RL-based Approaches for Fulfillment Policies in a Multi-Echelon Inventory System

Xiaohui Tu [11255635][1], Mingliang Wei [11274244][2]

[1,2]HEC Montréal, Canada

e-mail(s): [1]xiaohui.tu@hec.ca, [2]mingliang.wei@hec.ca

MOTIVATION OF OUR WORK

- **Application Importance** Inventory prediction is important in the supply chain management to control costs and to satisfy demands;
- **Problem Difficulty** The uncertainty from the customers, the orders from downstream, distribution from the upstream and backlog orders are changing all the time, traditional methods often fail.
- **Research Scarcity** Although reinforcement Learning is proved to be an effective tool, limited research focuses on finding out adaptive fulfillment policies in a frequently changing multi-echelon inventory system.

RELATED WORK

- **Stock-based/Age-based policy** policies are explored to deal with the system of perishable products.(REINFORCEMENT..., 2018)
- **Case-based Reinforcement Learning algorithm** CRL being used in a simplified two-echelon supply chain under the time-triggered and event triggered ordering policies.(CASE-BASED..., 2009)
- **Bullwhip Effect** RL mechanism to alleviate bullwhip effect in a multi-layer ordering control strategy.(ZHAO; SUN, 2010)
- **Agent-Based** Agent-based supply chain ordering management can be adaptable in an ever-chaning business environment.(CHAHARSOOGHI; HEYDARI; ZEGORDI, 2008)

SEQUENTIAL SUPPLY CHAIN MODEL

Model Description

- *three echelons*: manufacturer, warehouse, and retailer;
- *material flow* (solid arrow): upstream $\rightarrow$ downstream;
- *information flow* (dashed arrow): upstream $\leftarrow$ downstream;

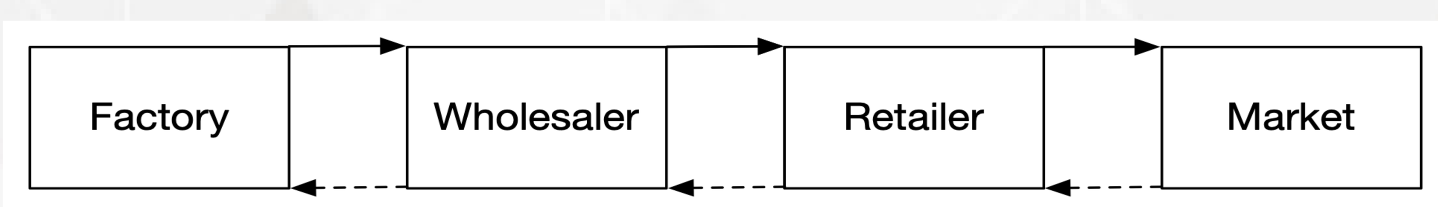


Figure 1 – Schematic diagram of a retailer inventory system

Optimization goal The objective is to maximize the total profit (*cumulated reward*) of the chain, namely the cumulated income P minus the cumulated costs C.

$$\max I = P - C = p \cdot q - (C_o^i + C_i^i + C_p^i + C_d)$$

- P: sale price: p/sale amount q
- C: ordering cost C_o^i/holding cost C_i^i/transportation cost C_p^i /shortage cost C_d

Model Assumptions

- *no lost sales*: on-hand inventory is provided if sufficient, otherwise backlogs are accumulated;
- *constant decision interval*: each manager has to place order Q_d^i at decision interval T with the upstream supplier, where $i = \{F, W, R\}$;
- *lead time*: information flow has no delays; material flow has an uncertain nominal shipment lead time LT^i following discrete uniform distribution;
- *information-separate*: the downstream and the upstream status are unknown, and decisions at echelon *i* is made with *local information*;
- *cost structure*: the ordering cost is constant and only occurs when ordering; other costs are linear to the order qualities.

Model Parameters

- decision interval: 10 days; sale price: 1000
- ordering: 80/ holding: [3, 5, 10]/ transportation: [3, 5, 10]/ backlogging: 50
- lead times: discrete uniform distribution [1, 4]
- demand: erlang distribution with a mean of 1 and a variance of 1
- maximal ordering: 30; supply level of the factory: infinity
- safety stock (used for benchmark agent): [1, 1, 7]

THEORETICAL BASICS

- *State*: costs at the end of each decision interval;
- *Action*: order quantities at the beginning of each decision interval;
- *Reward*: total profit.
- *Environment*:
  - **Input**: action;
  - **Simulation**: Within the decision interval, we update costs $\rightarrow$ order from upstream $\rightarrow$ manufacturer updates production $\rightarrow$ manufacturer updates supply $\rightarrow$ downstream updates logistics $\rightarrow$ market updates demand $\rightarrow$ market updates trade $\rightarrow$ another iteration of simulation;
  - **Return**: state, reward.

METHODOLOGY

Benchmark Agent - Traditional Prediction

Action: safety stock + target stock - inventory level at T time later;

- target stock: day $\times$ mean of the demand distribution
  - day: time needed to be covered (if we order according to the EOQ model, how many days can be covered) + lead time (mean of the lead time distribution)
- inventory: on-hold + upstream order at T time later - downstream order at T time later

witness: there is nothing to witness *train*: there is nothing to train

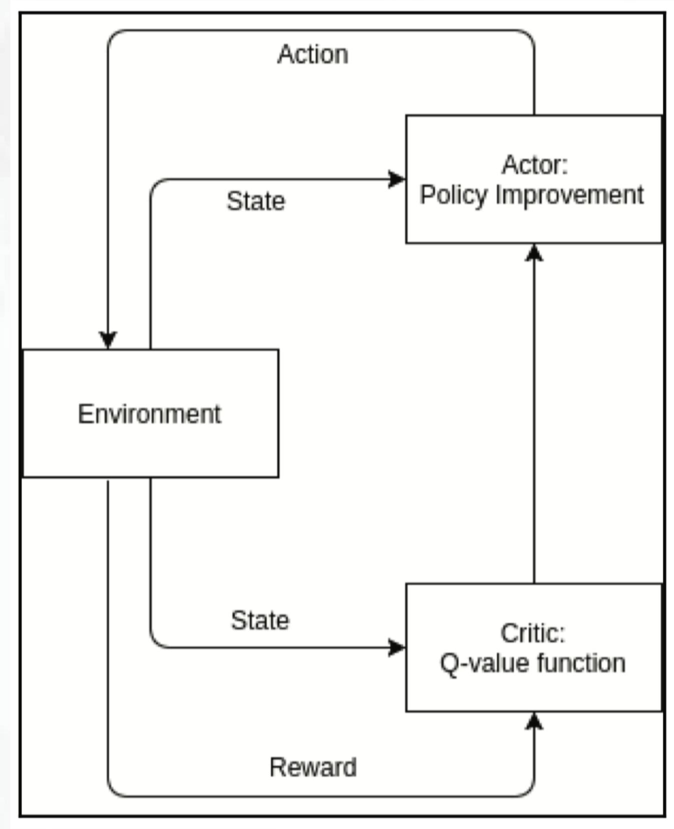
SARSA(λ) - TD Learning

- Different from Q-learning, SARSA(λ) will also update the steps lead to the reward using its eligibility trace, λ is the number of steps algorithm will take into consideration.
- By using the eligibility trace, SARSA(λ) will not be so radical as Q-learning, which is always finding the maximum $Q(s,a)$, this will make SARSA(λ) more efficient.

DDPG Agent - Actor-Critic

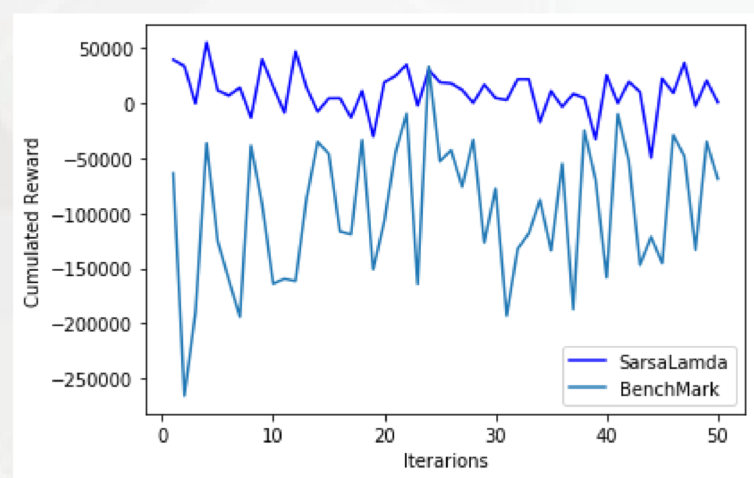
- One network acts as a **critic** $Q(s,a;w)$, which evaluates the actor's action by computing the temporal difference error and updates the weight parameter vector w;
- Other network acts as an **actor** $\mu(s;\theta)$, which performs a policy gradient to select the actions and updates the policy parameter vector θ.
- soft update the target Actor and the target Critic networks.

Figure 2 – Actor-Critic Scheme



EXPERIMENTAL RESULTS

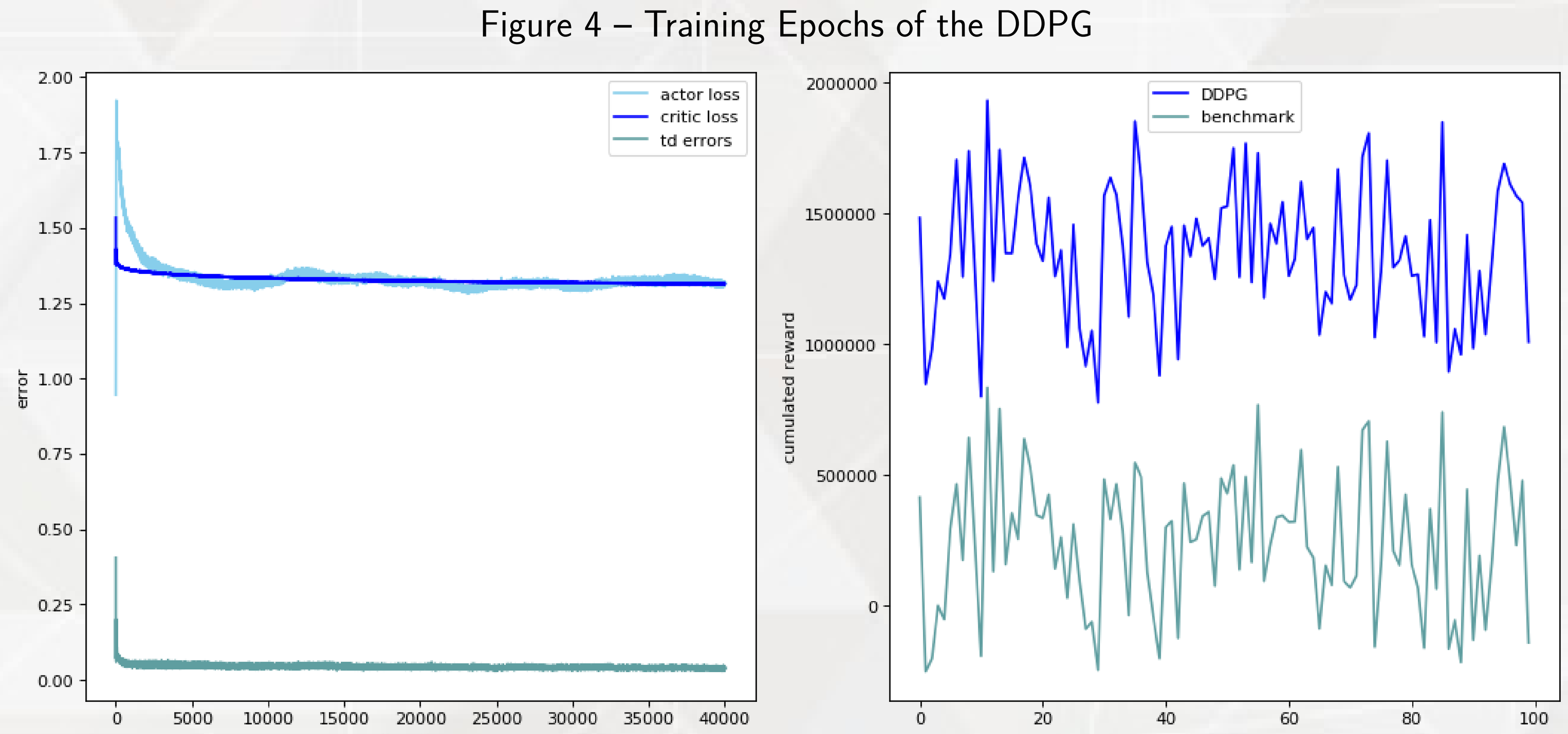
Figure 3 – Training Epochs of the Sarsa(λ)



Implemented on Python 3.7 with Tensorflow 2.0.

- Sarsa(λ)
  - Train: 50 episodes, 100 days, to shorten the training time, maximum ordering as been set as 10 (4 hours for one iteration if the maximum ordering is 30).
  - Test: The parameters are not finely tuned, so the agent has not converged, as well as not finish testing.
- DDPG
  - Train: 100 episodes, 400 days;
  - Test: 50 episodes, 800 days;
  - Test Results: The improved prediction results are not statistically significant, but the variance has decreased.

Figure 4 – Training Epochs of the DDPG



CONCLUSIONS

- **Effectiveness**: Both the Sarsa(λ) and the DDPG methods are proved to be better than the Benchmark Agent.
- **Deficiency**: Sarsa(λ) is not so effective when facing a sophisticated environment with a large number of actions and states, while the DDPG outperforms.
- **Advantage**: Errors of the DDPG decreases fast and and the network generalizes well with the current policy, but the reward is expected to increase with training.
- **Art and Science**: Parameter tuning is effort-intensive and an art of science. We are left to explore how to cover better policies.

FUTURE WORK

- Hyper-parameter tuning / Searching method improvement on Q-table
- Non-stationary evaluation / Good Algorithms to good ordering decisions

REFERENCE

CASE-BASED reinforcement learning for dynamic inventory control in a multi-agent supply-chain system. **Expert Systems with Applications**, v. 36, 3, Part 2, p. 6520–6526, 2009. ISSN 0957-4174. DOI: https://doi.org/10.1016/j.eswa.2008.07.036.

CHAHARSOOGHI, S. K.; HEYDARI, J.; ZEGORDI, S. H. A reinforcement learning model for supply chain ordering management: An application to the beer game. **Decision Support Systems**, v. 45, n. 4, p. 949–959, 2008. ISSN 0167-9236. DOI: https://doi.org/10.1016/j.dss.2008.03.007. Available from: <http://www.sciencedirect.com/science/article/pii/S0167923608000560>.

REINFORCEMENT learning approaches for specifying ordering policies of perishable inventory systems. **Expert Systems with Applications**, v. 91, p. 150–158, 2018. ISSN 0957-4174.

ZHAO, G.; SUN, R. Application of multi-agent reinforcement learning to supply chain ordering management. In: **2010 Sixth International Conference on Natural Computation**. [S.l.]: IEEE, 2010. v. 7, p. 3830–3834. ISBN 1-4244-5961-3.