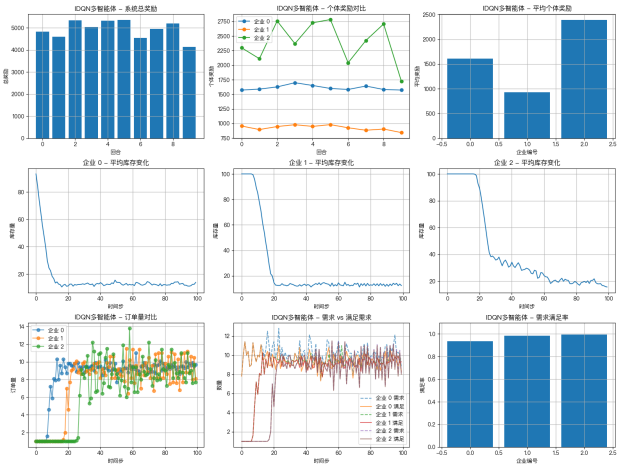

Figure 3. IDQN 多智能体系统综合测试结果。

如图 3 所示，IDQN 各企业的平均得分分别为：

- 企业 0 (下游): 1612.40
- 企业 1 (中游): 926.55
- 企业 2 (上游): 2394.50

尽管每个企业的利润空间设计为相同 (2 元)，但它们的最终收益却出现了巨大差异。上游企业（制造商）获得了最高的奖励，而中游企业（分销商）的奖励最低。这揭示了一个有趣的现象：在非合作的供应链博弈中，中间环节可能面临“两头受气”的困境，其盈利能力最弱。上游企业由于距离终端市场最远，信息最不透明，反而通过更激进的库存和订单策略获得了最高的利润。

5. 结论与启发

本研究通过对比三种 DRL 算法成功地探索了其在供应链库存管理问题中的应用。我们得出以下结论

与启发：

1. **算法有效性**：Double DQN 相较于标准 DQN，在学习稳定性和最终性能上均有可验证的提升，是更优的单智能体基线选择。

2. **牛鞭效应的模拟**：基于独立学习者的多智能体模型（IDQN）能够在没有预设规则的情况下，成功地再现供应链中的牛鞭效应。这证明了 MARL 作为一种“自下而上”的建模工具，在研究复杂经济系统涌现行为方面的巨大潜力。

3. **中间商困境**：IDQN 实验揭示了供应链中不同位置的结构性优劣势。即使利润空间相同，中间环节的企业在独立决策的模式下也可能处于最不利的地位。这为企业在供应链中的定位和合作策略提供了深刻的启示。

未来的研究可以向两个方向拓展：一是探索更先进的 MARL 算法，如引入通信机制或中心化训练-去中心化执行（CTDE）框架，以研究是否能通过协作来抑制牛鞭效应，提升整个供应链的系统总效益。二是将环境模型进一步复杂化，引入运输延迟、多产品、供应商选择等更贴近现实的因素。